

libemf

Generated by Doxygen 1.9.4

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Data Structure Index	3
2.1 Data Structures	3
3 File Index	7
3.1 File List	7
4 Data Structure Documentation	8
4.1 EMF::BRUSH Class Reference	8
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.3 Member Function Documentation	9
4.2 EMF::BYTEARRAY Struct Reference	9
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	10
4.3 EMF::CHARSTR Struct Reference	10
4.3.1 Detailed Description	11
4.3.2 Constructor & Destructor Documentation	11
4.4 EMF::DATASTREAM Class Reference	11
4.4.1 Detailed Description	12
4.4.2 Constructor & Destructor Documentation	13
4.4.3 Member Function Documentation	13
4.5 EMF::DWORDARRAY Struct Reference	29
4.5.1 Detailed Description	29
4.5.2 Constructor & Destructor Documentation	30
4.6 EMF::EMRARC Class Reference	30
4.6.1 Detailed Description	30
4.6.2 Constructor & Destructor Documentation	31
4.6.3 Member Function Documentation	31
4.7 EMF::EMRARCTO Class Reference	32
4.7.1 Detailed Description	33
4.7.2 Constructor & Destructor Documentation	33
4.7.3 Member Function Documentation	34
4.8 EMF::EMRBEGINPATH Class Reference	35
4.8.1 Detailed Description	35
4.8.2 Constructor & Destructor Documentation	35
4.8.3 Member Function Documentation	36
4.9 EMF::EMRCLOSEFIGURE Class Reference	37
4.9.1 Detailed Description	37
4.9.2 Constructor & Destructor Documentation	37
4.9.3 Member Function Documentation	38

4.10 EMF::EMRCREATEBRUSHINDIRECT Class Reference	39
4.10.1 Detailed Description	39
4.10.2 Constructor & Destructor Documentation	39
4.10.3 Member Function Documentation	40
4.11 EMF::EMRCREATEPALETTE Class Reference	41
4.11.1 Detailed Description	41
4.11.2 Constructor & Destructor Documentation	41
4.11.3 Member Function Documentation	42
4.12 EMF::EMRCREATEPEN Class Reference	43
4.12.1 Detailed Description	43
4.12.2 Constructor & Destructor Documentation	43
4.12.3 Member Function Documentation	44
4.13 EMF::EMRDELETEOBJECT Class Reference	45
4.13.1 Detailed Description	45
4.13.2 Constructor & Destructor Documentation	45
4.13.3 Member Function Documentation	46
4.14 EMF::EMRELIPSE Class Reference	47
4.14.1 Detailed Description	47
4.14.2 Constructor & Destructor Documentation	47
4.14.3 Member Function Documentation	48
4.15 EMF::EMRENDPATH Class Reference	49
4.15.1 Detailed Description	49
4.15.2 Constructor & Destructor Documentation	49
4.15.3 Member Function Documentation	50
4.16 EMF::EMREOF Class Reference	51
4.16.1 Detailed Description	51
4.16.2 Constructor & Destructor Documentation	51
4.16.3 Member Function Documentation	52
4.17 EMF::EMREXTCREATEFONTINDIRECTW Class Reference	53
4.17.1 Detailed Description	53
4.17.2 Constructor & Destructor Documentation	53
4.17.3 Member Function Documentation	54
4.18 EMF::EMREXTCREATEPEN Class Reference	55
4.18.1 Detailed Description	55
4.18.2 Constructor & Destructor Documentation	55
4.18.3 Member Function Documentation	56
4.19 EMF::EMREXTTEXTOUTA Class Reference	57
4.19.1 Detailed Description	57
4.19.2 Constructor & Destructor Documentation	57
4.19.3 Member Function Documentation	58
4.20 EMF::EMREXTTEXTOUTW Class Reference	59
4.20.1 Detailed Description	60

4.20.2 Constructor & Destructor Documentation	60
4.20.3 Member Function Documentation	61
4.21 EMF::EMRFILLPATH Class Reference	61
4.21.1 Detailed Description	62
4.21.2 Constructor & Destructor Documentation	62
4.21.3 Member Function Documentation	62
4.22 EMF::EMRLINETO Class Reference	63
4.22.1 Detailed Description	64
4.22.2 Constructor & Destructor Documentation	64
4.22.3 Member Function Documentation	64
4.23 EMF::EMRMODIFYWORLDTRANSFORM Class Reference	65
4.23.1 Detailed Description	66
4.23.2 Constructor & Destructor Documentation	66
4.23.3 Member Function Documentation	67
4.24 EMF::EMRMOVETOEX Class Reference	67
4.24.1 Detailed Description	68
4.24.2 Constructor & Destructor Documentation	68
4.24.3 Member Function Documentation	68
4.25 EMF::EMRPOLYBEZIER Class Reference	69
4.25.1 Detailed Description	70
4.25.2 Constructor & Destructor Documentation	70
4.25.3 Member Function Documentation	71
4.26 EMF::EMRPOLYBEZIER16 Class Reference	72
4.26.1 Detailed Description	72
4.26.2 Constructor & Destructor Documentation	72
4.26.3 Member Function Documentation	73
4.27 EMF::EMRPOLYBEZIERTO Class Reference	74
4.27.1 Detailed Description	75
4.27.2 Constructor & Destructor Documentation	75
4.27.3 Member Function Documentation	76
4.28 EMF::EMRPOLYBEZIERTO16 Class Reference	77
4.28.1 Detailed Description	77
4.28.2 Constructor & Destructor Documentation	77
4.28.3 Member Function Documentation	78
4.29 EMF::EMRPOLYGON Class Reference	79
4.29.1 Detailed Description	80
4.29.2 Constructor & Destructor Documentation	80
4.29.3 Member Function Documentation	80
4.30 EMF::EMRPOLYGON16 Class Reference	81
4.30.1 Detailed Description	82
4.30.2 Constructor & Destructor Documentation	82
4.30.3 Member Function Documentation	83

4.31 EMF::EMRPOLYLINE Class Reference	84
4.31.1 Detailed Description	84
4.31.2 Constructor & Destructor Documentation	84
4.31.3 Member Function Documentation	85
4.32 EMF::EMRPOLYLINE16 Class Reference	87
4.32.1 Detailed Description	87
4.32.2 Constructor & Destructor Documentation	87
4.32.3 Member Function Documentation	88
4.33 EMF::EMRPOLYLINETO Class Reference	89
4.33.1 Detailed Description	90
4.33.2 Constructor & Destructor Documentation	90
4.33.3 Member Function Documentation	91
4.34 EMF::EMRPOLYLINETO16 Class Reference	92
4.34.1 Detailed Description	92
4.34.2 Constructor & Destructor Documentation	92
4.34.3 Member Function Documentation	93
4.35 EMF::EMRPOLYPOLYGON Class Reference	94
4.35.1 Detailed Description	95
4.35.2 Constructor & Destructor Documentation	95
4.35.3 Member Function Documentation	95
4.36 EMF::EMRPOLYPOLYGON16 Class Reference	96
4.36.1 Detailed Description	97
4.36.2 Constructor & Destructor Documentation	97
4.36.3 Member Function Documentation	98
4.37 EMF::EMRRECTANGLE Class Reference	99
4.37.1 Detailed Description	99
4.37.2 Constructor & Destructor Documentation	99
4.37.3 Member Function Documentation	100
4.38 EMF::EMRRESTOREDC Class Reference	101
4.38.1 Detailed Description	101
4.38.2 Constructor & Destructor Documentation	101
4.38.3 Member Function Documentation	103
4.39 EMF::EMRSAVEDC Class Reference	104
4.39.1 Detailed Description	104
4.39.2 Constructor & Destructor Documentation	104
4.39.3 Member Function Documentation	105
4.40 EMF::EMRSCALEVIEWPORTEXTEX Class Reference	106
4.40.1 Detailed Description	106
4.40.2 Constructor & Destructor Documentation	106
4.40.3 Member Function Documentation	107
4.41 EMF::EMRSCALEWINDOWEXTEx Class Reference	108
4.41.1 Detailed Description	108

4.41.2 Constructor & Destructor Documentation	108
4.41.3 Member Function Documentation	109
4.42 EMF::EMRSELECTOBJECT Class Reference	110
4.42.1 Detailed Description	110
4.42.2 Constructor & Destructor Documentation	110
4.42.3 Member Function Documentation	111
4.43 EMF::EMRSETBKCOLOR Class Reference	112
4.43.1 Detailed Description	112
4.43.2 Constructor & Destructor Documentation	112
4.43.3 Member Function Documentation	113
4.44 EMF::EMRSETBKMODE Class Reference	114
4.44.1 Detailed Description	114
4.44.2 Constructor & Destructor Documentation	114
4.44.3 Member Function Documentation	115
4.45 EMF::EMRSETMAPMODE Class Reference	116
4.45.1 Detailed Description	116
4.45.2 Constructor & Destructor Documentation	116
4.45.3 Member Function Documentation	117
4.46 EMF::EMRSETMETARGN Class Reference	118
4.46.1 Detailed Description	118
4.46.2 Constructor & Destructor Documentation	118
4.46.3 Member Function Documentation	119
4.47 EMF::EMRSETMITERLIMIT Class Reference	120
4.47.1 Detailed Description	120
4.47.2 Constructor & Destructor Documentation	120
4.47.3 Member Function Documentation	121
4.48 EMF::EMRSETPIXELV Class Reference	122
4.48.1 Detailed Description	122
4.48.2 Constructor & Destructor Documentation	122
4.48.3 Member Function Documentation	123
4.49 EMF::EMRSETPOLYFILLMODE Class Reference	124
4.49.1 Detailed Description	124
4.49.2 Constructor & Destructor Documentation	124
4.49.3 Member Function Documentation	125
4.50 EMF::EMRSETTEXTALIGN Class Reference	126
4.50.1 Detailed Description	126
4.50.2 Constructor & Destructor Documentation	126
4.50.3 Member Function Documentation	127
4.51 EMF::EMRSETTEXTCOLOR Class Reference	128
4.51.1 Detailed Description	128
4.51.2 Constructor & Destructor Documentation	128
4.51.3 Member Function Documentation	129

4.52 EMF::EMRSETVIEWPORTEXTEX Class Reference	130
4.52.1 Detailed Description	130
4.52.2 Constructor & Destructor Documentation	130
4.52.3 Member Function Documentation	131
4.53 EMF::EMRSETVIEWPORTORGEX Class Reference	132
4.53.1 Detailed Description	132
4.53.2 Constructor & Destructor Documentation	132
4.53.3 Member Function Documentation	133
4.54 EMF::EMRSETWINDOWEXTEX Class Reference	134
4.54.1 Detailed Description	134
4.54.2 Constructor & Destructor Documentation	134
4.54.3 Member Function Documentation	135
4.55 EMF::EMRSETWINDOWORGEX Class Reference	136
4.55.1 Detailed Description	136
4.55.2 Constructor & Destructor Documentation	136
4.55.3 Member Function Documentation	137
4.56 EMF::EMRSETWORLDTRANSFORM Class Reference	138
4.56.1 Detailed Description	138
4.56.2 Constructor & Destructor Documentation	138
4.56.3 Member Function Documentation	140
4.57 EMF::EMRSTROKEANDFILLPATH Class Reference	141
4.57.1 Detailed Description	141
4.57.2 Constructor & Destructor Documentation	141
4.57.3 Member Function Documentation	142
4.58 EMF::EMRSTROKEPATH Class Reference	143
4.58.1 Detailed Description	143
4.58.2 Constructor & Destructor Documentation	143
4.58.3 Member Function Documentation	144
4.59 EMF::ENHMETAHEADER Class Reference	145
4.59.1 Detailed Description	145
4.59.2 Constructor & Destructor Documentation	145
4.59.3 Member Function Documentation	146
4.60 EMF::EXTPEN Class Reference	147
4.60.1 Detailed Description	147
4.60.2 Constructor & Destructor Documentation	147
4.60.3 Member Function Documentation	147
4.61 EMF::FONT Class Reference	148
4.61.1 Detailed Description	149
4.61.2 Constructor & Destructor Documentation	149
4.61.3 Member Function Documentation	149
4.62 EMF::GLOBALOBJECTS Class Reference	150
4.62.1 Detailed Description	152

4.62.2 Member Function Documentation	152
4.63 EMF::GRAPHICSOBJECT Class Reference	155
4.63.1 Detailed Description	155
4.63.2 Member Function Documentation	155
4.63.3 Field Documentation	156
4.64 EMF::INTARRAY Struct Reference	156
4.64.1 Detailed Description	156
4.64.2 Constructor & Destructor Documentation	156
4.65 EMF::METAFILEDEVICECONTEXT Class Reference	157
4.65.1 Detailed Description	158
4.65.2 Constructor & Destructor Documentation	158
4.65.3 Member Function Documentation	159
4.65.4 Field Documentation	161
4.66 EMF::METARECORD Class Reference	162
4.66.1 Detailed Description	163
4.66.2 Constructor & Destructor Documentation	163
4.66.3 Member Function Documentation	163
4.67 EMF::OBJECT Class Reference	165
4.67.1 Detailed Description	165
4.67.2 Constructor & Destructor Documentation	165
4.67.3 Member Function Documentation	165
4.67.4 Field Documentation	166
4.68 EMF::PADDING Struct Reference	166
4.68.1 Detailed Description	166
4.68.2 Constructor & Destructor Documentation	166
4.69 EMF::PALETTE Class Reference	167
4.69.1 Detailed Description	167
4.69.2 Constructor & Destructor Documentation	167
4.69.3 Member Function Documentation	168
4.70 EMF::PEN Class Reference	168
4.70.1 Detailed Description	169
4.70.2 Constructor & Destructor Documentation	169
4.70.3 Member Function Documentation	169
4.71 EMF::POINT16ARRAY Struct Reference	170
4.71.1 Detailed Description	170
4.71.2 Constructor & Destructor Documentation	170
4.72 EMF::POINTLARRAY Struct Reference	171
4.72.1 Detailed Description	171
4.72.2 Constructor & Destructor Documentation	171
4.73 EMF::WCHARSTR Struct Reference	172
4.73.1 Detailed Description	172
4.73.2 Constructor & Destructor Documentation	172

5 File Documentation	173
5.1 emf.h	173
5.2 basetsd.h	173
5.3 guiddef.h	175
5.4 poppack.h	176
5.5 pshpack2.h	177
5.6 pshpack4.h	177
5.7 w16.h	177
5.8 winbase.h	178
5.9 windef.h	199
5.10 winerror.h	202
5.11 wingdi.h	224
5.12 winnt.h	262
5.13 winuser.h	317
5.14 libemf.h	365
Index	419

1 Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

EMF::BYTEARRAY	9
EMF::CHARSTR	10
EMF::DATASTREAM	11
EMF::DWORDARRAY	29
EMF::GLOBALOBJECTS	150
EMF::INTARRAY	156
EMF::METARECORD	162
EMF::EMRARC	30
EMF::EMRARCTO	32
EMF::EMRBEGINPATH	35
EMF::EMRCLOSEFIGURE	37
EMF::EMRCREATEBRUSHINDIRECT	39
EMF::EMRCREATEPALETTE	41
EMF::EMRCREATEPEN	43

EMF::EMRDELETEOBJECT	45
EMF::EMRELLIPSE	47
EMF::EMRENDPATH	49
EMF::EMREOF	51
EMF::EMREXTCREATEFONTINDIRECTW	53
EMF::EMREXTCREATEPEN	55
EMF::EMREXTTEXTOUTA	57
EMF::EMREXTTEXTOUTW	59
EMF::EMRFILLPATH	61
EMF::EMRLINETO	63
EMF::EMRMODIFYWORLDTRANSFORM	65
EMF::EMRMOVETOEX	67
EMF::EMRPOLYBEZIER	69
EMF::EMRPOLYBEZIERTO	74
EMF::EMRPOLYBEZIER16	72
EMF::EMRPOLYBEZIERTO16	77
EMF::EMRPOLYBEZIERTO	74
EMF::EMRPOLYBEZIERTO16	77
EMF::EMRPOLYGON	79
EMF::EMRPOLYGON16	81
EMF::EMRPOLYLINE	84
EMF::EMRPOLYLINE16	87
EMF::EMRPOLYLINETO	89
EMF::EMRPOLYLINETO16	92
EMF::EMRPOLYPOLYGON	94
EMF::EMRPOLYPOLYGON16	96
EMF::EMRRECTANGLE	99
EMF::EMRRESTOREDC	101
EMF::EMRSAVEDC	104
EMF::EMRSCALEVIEWPORTEXT	106
EMF::EMRSCALEWINDOWEXT	108
EMF::EMRSELECTOBJECT	110

EMF::EMRSETBKCOLOR	112
EMF::EMRSETBKMODE	114
EMF::EMRSETMAPMODE	116
EMF::EMRSETMETARGN	118
EMF::EMRSETMITERLIMIT	120
EMF::EMRSETPIXELV	122
EMF::EMRSETPOLYFILLMODE	124
EMF::EMRSETTEXTALIGN	126
EMF::EMRSETTEXTCOLOR	128
EMF::EMRSETVIEWPORTEXTEX	130
EMF::EMRSETVIEWPORTORGEX	132
EMF::EMRSETWINDOWEXTEX	134
EMF::EMRSETWINDOWORGEX	136
EMF::EMRSETWORLDTRANSFORM	138
EMF::EMRSTROKEANDFILLPATH	141
EMF::EMRSTROKEPATH	143
EMF::ENHMETAHEADER	145
EMF::OBJECT	165
EMF::GRAPHICSOBJECT	155
EMF::BRUSH	8
EMF::EXTPEN	147
EMF::FONT	148
EMF::PALETTE	167
EMF::PEN	168
EMF::METAFILEDEVICECONTEXT	157
EMF::PADDING	166
EMF::POINT16ARRAY	170
EMF::POINTLARRAY	171
EMF::WCHARSTR	172

2 Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

EMF::BRUSH	
Graphics Brush	8
EMF::BYTEARRAY	
Represent a byte array in a simple way	9
EMF::CHARSTR	
Represent an ASCII character string in a simple way	10
EMF::DATASTREAM	
Support different endian modes when reading and writing the metafile	11
EMF::DWORDARRAY	
Represent an array of double word integers in a simple way	29
EMF::EMRARC	
EMF Arc	30
EMF::EMRARCTO	
EMF Arc To	32
EMF::EMRBEGINPATH	
EMF Begin Path	35
EMF::EMRCLOSEFIGURE	
EMF Close Figure	37
EMF::EMRCREATEBRUSHINDIRECT	
EMF Brush	39
EMF::EMRCREATEPALETTE	
EMF Palette	41
EMF::EMRCREATEPEN	
EMF Pen	43
EMF::EMRDELETEOBJECT	
EMF Delete Object	45
EMF::EMRELLIPSE	
EMF Ellipse	47
EMF::EMRENDPATH	
EMF End Path	49
EMF::EMREOF	
EMF End of File Record	51
EMF::EMREXTCREATEFONTINDIRECTW	
EMF Font	53
EMF::EMREXTCREATEPEN	
EMF Extended Pen	55

EMF::EMREXTTEXTOUTA	
EMF Extended Text Output ASCII	57
EMF::EMREXTTEXTOUTW	
EMF Extended Text Output Wide character	59
EMF::EMRFILLPATH	
EMF Fill path	61
EMF::EMRLINETO	
EMF Line To	63
EMF::EMRMODIFYWORLDTRANSFORM	
EMF Modify World Transform	65
EMF::EMRMOVETOEX	
EMF MoveTo (ex)	67
EMF::EMRPOLYBEZIER	
EMF Polybezier	69
EMF::EMRPOLYBEZIER16	
EMF Polybezier16	72
EMF::EMRPOLYBEZIERTO	
EMF PolyBezierTo	74
EMF::EMRPOLYBEZIERTO16	
EMF PolyBezierTo16	77
EMF::EMRPOLYGON	
EMF Filled Polygon	79
EMF::EMRPOLYGON16	
EMF Filled Polygon16	81
EMF::EMRPOLYLINE	
EMF Polyline	84
EMF::EMRPOLYLINE16	
EMF Polyline16	87
EMF::EMRPOLYLINETO	
EMF PolylineTo	89
EMF::EMRPOLYLINETO16	
EMF PolylineTo16	92
EMF::EMRPOLYPOLYGON	
EMF Poly Polygon	94
EMF::EMRPOLYPOLYGON16	
EMF Poly Polygon16	96
EMF::EMRRECTANGLE	
EMF Rectangle	99
EMF::EMRRESTOREDC	
EMF Restore DC	101
EMF::EMRSAVEDC	
EMF Save DC	104

EMF::EMRSCALEVIEWPORTEXTEx	
EMF Scale Viewport Extents (ex)	106
EMF::EMRSCALEWINDOWEXTEx	
EMF Scale Window Extents (ex)	108
EMF::EMRSELECTOBJECT	
EMF Select Object	110
EMF::EMRSETBKCOLOR	
EMF Set Background Color	112
EMF::EMRSETBKMODE	
EMF Set Background Mode	114
EMF::EMRSETMAPMODE	
EMF Set Mapping Mode	116
EMF::EMRSETMETARGN	
EMF Set Meta Region	118
EMF::EMRSETMITERLIMIT	
EMF SetMiterLimit	120
EMF::EMRSETPIXELV	
EMF Set Pixel	122
EMF::EMRSETPOLYFILLMODE	
EMF Set the Polygon Fill Mode	124
EMF::EMRSETTEXTALIGN	
EMF Set Text Alignment	126
EMF::EMRSETTEXTCOLOR	
EMF Set Text Color	128
EMF::EMRSETVIEWPORTEXTEx	
EMF Set Viewport Extents (ex)	130
EMF::EMRSETVIEWPORTORGEX	
EMF Set Viewport Origin (ex)	132
EMF::EMRSETWINDOWEXTEx	
EMF Set Window Extent (ex)	134
EMF::EMRSETWINDOWORGEX	
EMF Set Window Origin (ex)	136
EMF::EMRSETWORLDTRANSFORM	
EMF Set World Transform	138
EMF::EMRSTROKEANDFILLPATH	
EMF Stroke and Fill path	141
EMF::EMRSTROKEPATH	
EMF Stroke path	143
EMF::ENHMETAHEADER	
Enhanced Metafile Header Record	145
EMF::EXTPEN	
Extended Graphics Pen	147

EMF::FONT	
Graphics Font	148
EMF::GLOBALOBJECTS	150
EMF::GRAPHICSOBJECT	
A global graphics object	155
EMF::INTARRAY	
Represent an array of integers in a simple way	156
EMF::METAFILEDEVICECONTEXT	
Graphics Device Context	157
EMF::METARECORD	
The base class of all metafile records	162
EMF::OBJECT	
Global GDI object	165
EMF::PADDING	
All metafile records must be padded out to a multiple of 4 bytes	166
EMF::PALETTE	
Graphics Palette	167
EMF::PEN	
Graphics Pen	168
EMF::POINT16ARRAY	
Represent an array of 16-bit point in a simple way	170
EMF::POINTLARRAY	
Represent an array of points in a simple way	171
EMF::WCHARSTR	
Represent a wide (UNICODE) character string in a simple way	172

3 File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

emf.h	173
basetsd.h	173
guiddef.h	175
poppack.h	176
pshpack2.h	177
pshpack4.h	177
w16.h	177

winbase.h	178
windef.h	199
winerror.h	202
wingdi.h	224
winnt.h	262
winuser.h	317
libemf.h	365

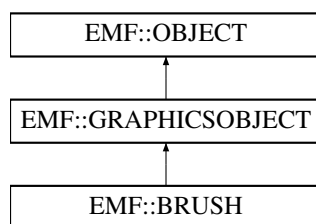
4 Data Structure Documentation

4.1 EMF::BRUSH Class Reference

Graphics Brush.

```
#include <libemf.h>
```

Inheritance diagram for EMF::BRUSH:



Public Member Functions

- [BRUSH](#) (const LOGBRUSH *lbrush)
- OBJECTTYPE [getType](#) (void) const
- [METARECORD](#) * [newEMR](#) (HDC dc, HGDIOBJ emf_handle)

Additional Inherited Members

4.1.1 Detailed Description

Graphics Brush.

Brushes are used for filling shapes.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 BRUSH() `EMF::BRUSH::BRUSH (const LOGBRUSH * lbrush) [inline]`

Parameters

<i>lbrush</i>	the (logical?) brush definition.
---------------	----------------------------------

4.1.3 Member Function Documentation

4.1.3.1 getType() OBJECTTYPE EMF::BRUSH::getType (
 void) const [inline], [virtual]

Return the type of this object (could probably do better with RTTI()).

Implements [EMF::OBJECT](#).

4.1.3.2 newEMR() METARECORD * EMF::BRUSH::newEMR (
 HDC *dc*,
 HGDIOBJ *emf_handle*) [inline], [virtual]

Return a new metarecord for this object. And record its selection into the given device context.

Parameters

<i>dc</i>	handle of device context into which this object is being selected.
<i>emf_handle</i>	the EMF handle associated with the BRUSH .

Implements [EMF::GRAPHICSOBJECT](#).

References [EMF::GRAPHICSOBJECT::contexts](#).

The documentation for this class was generated from the following file:

- libemf.h

4.2 EMF::BYTEARRAY Struct Reference

Represent a byte array in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [BYTEARRAY](#) (BYTE *const array, const int n)

Data Fields

- `BYTE *const array_`
Array of unsigned bytes.
- `const int n_`
Number of bytes in array.

4.2.1 Detailed Description

Represent a byte array in a simple way.

Evidently, an unsigned array of bytes with no particular encoding implied.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 BYTEARRAY() `EMF::BYTEARRAY::BYTEARRAY (`
 `BYTE *const array,`
 `const int n) [inline]`

Simple constructor.

Parameters

<i>array</i>	pointer to array of bytes
<i>n</i>	number of bytes in array

The documentation for this struct was generated from the following file:

- `libemf.h`

4.3 EMF::CHARSTR Struct Reference

Represent an ASCII character string in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [CHARSTR](#) (`CHAR *const string`, `const int length`)

Data Fields

- `CHAR *const string_`
Array of single byte characters.
- `const int length_`
Number of single byte characters in array.

4.3.1 Detailed Description

Represent an ASCII character string in a simple way.

ASCII strings don't have to be byte swapped, but this structure allows us to provide a uniform stream-like interface for writing out all the components of metafiles.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 CHARSTR() `EMF::CHARSTR::CHARSTR (`
`CHAR *const string,`
`const int length) [inline]`

Simple constructor.

Parameters

<i>string</i>	pointer to array of single byte characters.
<i>length</i>	number of bytes in array.

The documentation for this struct was generated from the following file:

- libemf.h

4.4 EMF::DATASTREAM Class Reference

Support different endian modes when reading and writing the metafile.

```
#include <libemf.h>
```

Public Member Functions

- [DATASTREAM](#) (::FILE *fp=0)
- void [setStream](#) (::FILE *fp)
- [DATASTREAM](#) & [operator<<](#) (const BYTE &byte)
- [DATASTREAM](#) & [operator>>](#) (BYTE &byte)
- [DATASTREAM](#) & [operator<<](#) (const WORD &word)
- [DATASTREAM](#) & [operator>>](#) (WORD &word)
- [DATASTREAM](#) & [operator<<](#) (const INT16 &word)
- [DATASTREAM](#) & [operator>>](#) (INT16 &word)
- [DATASTREAM](#) & [operator<<](#) (const DWORD &dword)
- [DATASTREAM](#) & [operator>>](#) (DWORD &dword)
- [DATASTREAM](#) & [operator<<](#) (const LONG &long_)
- [DATASTREAM](#) & [operator>>](#) (LONG &long_)
- [DATASTREAM](#) & [operator<<](#) (const INT &int_)
- [DATASTREAM](#) & [operator>>](#) (INT &int_)

- `DATASTREAM & operator<<` (`const UINT &uint`)
- `DATASTREAM & operator>>` (`UINT &uint`)
- `DATASTREAM & operator<<` (`const FLOAT &float_`)
- `DATASTREAM & operator>>` (`FLOAT &float_`)
- `DATASTREAM & operator<<` (`const PADDING &padding`)
- `DATASTREAM & operator<<` (`const RECTL &rectl`)
- `DATASTREAM & operator>>` (`RECTL &rectl`)
- `DATASTREAM & operator<<` (`const SIZEL &szel`)
- `DATASTREAM & operator>>` (`SIZEL &szel`)
- `DATASTREAM & operator<<` (`const WCHARSTR &wcharstr`)
- `DATASTREAM & operator>>` (`WCHARSTR &wcharstr`)
- `DATASTREAM & operator<<` (`const CHARSTR &charstr`)
- `DATASTREAM & operator>>` (`CHARSTR &charstr`)
- `DATASTREAM & operator<<` (`const ::EMR &emr`)
- `DATASTREAM & operator>>` (`::EMR &emr`)
- `DATASTREAM & operator<<` (`const POINT &point`)
- `DATASTREAM & operator>>` (`POINT &point`)
- `DATASTREAM & operator<<` (`const POINTL &pointl`)
- `DATASTREAM & operator>>` (`POINTL &pointl`)
- `DATASTREAM & operator<<` (`const POINT16 &point`)
- `DATASTREAM & operator>>` (`POINT16 &point`)
- `DATASTREAM & operator<<` (`const XFORM &xform`)
- `DATASTREAM & operator>>` (`XFORM &xform`)
- `DATASTREAM & operator<<` (`const BYTEARRAY &array`)
- `DATASTREAM & operator>>` (`BYTEARRAY &array`)
- `DATASTREAM & operator<<` (`const POINTLARRAY &array`)
- `DATASTREAM & operator>>` (`POINTLARRAY &array`)
- `DATASTREAM & operator<<` (`const POINT16ARRAY &array`)
- `DATASTREAM & operator>>` (`POINT16ARRAY &array`)
- `DATASTREAM & operator<<` (`const INTARRAY &array`)
- `DATASTREAM & operator>>` (`INTARRAY &array`)
- `DATASTREAM & operator<<` (`const DWORDARRAY &array`)
- `DATASTREAM & operator>>` (`DWORDARRAY &array`)
- `DATASTREAM & operator<<` (`const ::EMRTEXT &text`)
- `DATASTREAM & operator>>` (`::EMRTEXT &text`)
- `DATASTREAM & operator<<` (`const LOGPEN &pen`)
- `DATASTREAM & operator>>` (`LOGPEN &pen`)
- `DATASTREAM & operator<<` (`const EXTLOGPEN &pen`)
- `DATASTREAM & operator>>` (`EXTLOGPEN &pen`)
- `DATASTREAM & operator<<` (`const LOGBRUSH &brush`)
- `DATASTREAM & operator>>` (`LOGBRUSH &brush`)
- `DATASTREAM & operator<<` (`const LOGFONTW &font`)
- `DATASTREAM & operator>>` (`LOGFONTW &font`)
- `DATASTREAM & operator<<` (`const PANOSE &panose`)
- `DATASTREAM & operator>>` (`PANOSE &panose`)
- `DATASTREAM & operator<<` (`const EXTLOGFONTW &font`)
- `DATASTREAM & operator>>` (`EXTLOGFONTW &font`)
- `DATASTREAM & operator<<` (`const LOGPALETTE &palette`)
- `DATASTREAM & operator>>` (`LOGPALETTE &palette`)

4.4.1 Detailed Description

Support different endian modes when reading and writing the metafile.

To support different endian modes, rather than just writing the structures directly to a file via `fwrite(&emr, ...)`, we have to write each element of the structure separately, swapping bytes as necessary. `datastream` supports this. Remarkably similar to the `QDataStream` class from Qt. So, too, for reading.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 DATASTREAM() `EMF::DATASTREAM::DATASTREAM (` `::FILE * fp = 0) [inline]`

Constructor for [DATASTREAM](#).

Parameters

<i>fp</i>	optional file pointer (but must be assigned before any output occurs.)
-----------	--

4.4.3 Member Function Documentation

4.4.3.1 operator<<() [1/31] `DATASTREAM & EMF::DATASTREAM::operator<< (` `const ::EMR & emr) [inline]`

Output an Enhanced Metafile Record header.

Parameters

<i>emr</i>	Enhanced Metafile Record header to output.
------------	--

4.4.3.2 operator<<() [2/31] `DATASTREAM & EMF::DATASTREAM::operator<< (` `const ::EMRTEXT & text) [inline]`

Output an Enhanced Metafile Text Record.

Parameters

<i>text</i>	Enhanced Metafile Text Record to output.
-------------	--

4.4.3.3 operator<<() [3/31] `DATASTREAM & EMF::DATASTREAM::operator<< (` `const BYTE & byte) [inline]`

Output a byte to the stream (not swabbed or anything).

Parameters

<i>byte</i>	byte to output.
-------------	-----------------

4.4.3.4 operator<<() [4/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
const [BYTEARRAY](#) & *array*) [inline]

Output an array of BYTEs.

Parameters

<i>array</i>	array of BYTEs to output.
--------------	---------------------------

References [EMF::BYTEARRAY::array_](#), and [EMF::BYTEARRAY::n_](#).

4.4.3.5 operator<<() [5/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
const [CHARSTR](#) & *charstr*) [inline]

Output a single byte character string.

Parameters

<i>charstr</i>	structure to output.
----------------	----------------------

References [EMF::CHARSTR::length_](#), and [EMF::CHARSTR::string_](#).

4.4.3.6 operator<<() [6/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
const [DWORD](#) & *dword*) [inline]

Output a double word (long) to the stream (swabbed).

Parameters

<i>dword</i>	double word (long) to output.
--------------	-------------------------------

4.4.3.7 operator<<() [7/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
const [DWORDARRAY](#) & *array*) [inline]

Output an array of double words (longs).

Parameters

<i>array</i>	array of double words (longs) to output.
--------------	--

References [EMF::DWORDARRAY::dwords_](#), and [EMF::DWORDARRAY::n_](#).

4.4.3.8 operator<<() [8/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
 const EXTLOGFONTW & *font*) [inline]

Output an Extended Logical Font definition (using WCHAR strings).

Parameters

<i>font</i>	Extended Logical Font definition to output.
-------------	---

4.4.3.9 operator<<() [9/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
 const EXTLOGPEN & *pen*) [inline]

Output an Extended Logical Pen definition.

Parameters

<i>pen</i>	Extended Logical Pen definition to output.
------------	--

4.4.3.10 operator<<() [10/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
 const FLOAT & *float_*) [inline]

Output a single precision float to the stream (swabbed).

Parameters

<i>float</i> ↔	single precision float to output.
—	

4.4.3.11 operator<<() [11/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
 const INT & *int_*) [inline]

Output a (long) int to the stream (swabbed).

Parameters

<i>int</i> ↔	(long) int to output.
—	

4.4.3.12 `operator<<()` [12/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
 `const INT16 & word`) `[inline]`

Output a (short, 16-bit) word to the stream (swabbed).

Parameters

<i>word</i>	(short, 16-bit) word to output.
-------------	---------------------------------

4.4.3.13 `operator<<()` [13/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
 `const INTARRAY & array`) `[inline]`

Output an array of (long) ints.

Parameters

<i>array</i>	array of (long) ints to output.
--------------	---------------------------------

References [EMF::INTARRAY::ints_](#), and [EMF::INTARRAY::n_](#).

4.4.3.14 `operator<<()` [14/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
 `const LOGBRUSH & brush`) `[inline]`

Output a Logical Brush definition.

Parameters

<i>brush</i>	Logical Brush definition to output.
--------------	-------------------------------------

4.4.3.15 `operator<<()` [15/31] [DATASTREAM](#) & [EMF::DATASTREAM::operator<<](#) (
 `const LOGFONTW & font`) `[inline]`

Output a Logical Font definition (using WCHAR strings).

Parameters

<i>font</i>	Logical Font definition to output.
-------------	------------------------------------

4.4.3.16 operator<<() [16/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const LOGPALETTE & *palette*) [inline]

Output a Logical Palette.

Parameters

<i>palette</i>	Logical Palette to output.
----------------	----------------------------

4.4.3.17 operator<<() [17/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const LOGPEN & *pen*) [inline]

Output a Logical Pen definition.

Parameters

<i>pen</i>	Logical Pen definition to output.
------------	-----------------------------------

4.4.3.18 operator<<() [18/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const LONG & *long_*) [inline]

Output a long int to the stream (swabbed).

Parameters

<i>long_</i>	long int to output.
--------------	---------------------

4.4.3.19 operator<<() [19/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const [PADDING](#) & *padding*) [inline]

Output a series of '\0's to pad out a record.

Parameters

<i>padding</i>	simple padding structure (length and number of nulls).
----------------	--

References [EMF::PADDING::padding_](#), and [EMF::PADDING::size_](#).

4.4.3.20 operator<<() [20/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const PANOSE & *panose*) [inline]

Output a Panose structure.

Parameters

<i>panose</i>	Panose structure to output.
---------------	-----------------------------

4.4.3.21 operator<<() [21/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const POINT & *point*) [inline]

Output a POINT structure.

Parameters

<i>point</i>	POINT to output.
--------------	------------------

4.4.3.22 operator<<() [22/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const POINT16 & *point*) [inline]

Output a POINT16 structure.

Parameters

<i>point</i>	POINT16 to output.
--------------	--------------------

4.4.3.23 operator<<() [23/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const [POINT16ARRAY](#) & *array*) [inline]

Output an array of POINT16s.

Parameters

<i>array</i>	array of POINT16s to output.
--------------	------------------------------

References [EMF::POINT16ARRAY::n_](#), and [EMF::POINT16ARRAY::points_](#).

4.4.3.24 operator<<() [24/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const POINTL & *pointl*) [inline]

Output a POINTL structure.

Parameters

<i>pointl</i>	POINTL to output.
---------------	-------------------

4.4.3.25 operator<<() [25/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const [POINTLARRAY](#) & array) [inline]

Output an array of POINTLs.

Parameters

<i>array</i>	array of POINTLs to output.
--------------	-----------------------------

References [EMF::POINTLARRAY::n_](#), and [EMF::POINTLARRAY::points_](#).

4.4.3.26 operator<<() [26/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const RECTL & rectl) [inline]

Output a RECTL structure.

Parameters

<i>rectl</i>	structure to output.
--------------	----------------------

4.4.3.27 operator<<() [27/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const SIZEL & sizel) [inline]

Output a SIZEL structure.

Parameters

<i>sizel</i>	structure to output.
--------------	----------------------

4.4.3.28 operator<<() [28/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const UINT & uint) [inline]

Output a (long) unsigned int to the stream (swabbed).

Parameters

<i>uint</i>	(long) unsigned int to output.
-------------	--------------------------------

4.4.3.29 operator<<() [29/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const [WCHARSTR](#) & *wcharstr*) [inline]

Output a WCHAR string (note: the individual characters are swabbed).

Parameters

<i>wcharstr</i>	structure to output.
-----------------	----------------------

References [EMF::WCHARSTR::length_](#), and [EMF::WCHARSTR::string_](#).

4.4.3.30 operator<<() [30/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const WORD & *word*) [inline]

Output a (short) word to the stream (swabbed).

Parameters

<i>word</i>	(short) word to output.
-------------	-------------------------

4.4.3.31 operator<<() [31/31] [DATASTREAM](#) & EMF::DATASTREAM::operator<< (
const XFORM & *xform*) [inline]

Output an XFORM structure.

Parameters

<i>xform</i>	XFORM to output.
--------------	------------------

4.4.3.32 operator>>() [1/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
::EMR & *emr*) [inline]

Input an Enhanced Metafile Record header.

Parameters

<i>emr</i>	destination of Enhanced Metafile Record header.
------------	---

4.4.3.33 operator>>() [2/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 ::EMRTEXT & *text*) [inline]

Input an Enhanced Metafile Text Record.

Parameters

<i>text</i>	destination of Enhanced Metafile Text Record.
-------------	---

4.4.3.34 operator>>() [3/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 BYTE & *byte*) [inline]

Input a byte from the stream (not swabbed or anything).

Parameters

<i>byte</i>	destination for input byte.
-------------	-----------------------------

4.4.3.35 operator>>() [4/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 BYTEARRAY & *array*) [inline]

Input an array of BYTES.

Parameters

<i>array</i>	destination of array of input BYTES.
--------------	--------------------------------------

References [EMF::BYTEARRAY::array_](#), and [EMF::BYTEARRAY::n_](#).

4.4.3.36 operator>>() [5/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 CHARSTR & *charstr*) [inline]

Input a single byte character string.

Parameters

<i>charstr</i>	destination of input CHAR string.
----------------	-----------------------------------

References [EMF::CHARSTR::length_](#), and [EMF::CHARSTR::string_](#).

4.4.3.37 operator>>() [6/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 DWORD & *dword*) [inline]

Input a double word (long) from the stream (swabbed).

Parameters

<i>dword</i>	destination for double word (long).
--------------	-------------------------------------

4.4.3.38 operator>>() [7/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 DWORDARRAY & *array*) [inline]

Input an array of double words (longs).

Parameters

<i>array</i>	destination of array of input double words (longs).
--------------	---

References [EMF::DWORDARRAY::dwords_](#), and [EMF::DWORDARRAY::n_](#).

4.4.3.39 operator>>() [8/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 EXTLOGFONTW & *font*) [inline]

Input an Extended Logical Font definition (using WCHAR strings).

Parameters

<i>font</i>	destination of Extended Logical Font definition.
-------------	--

4.4.3.40 operator>>() [9/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 EXTLOGPEN & *pen*) [inline]

Input an Extended Logical Pen definition.

Parameters

<i>pen</i>	destination of Extended Logical Pen definition.
------------	---

4.4.3.41 operator>>() [10/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
FLOAT & *float_*) [inline]

Input a single precision float from the stream (swabbed).

Parameters

<i>float</i> \leftrightarrow	destination for single precision float.
—	

4.4.3.42 operator>>() [11/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
INT & *int_*) [inline]

Input a (long) int from the stream (swabbed).

Parameters

<i>int</i> \leftrightarrow	destination for (long) int.
—	

4.4.3.43 operator>>() [12/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
INT16 & *word*) [inline]

Input a (short, 16-bit) word from the stream (swabbed).

Parameters

<i>word</i>	destination for (short, 16-bit) word.
-------------	---------------------------------------

4.4.3.44 operator>>() [13/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
[INTARRAY](#) & *array*) [inline]

Input an array of (long) ints.

Parameters

<i>array</i>	destination of array of input (long) ints.
--------------	--

References [EMF::INTARRAY::ints_](#), and [EMF::INTARRAY::n_](#).

4.4.3.45 operator>>() [14/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (LOGBRUSH & *brush*) [inline]

Input a Logical Brush definition.

Parameters

<i>brush</i>	destination of Logical Brush definition.
--------------	--

4.4.3.46 operator>>() [15/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (LOGFONTW & *font*) [inline]

Input a Logical Font definition (using WCHAR strings).

Parameters

<i>font</i>	destination of Logical Font definition.
-------------	---

4.4.3.47 operator>>() [16/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (LOGPALETTE & *palette*) [inline]

Input a Logical Palette.

Parameters

<i>palette</i>	destination of input Logical Palette.
----------------	---------------------------------------

4.4.3.48 operator>>() [17/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (LOGPEN & *pen*) [inline]

Input a Logical Pen definition.

Parameters

<i>pen</i>	destination of Logical Pen definition.
------------	--

4.4.3.49 operator>>() [18/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (LONG & *long_*) [inline]

Input a long int from the stream (swabbed).

Parameters

<i>long</i> \leftrightarrow	destination for long int.
—	

4.4.3.50 operator>>() [19/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (PANOSE & *panose*) [inline]

Input a Panose structure.

Parameters

<i>panose</i>	destinatino of input Panose structure.
---------------	--

4.4.3.51 operator>>() [20/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (POINT & *point*) [inline]

Input a POINT structure.

Parameters

<i>point</i>	destination of input POINT.
--------------	-----------------------------

4.4.3.52 operator>>() [21/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (POINT16 & *point*) [inline]

Input a POINT16 structure.

Parameters

<i>point</i>	destination of input POINT16.
--------------	-------------------------------

4.4.3.53 operator>>() [22/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> ([POINT16ARRAY](#) & *array*) [inline]

Input an array of POINT16s.

Parameters

<i>array</i>	destination of array of input POINT16s.
--------------	---

References [EMF::POINT16ARRAY::n_](#), and [EMF::POINT16ARRAY::points_](#).

4.4.3.54 operator>>() [23/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> ([POINTL](#) & *pointl*) [inline]

Input a POINTL structure.

Parameters

<i>pointl</i>	destination of input POINTL.
---------------	------------------------------

4.4.3.55 operator>>() [24/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> ([POINTLARRAY](#) & *array*) [inline]

Input an array of POINTLs.

Parameters

<i>array</i>	destination of array of input POINTLs.
--------------	--

References [EMF::POINTLARRAY::n_](#), and [EMF::POINTLARRAY::points_](#).

4.4.3.56 operator>>() [25/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> ([RECTL](#) & *rectl*) [inline]

Input a RECTL structure.

Parameters

<i>rectl</i>	destination of input RECTL.
--------------	-----------------------------

4.4.3.57 operator>>() [26/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 [SIZEL](#) & *szel*) [inline]

Input a SIZEL structure.

Parameters

<i>szel</i>	destination of input SIZEL.
-------------	-----------------------------

4.4.3.58 operator>>() [27/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 [UINT](#) & *uint*) [inline]

Input a (long) unsigned int from the stream (swabbed).

Parameters

<i>uint</i>	destination for (long) unsigned int.
-------------	--------------------------------------

4.4.3.59 operator>>() [28/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 [WCHARSTR](#) & *wcharstr*) [inline]

Input a WCHAR string (note: the individual characters are swabbed.)

Parameters

<i>wcharstr</i>	destination of input WCHAR string.
-----------------	------------------------------------

References [EMF::WCHARSTR::length_](#), and [EMF::WCHARSTR::string_](#).

4.4.3.60 operator>>() [29/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (
 [WORD](#) & *word*) [inline]

Input a (short) word from the stream (swabbed).

Parameters

<i>word</i>	destination for (short) word.
-------------	-------------------------------

4.4.3.61 operator>>() [30/30] [DATASTREAM](#) & EMF::DATASTREAM::operator>> (XFORM & *xform*) [inline]

Input an XFORM structure.

Parameters

<i>xform</i>	destination of input XFORM.
--------------	-----------------------------

4.4.3.62 setStream() void EMF::DATASTREAM::setStream (::FILE * *fp*) [inline]

Use the given FILE stream as the input/output destination.

Parameters

<i>fp</i>	file point for i/o.
-----------	---------------------

The documentation for this class was generated from the following file:

- libemf.h

4.5 EMF::DWORDARRAY Struct Reference

Represent an array of double word integers in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [DWORDARRAY](#) (DWORD *const *dwords*, const DWORD *n*)

Data Fields

- DWORD *const **dwords_**
Array of double words.
- const DWORD **n_**
Number of double words in array.

4.5.1 Detailed Description

Represent an array of double word integers in a simple way.

Allow an array of DWORD's to be written out at once.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 DWORDARRAY() `EMF::DWORDARRAY::DWORDARRAY (`
`DWORD *const dwords,`
`const DWORD n) [inline]`

simple constructor.

Parameters

<i>dwords</i>	pointer to double words.
<i>n</i>	number double words in array.

The documentation for this struct was generated from the following file:

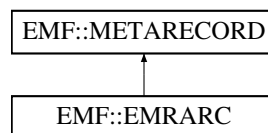
- libemf.h

4.6 EMF::EMRARC Class Reference

EMF Arc.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRARC:



Public Member Functions

- [EMRARC](#) (INT left, INT top, INT right, INT bottom, INT xstart, INT ystart, INT xend, INT yend)
- [EMRARC](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.6.1 Detailed Description

EMF Arc.

Draw an arc. Not sure what the specification here means, though.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 EMRARC() [1/2] `EMF::EMRARC::EMRARC (`
`INT left,`
`INT top,`
`INT right,`
`INT bottom,`
`INT xstart,`
`INT ystart,`
`INT xend,`
`INT yend) [inline]`

Take these descriptions with a grain of salt...

Parameters

<i>left</i>	x position of left edge of arc box.
<i>top</i>	y position of top edge of arc box.
<i>right</i>	x position of right edge of arc box.
<i>bottom</i>	y position bottom edge of arc box.
<i>xstart</i>	x position of arc start.
<i>ystart</i>	y position of arc start.
<i>xend</i>	x position of arc end.
<i>yend</i>	y position of arc end.

References [EMRARC\(\)](#).

Referenced by [EMRARC\(\)](#).

4.6.2.2 EMRARC() [2/2] `EMF::EMRARC::EMRARC (`
`DATASTREAM & ds) [inline]`

Construct an Arc record from the input datastream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.6.3 Member Function Documentation

4.6.3.1 execute() `void EMF::EMRARC::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.6.3.2 serialize() `bool EMF::EMRARC::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.6.3.3 size() `int EMF::EMRARC::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

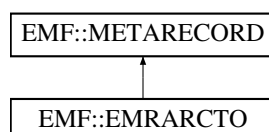
- libemf.h

4.7 EMF::EMRARCTO Class Reference

EMF Arc To.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRARCTO:



Public Member Functions

- [EMRARCTO](#) (INT left, INT top, INT right, INT bottom, INT xstart, INT ystart, INT xend, INT yend)
- [EMRARCTO](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.7.1 Detailed Description

EMF Arc To.

Draw another arc. Not sure what the specification here means, though.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 EMRARCTO() [1/2] `EMF::EMRARCTO::EMRARCTO (`
`INT left,`
`INT top,`
`INT right,`
`INT bottom,`
`INT xstart,`
`INT ystart,`
`INT xend,`
`INT yend) [inline]`

Take these descriptions with a grain of salt...

Parameters

<i>left</i>	x position of left edge of arc box.
<i>top</i>	y position of top edge of arc box.
<i>right</i>	x position of right edge of arc box.
<i>bottom</i>	y position bottom edge of arc box.
<i>xstart</i>	x position of arc start.
<i>ystart</i>	y position of arc start.
<i>xend</i>	x position of arc end.
<i>yend</i>	y position of arc end.

References [EMRARCTO\(\)](#).

Referenced by [EMRARCTO\(\)](#).

4.7.2.2 EMRARCTO() [2/2] `EMF::EMRARCTO::EMRARCTO (`
`DATASTREAM & ds) [inline]`

Construct an ArcTo record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.7.3 Member Function Documentation

4.7.3.1 execute() `void EMF::EMRARCTO::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.7.3.2 serialize() `bool EMF::EMRARCTO::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.7.3.3 size() `int EMF::EMRARCTO::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

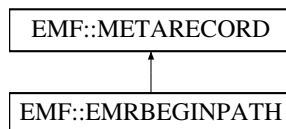
- libemf.h

4.8 EMF::EMRBEGINPATH Class Reference

EMF Begin Path.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRBEGINPATH:



Public Member Functions

- [EMRBEGINPATH](#) (void)
- [EMRBEGINPATH](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.8.1 Detailed Description

EMF Begin Path.

Begin the current path definition.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 EMRBEGINPATH() [1/2] EMF::EMRBEGINPATH::EMRBEGINPATH (void) [inline]

Create a Begin Path record.

References [EMRBEGINPATH\(\)](#).

Referenced by [EMRBEGINPATH\(\)](#).

4.8.2.2 EMRBEGINPATH() [2/2] EMF::EMRBEGINPATH::EMRBEGINPATH (DATASTREAM & ds) [inline]

Construct a BeginPath record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.8.3 Member Function Documentation

4.8.3.1 execute() `void EMF::EMRBEGINPATH::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.8.3.2 serialize() `bool EMF::EMRBEGINPATH::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.8.3.3 size() `int EMF::EMRBEGINPATH::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

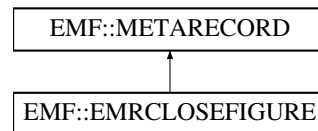
- libemf.h

4.9 EMF::EMRCLOSEFIGURE Class Reference

EMF Close Figure.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRCLOSEFIGURE:



Public Member Functions

- [EMRCLOSEFIGURE](#) (void)
- [EMRCLOSEFIGURE](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.9.1 Detailed Description

EMF Close Figure.

Close the current figure.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 EMRCLOSEFIGURE() [1/2] `EMF::EMRCLOSEFIGURE::EMRCLOSEFIGURE (void) [inline]`

Create a Close Figure record.

References [EMRCLOSEFIGURE\(\)](#).

Referenced by [EMRCLOSEFIGURE\(\)](#).

4.9.2.2 EMRCLOSEFIGURE() [2/2] `EMF::EMRCLOSEFIGURE::EMRCLOSEFIGURE (DATASTREAM & ds) [inline]`

Construct a CloseFigure record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.9.3 Member Function Documentation

4.9.3.1 execute() `void EMF::EMRCLOSEFIGURE::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.9.3.2 serialize() `bool EMF::EMRCLOSEFIGURE::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.9.3.3 size() `int EMF::EMRCLOSEFIGURE::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

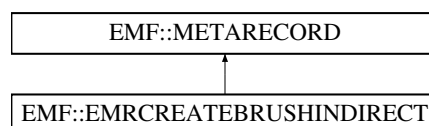
- libemf.h

4.10 EMF::EMRCREATEBRUSHINDIRECT Class Reference

EMF Brush.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRCREATEBRUSHINDIRECT:



Public Member Functions

- [EMRCREATEBRUSHINDIRECT](#) ([BRUSH](#) *brush, HGDIOBJ handle)
- [EMRCREATEBRUSHINDIRECT](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.10.1 Detailed Description

EMF Brush.

Create a new brush (used for filling shapes).

4.10.2 Constructor & Destructor Documentation

4.10.2.1 EMRCREATEBRUSHINDIRECT() [1/2] `EMRCREATEBRUSHINDIRECT::EMRCREATEBRUSHINDIRECT (
 BRUSH * brush,
 HGDIOBJ handle)`

Parameters

<i>brush</i>	an instance of a BRUSH object.
<i>handle</i>	the BRUSH object's handle.

References [EMRCREATEBRUSHINDIRECT\(\)](#).

Referenced by [EMRCREATEBRUSHINDIRECT\(\)](#).

4.10.2.2 EMRCREATEBRUSHINDIRECT() [2/2] `EMRCREATEBRUSHINDIRECT::EMRCREATEBRUSHINDIRECT (DATASTREAM & ds)`

Create a CreateBrushIndirect record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.10.3 Member Function Documentation

4.10.3.1 execute() `void EMRCREATEBRUSHINDIRECT::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.10.3.2 serialize() `bool EMF::EMRCREATEBRUSHINDIRECT::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.10.3.3 size() `int EMF::EMRCREATEBRUSHINDIRECT::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

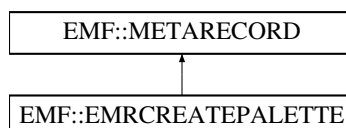
- libemf.h
- libemf.cpp

4.11 EMF::EMRCREATEPALETTE Class Reference

EMF Palette.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRCREATEPALETTE:



Public Member Functions

- [EMRCREATEPALETTE](#) ([PALETTE](#) *palette, HGDIOBJ handle)
- [EMRCREATEPALETTE](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.11.1 Detailed Description

EMF Palette.

Create a new palette.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 EMRCREATEPALETTE() [1/2] `EMRCREATEPALETTE::EMRCREATEPALETTE (
 PALETTE * palette,
 HGDIOBJ handle)`

Parameters

<i>palette</i>	an instance of a PALETTE object.
<i>handle</i>	the PALETTE object's handle.

References [EMRCREATEPALETTE\(\)](#).

Referenced by [EMRCREATEPALETTE\(\)](#).

4.11.2.2 EMRCREATEPALETTE() [2/2] `EMF::EMRCREATEPALETTE::EMRCREATEPALETTE (DATASTREAM & ds)`

Construct a CreatePalette record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.11.3 Member Function Documentation

4.11.3.1 execute() `void EMRCREATEPALETTE::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.11.3.2 serialize() `bool EMF::EMRCREATEPALETTE::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.11.3.3 size() `int EMF::EMRCREATEPALETTE::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

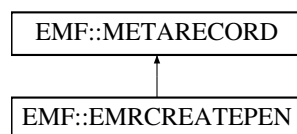
- libemf.h
- libemf.cpp

4.12 EMF::EMRCREATEPEN Class Reference

EMF Pen.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRCREATEPEN:



Public Member Functions

- [EMRCREATEPEN](#) ([PEN](#) *pen, HGDIOBJ handle)
- [EMRCREATEPEN](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.12.1 Detailed Description

EMF Pen.

Create a new pen (used for drawing lines, arcs, rectangles, etc.).

4.12.2 Constructor & Destructor Documentation

4.12.2.1 EMRCREATEPEN() [1/2] `EMRCREATEPEN::EMRCREATEPEN (
 PEN * pen,
 HGDIOBJ handle)`

Parameters

<i>pen</i>	an instance of a PEN object.
<i>handle</i>	the PEN object's handle.

References [EMRCREATEPEN\(\)](#).

Referenced by [EMRCREATEPEN\(\)](#).

4.12.2.2 EMRCREATEPEN() [2/2] `EMRCREATEPEN::EMRCREATEPEN (DATASTREAM & ds)`

Construct a CreatePen record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.12.3 Member Function Documentation

4.12.3.1 execute() `void EMRCREATEPEN::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.12.3.2 serialize() `bool EMF::EMRCREATEPEN::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.12.3.3 size() `int EMF::EMRCREATEPEN::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

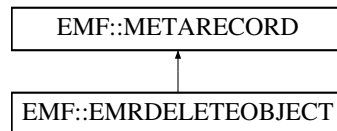
- libemf.h
- libemf.cpp

4.13 EMF::EMRDELETEOBJECT Class Reference

EMF Delete Object.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRDELETEOBJECT:



Public Member Functions

- [EMRDELETEOBJECT](#) (HGDIOBJ object)
- [EMRDELETEOBJECT](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.13.1 Detailed Description

EMF Delete Object.

Delete the given object, such as a pen, brush or font.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 EMRDELETEOBJECT() [1/2] `EMF::EMRDELETEOBJECT::EMRDELETEOBJECT (HGDIOBJ object) [inline]`

Parameters

<i>object</i>	the object to delete.
---------------	-----------------------

References [EMRDELETEOBJECT\(\)](#).

Referenced by [EMRDELETEOBJECT\(\)](#).

4.13.2.2 EMRDELETEOBJECT() [2/2] `EMF::EMRDELETEOBJECT::EMRDELETEOBJECT (DATASTREAM & ds) [inline]`

Construct a DeleteObject record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.13.3 Member Function Documentation

4.13.3.1 execute() `void EMRDELETEOBJECT::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.13.3.2 serialize() `bool EMF::EMRDELETEOBJECT::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.13.3.3 size() `int EMF::EMRDELETEOBJECT::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

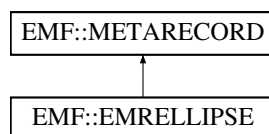
- libemf.h
- libemf.cpp

4.14 EMF::EMRELLIPSE Class Reference

EMF Ellipse.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRELLIPSE:



Public Member Functions

- [EMRELLIPSE](#) (INT left, INT top, INT right, INT bottom)
- [EMRELLIPSE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.14.1 Detailed Description

EMF Ellipse.

Draw an ellipse. (I have no idea how the ellipse is defined!)

4.14.2 Constructor & Destructor Documentation

4.14.2.1 EMRELLIPSE() [1/2] `EMF::EMRELLIPSE::EMRELLIPSE (`
`INT left,`
`INT top,`
`INT right,`
`INT bottom) [inline]`

Take these descriptions with a grain of salt...

Parameters

<i>left</i>	x position of left extrema of ellipse.
<i>top</i>	y position of top extrema of ellipse.
<i>right</i>	x position of right extrema of ellipse.
<i>bottom</i>	y position of bottom extrema of ellipse.

References [EMRELLIPSE\(\)](#).

Referenced by [EMRELLIPSE\(\)](#).

4.14.2.2 EMRELLIPSE() [2/2] `EMF::EMRELLIPSE::EMRELLIPSE (
 DATASTREAM & ds) [inline]`

Construct an Ellipse record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.14.3 Member Function Documentation

4.14.3.1 execute() `void EMF::EMRELLIPSE::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.14.3.2 serialize() `bool EMF::EMRELLIPSE::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.14.3.3 size() `int EMF::EMRELLIPSE::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

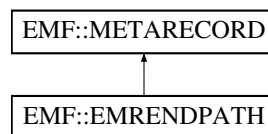
- libemf.h

4.15 EMF::EMRENDPATH Class Reference

EMF End Path.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRENDPATH:



Public Member Functions

- [EMRENDPATH](#) (void)
- [EMRENDPATH](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.15.1 Detailed Description

EMF End Path.

End the current path definition.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 EMRENDPATH() [1/2] EMF::EMRENDPATH::EMRENDPATH (void) [inline]

Create an End Path record.

References [EMRENDPATH\(\)](#).

Referenced by [EMRENDPATH\(\)](#).

4.15.2.2 EMRENDPATH() [2/2] EMF::EMRENDPATH::EMRENDPATH (DATASTREAM & ds) [inline]

Construct an EndPath record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.15.3 Member Function Documentation

4.15.3.1 execute() `void EMF::EMRENDPATH::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.15.3.2 serialize() `bool EMF::EMRENDPATH::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.15.3.3 size() `int EMF::EMRENDPATH::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

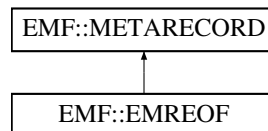
- libemf.h

4.16 EMF::EMREOF Class Reference

EMF End of File Record.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMREOF:



Public Member Functions

- [EMREOF](#) (void)
- [EMREOF](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.16.1 Detailed Description

EMF End of File Record.

Every metafile must have an End of File record. A palette may also be recorded in the EOF record, but it is currently unused by this library (all colors are specified in full RGB coordinates).

4.16.2 Constructor & Destructor Documentation

4.16.2.1 EMREOF() [1/2] `EMF::EMREOF::EMREOF (void) [inline]`

Constructor contains no user serviceable parts.

References [EMREOF\(\)](#).

Referenced by [EMREOF\(\)](#).

4.16.2.2 EMREOF() [2/2] `EMF::EMREOF::EMREOF (DATASTREAM & ds) [inline]`

Construct an EOF record from the input stream

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.16.3 Member Function Documentation

4.16.3.1 execute() `void EMF::EMREOF::execute (`
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.16.3.2 serialize() `bool EMF::EMREOF::serialize (`
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.16.3.3 size() `int EMF::EMREOF::size (`
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

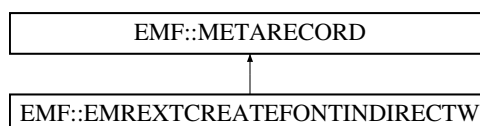
- libemf.h

4.17 EMF::EMREXTCREATEFONTINDIRECTW Class Reference

EMF Font.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMREXTCREATEFONTINDIRECTW:



Public Member Functions

- [EMREXTCREATEFONTINDIRECTW](#) ([FONT](#) *font, HGDIOBJ handle)
- [EMREXTCREATEFONTINDIRECTW](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.17.1 Detailed Description

EMF Font.

Create a new font.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 EMREXTCREATEFONTINDIRECTW() [1/2] EMREXTCREATEFONTINDIRECTW::EMREXTCREATEFONTINDIRECTW

```
(
    FONT * font,
    HGDIOBJ handle )
```

Parameters

<i>font</i>	an instance of a FONT object.
<i>handle</i>	the FONT object's handle.

References [EMREXTCREATEFONTINDIRECTW\(\)](#).

Referenced by [EMREXTCREATEFONTINDIRECTW\(\)](#).

4.17.2.2 EMREXTCREATEFONTINDIRECTW() [2/2] `EMREXTCREATEFONTINDIRECTW::EMREXTCREATEFONTINDIRECTW`
(
 [DATASTREAM](#) & *ds*)

Construct a CreateFontIndirectW record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.17.3 Member Function Documentation

4.17.3.1 execute() `void EMREXTCREATEFONTINDIRECTW::execute (`
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.17.3.2 serialize() `bool EMF::EMREXTCREATEFONTINDIRECTW::serialize (`
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.17.3.3 size() `int EMF::EMREXTCREATEFONTINDIRECTW::size (`
 void) const [inline], [virtual]

Returns

the size of the record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

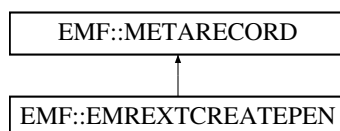
- libemf.h
- libemf.cpp

4.18 EMF::EMREXTCREATEPEN Class Reference

EMF Extended Pen.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMREXTCREATEPEN:

**Public Member Functions**

- [EMREXTCREATEPEN](#) ([EXTPEN](#) *pen, HGDIOBJ handle)
- [EMREXTCREATEPEN](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.18.1 Detailed Description

EMF Extended Pen.

Create a new pen (used for drawing lines, arcs, rectangles, etc.). Apparently uses extended attributes such as a bitmap mask.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 EMREXTCREATEPEN() [1/2] `EMREXTCREATEPEN::EMREXTCREATEPEN (
EXTPEN * pen,
HGDIOBJ handle)`

Parameters

<i>pen</i>	an instance of a PEN object.
<i>handle</i>	the PEN object's handle.

References [EMREXTCREATEPEN\(\)](#).

Referenced by [EMREXTCREATEPEN\(\)](#).

4.18.2.2 **EMREXTCREATEPEN()** [2/2] `EMREXTCREATEPEN::EMREXTCREATEPEN (DATASTREAM & ds)`

Construct a ExtCreatePen record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.18.3 Member Function Documentation

4.18.3.1 **execute()** `void EMREXTCREATEPEN::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.18.3.2 **serialize()** `bool EMF::EMREXTCREATEPEN::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.18.3.3 size() `int EMF::EMREXTCREATEPEN::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

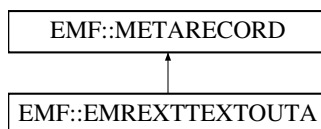
- libemf.h
- libemf.cpp

4.19 EMF::EMREXTTEXTOUTA Class Reference

EMF Extended Text Output ASCII.

`#include <libemf.h>`

Inheritance diagram for EMF::EMREXTTEXTOUTA:



Public Member Functions

- [EMREXTTEXTOUTA](#) (const RECTL *bounds, DWORD graphicsMode, FLOAT xScale, FLOAT yScale, const PEMRTEXT text, LPCSTR string, const INT *dx)
- [EMREXTTEXTOUTA](#) (DATASTREAM &ds)
- [~EMREXTTEXTOUTA](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.19.1 Detailed Description

EMF Extended Text Output ASCII.

Draw this text string with the current font, in the color of the current pen and with the given text background color. Individual character positioning can be given in the dx array.

4.19.2 Constructor & Destructor Documentation

4.19.2.1 EMREXTTEXTOUTA() [1/2] `EMF::EMREXTTEXTOUTA::EMREXTTEXTOUTA (const RECTL * bounds, DWORD graphicsMode, FLOAT xScale, FLOAT yScale, const PEMRTEXT text, LPCSTR string, const INT * dx) [inline]`

Parameters

<i>bounds</i>	bounding box of text string.
<i>graphicsMode</i>	(not entirely sure?)
<i>xScale</i>	width scale factor (of what?)
<i>yScale</i>	height scale factor (of what?)
<i>text</i>	a text metarecord containing the rendering style.
<i>string</i>	the text to render
<i>dx</i>	an array of positions for each character in string.

References [EMREXTTEXTOUTA\(\)](#).

Referenced by [EMREXTTEXTOUTA\(\)](#).

4.19.2.2 EMREXTTEXTOUTA() [2/2] `EMF::EMREXTTEXTOUTA::EMREXTTEXTOUTA (
 DATASTREAM & ds) [inline]`

Construct a ExtTextOutA record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.19.2.3 ~EMREXTTEXTOUTA() `EMF::EMREXTTEXTOUTA::~~EMREXTTEXTOUTA () [inline]`

Destructor frees its copy of the string and its character offset array

4.19.3 Member Function Documentation

4.19.3.1 execute() `void EMF::EMREXTTEXTOUTA::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.19.3.2 serialize() `bool EMF::EMREXTTEXTOUTA::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.19.3.3 size() `int EMF::EMREXTTEXTOUTA::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

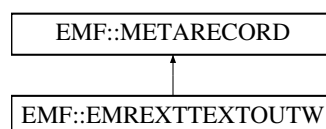
- `libemf.h`

4.20 EMF::EMREXTTEXTOUTW Class Reference

EMF Extended Text Output Wide character.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMREXTTEXTOUTW:



Public Member Functions

- [EMREXTTEXTOUTW](#) (`const RECTL *bounds, DWORD graphicsMode, FLOAT xScale, FLOAT yScale, const PEMRTEXT text, LPCWSTR string, const INT *dx`)
- [EMREXTTEXTOUTW](#) (`DATASTREAM &ds`)
- [~EMREXTTEXTOUTW](#) (`()`)
- `bool` [serialize](#) (`DATASTREAM ds`)
- `int` [size](#) (`void`) `const`
- `void` [execute](#) (`METAFILEDEVICECONTEXT *source, HDC dc`) `const`

4.20.1 Detailed Description

EMF Extended Text Output Wide character.

Draw this text string with the current font, in the color of the current pen and with the given text background color. Individual character positioning can be given in the dx array.

4.20.2 Constructor & Destructor Documentation

4.20.2.1 EMREXTTEXTOUTW() [1/2] `EMF::EMREXTTEXTOUTW::EMREXTTEXTOUTW (`
`const RECTL * bounds,`
`DWORD graphicsMode,`
`FLOAT xScale,`
`FLOAT yScale,`
`const PEMRTEXT text,`
`LPCWSTR string,`
`const INT * dx) [inline]`

Parameters

<i>bounds</i>	bounding box of text string.
<i>graphicsMode</i>	(not entirely sure?)
<i>xScale</i>	width scale factor (of what?)
<i>yScale</i>	height scale factor (of what?)
<i>text</i>	a text metarecord containing the rendering style.
<i>string</i>	the text to render
<i>dx</i>	an array of positions for each character in string.

References [EMREXTTEXTOUTW\(\)](#).

Referenced by [EMREXTTEXTOUTW\(\)](#).

4.20.2.2 EMREXTTEXTOUTW() [2/2] `EMF::EMREXTTEXTOUTW::EMREXTTEXTOUTW (`
`DATASTREAM & ds) [inline]`

Construct a ExtTextOutA record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.20.2.3 ~EMREXTTEXTOUTW() `EMF::EMREXTTEXTOUTW::~~EMREXTTEXTOUTW () [inline]`

Destructor frees its copy of the string and its character offset array

4.20.3 Member Function Documentation

4.20.3.1 execute() `void EMF::EMREXTTEXTOUTW::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.20.3.2 serialize() `bool EMF::EMREXTTEXTOUTW::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.20.3.3 size() `int EMF::EMREXTTEXTOUTW::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

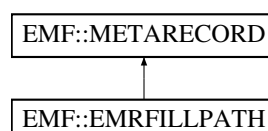
- libemf.h

4.21 EMF::EMRFILLPATH Class Reference

EMF Fill path.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRFILLPATH:



Public Member Functions

- [EMRFILLPATH](#) (const RECTL *bounds)
- [EMRFILLPATH](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.21.1 Detailed Description

EMF Fill path.

Fill the path.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 EMRFILLPATH() [1/2] `EMF::EMRFILLPATH::EMRFILLPATH (const RECTL * bounds) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polygon.
---------------	----------------------------------

References [EMRFILLPATH\(\)](#).

Referenced by [EMRFILLPATH\(\)](#).

4.21.2.2 EMRFILLPATH() [2/2] `EMF::EMRFILLPATH::EMRFILLPATH (DATASTREAM & ds) [inline]`

Create a FillPath record from input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.21.3 Member Function Documentation

4.21.3.1 execute() `void EMF::EMRFILLPATH::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.21.3.2 serialize() `bool EMF::EMRFILLPATH::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.21.3.3 size() `int EMF::EMRFILLPATH::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

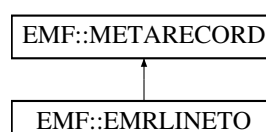
- libemf.h

4.22 EMF::EMRLINETO Class Reference

EMF Line To.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRLINETO:



Public Member Functions

- [EMRLINETO](#) (INT x, INT y)
- [EMRLINETO](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.22.1 Detailed Description

EMF Line To.

Draw a line using the current pen to the given position.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 [EMRLINETO\(\)](#) [1/2] `EMF::EMRLINETO::EMRLINETO (`
 INT x,
 INT y) [inline]

Parameters

x	x position to draw line to in logical coordinates.
y	y position to draw line to in logical coordinates.

References [EMRLINETO\(\)](#).

Referenced by [EMRLINETO\(\)](#).

4.22.2.2 [EMRLINETO\(\)](#) [2/2] `EMF::EMRLINETO::EMRLINETO (`
 [DATASTREAM](#) & ds) [inline]

Construct a LineTo record from the input stream.

Parameters

ds	Metafile datastream
----	---------------------

4.22.3 Member Function Documentation

4.22.3.1 execute() void EMF::EMRLINETO::execute (
[METAFILEDEVICECONTEXT](#) * *source*,
HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.22.3.2 serialize() bool EMF::EMRLINETO::serialize (
[DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream
-----------	---------------------

Implements [EMF::METARECORD](#).

4.22.3.3 size() int EMF::EMRLINETO::size (
void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

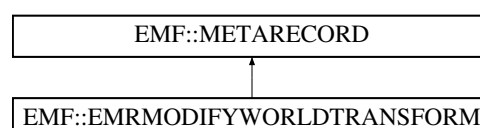
- libemf.h

4.23 EMF::EMRMODIFYWORLDTRANSFORM Class Reference

EMF Modify World Transform.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRMODIFYWORLDTRANSFORM:



Public Member Functions

- [EMRMODIFYWORLDTRANSFORM](#) (const XFORM *transform, DWORD mode)
- [EMRMODIFYWORLDTRANSFORM](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.23.1 Detailed Description

EMF Modify World Transform.

Enhanced metafiles have a Coordinate Transformation which allows the contents to be rotated and transformed. Does not appear to work properly in StarOffice (but it's also possible I don't understand how it's supposed to work either).

4.23.2 Constructor & Destructor Documentation

4.23.2.1 EMRMODIFYWORLDTRANSFORM() [1/2] `EMF::EMRMODIFYWORLDTRANSFORM::EMRMODIFYWORLDTRANSFORM`

```
(
    const XFORM * transform,
    DWORD mode ) [inline]
```

Parameters

<i>transform</i>	the transformation to apply
<i>mode</i>	the mode of the transformation application (namely, pre- or post-multiply)

References [EMRMODIFYWORLDTRANSFORM\(\)](#).

Referenced by [EMRMODIFYWORLDTRANSFORM\(\)](#).

4.23.2.2 EMRMODIFYWORLDTRANSFORM() [2/2] `EMF::EMRMODIFYWORLDTRANSFORM::EMRMODIFYWORLDTRANSFORM`

```
(
    DATASTREAM & ds ) [inline]
```

Construct a ModifyWorldTransform from the input datastream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.23.3 Member Function Documentation

4.23.3.1 execute() `void EMF::EMRMODIFYWORLDTRANSFORM::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.23.3.2 serialize() `bool EMF::EMRMODIFYWORLDTRANSFORM::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.23.3.3 size() `int EMF::EMRMODIFYWORLDTRANSFORM::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

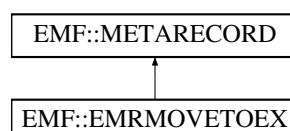
- libemf.h

4.24 EMF::EMRMOVETOEX Class Reference

EMF MoveTo (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRMOVETOEX:



Public Member Functions

- [EMRMOVETOEX](#) (INT x, INT y)
- [EMRMOVETOEX](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.24.1 Detailed Description

EMF MoveTo (ex)

Move the drawing point to the given position.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 [EMRMOVETOEX\(\)](#) [1/2] `EMF::EMRMOVETOEX::EMRMOVETOEX (`
 INT x,
 INT y) [inline]

Parameters

x	new x position in logical coordinates.
y	new y position in logical coordinates.

References [EMRMOVETOEX\(\)](#).

Referenced by [EMRMOVETOEX\(\)](#).

4.24.2.2 [EMRMOVETOEX\(\)](#) [2/2] `EMF::EMRMOVETOEX::EMRMOVETOEX (`
 DATASTREAM & ds) [inline]

Construct a MoveToEx record from the input stream.

Parameters

ds	Metafile datastream.
----	----------------------

4.24.3 Member Function Documentation

4.24.3.1 execute() void EMF::EMRMOVETOEX::execute (
[METAFILEDEVICECONTEXT](#) * *source*,
HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.24.3.2 serialize() bool EMF::EMRMOVETOEX::serialize (
[DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.24.3.3 size() int EMF::EMRMOVETOEX::size (
void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

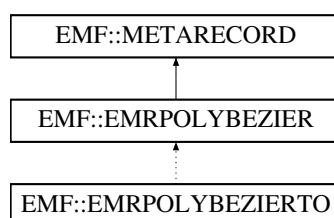
- libemf.h

4.25 EMF::EMRPOLYBEZIER Class Reference

EMF Polybezier.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYBEZIER:



Public Member Functions

- [EMRPOLYBEZIER](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYBEZIER](#) (DATASTREAM &ds)
- [~EMRPOLYBEZIER](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.25.1 Detailed Description

EMF Polybezier.

Draw a polygonal Bezier curve to (what?)

4.25.2 Constructor & Destructor Documentation

4.25.2.1 [EMRPOLYBEZIER\(\)](#) [1/2] `EMF::EMRPOLYBEZIER::EMRPOLYBEZIER (const RECTL * bounds, const POINT * points, INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIER\(\)](#).

Referenced by [EMRPOLYBEZIER\(\)](#).

4.25.2.2 [EMRPOLYBEZIER\(\)](#) [2/2] `EMF::EMRPOLYBEZIER::EMRPOLYBEZIER (DATASTREAM & ds) [inline]`

Construct a PolyBezier record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.25.2.3 [~EMRPOLYBEZIER\(\)](#) `EMF::EMRPOLYBEZIER::~~EMRPOLYBEZIER () [inline]`

Destructor frees a copy of the points it buffered.

4.25.3 Member Function Documentation

4.25.3.1 execute() `void EMF::EMRPOLYBEZIER::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO](#).

4.25.3.2 serialize() `bool EMF::EMRPOLYBEZIER::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO](#).

4.25.3.3 size() `int EMF::EMRPOLYBEZIER::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO](#).

The documentation for this class was generated from the following file:

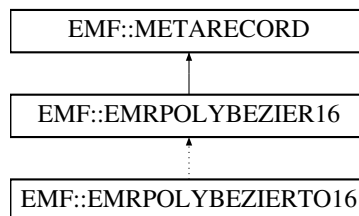
- libemf.h

4.26 EMF::EMRPOLYBEZIER16 Class Reference

EMF Polybezier16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYBEZIER16:



Public Member Functions

- [EMRPOLYBEZIER16](#) (const RECTL *bounds, const POINT16 *points, INT n)
- [EMRPOLYBEZIER16](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYBEZIER16](#) (DATASTREAM &ds)
- [~EMRPOLYBEZIER16](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.26.1 Detailed Description

EMF Polybezier16.

Draw a polygonal Bezier curve to (what?) using 16-bit points.

4.26.2 Constructor & Destructor Documentation

4.26.2.1 EMRPOLYBEZIER16() [1/3] `EMF::EMRPOLYBEZIER16::EMRPOLYBEZIER16 (const RECTL * bounds, const POINT16 * points, INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIER16\(\)](#).

Referenced by [EMRPOLYBEZIER16\(\)](#).

4.26.2.2 EMRPOLYBEZIER16() [2/3] `EMF::EMRPOLYBEZIER16::EMRPOLYBEZIER16 (`
`const RECTL * bounds,`
`const POINT * points,`
`INT n) [inline]`

Convenience constructor with POINTs.

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIER16\(\)](#).

4.26.2.3 EMRPOLYBEZIER16() [3/3] `EMF::EMRPOLYBEZIER16::EMRPOLYBEZIER16 (`
`DATASTREAM & ds) [inline]`

Construct a PolyBezier record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.26.2.4 ~EMRPOLYBEZIER16() `EMF::EMRPOLYBEZIER16::~~EMRPOLYBEZIER16 () [inline]`

Destructor frees a copy of the points it buffered.

4.26.3 Member Function Documentation

4.26.3.1 execute() `void EMF::EMRPOLYBEZIER16::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO16](#).

4.26.3.2 serialize() `bool EMF::EMRPOLYBEZIER16::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO16](#).

4.26.3.3 size() `int EMF::EMRPOLYBEZIER16::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

Reimplemented in [EMF::EMRPOLYBEZIERTO16](#).

The documentation for this class was generated from the following file:

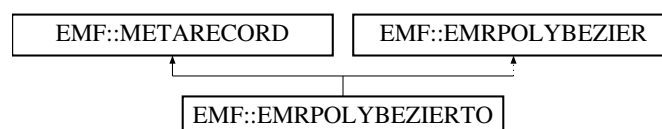
- libemf.h

4.27 EMF::EMRPOLYBEZIERTO Class Reference

EMF PolyBezierTo.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYBEZIERTO:



Public Member Functions

- [EMRPOLYBEZIERTO](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYBEZIERTO](#) (DATASTREAM &ds)
- [~EMRPOLYBEZIERTO](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.27.1 Detailed Description

EMF PolyBezierTo.

Draw a polygonal Bezier curve to (what?)

4.27.2 Constructor & Destructor Documentation

4.27.2.1 EMRPOLYBEZIERTO() [1/2] `EMF::EMRPOLYBEZIERTO::EMRPOLYBEZIERTO (const RECTL * bounds, const POINT * points, INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIERTO\(\)](#).

Referenced by [EMRPOLYBEZIERTO\(\)](#).

4.27.2.2 EMRPOLYBEZIERTO() [2/2] `EMF::EMRPOLYBEZIERTO::EMRPOLYBEZIERTO (DATASTREAM & ds) [inline]`

Construct a PolyBezier record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.27.2.3 ~EMRPOLYBEZIERTO() `EMF::EMRPOLYBEZIERTO::~~EMRPOLYBEZIERTO () [inline]`

Destructor frees a copy of the points it buffered.

4.27.3 Member Function Documentation

4.27.3.1 execute() `void EMF::EMRPOLYBEZIERTO::execute (`
 `METAFILEDEVICECONTEXT * source,`
 `HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.27.3.2 serialize() `bool EMF::EMRPOLYBEZIERTO::serialize (`
 `DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.27.3.3 size() `int EMF::EMRPOLYBEZIERTO::size (`
 `void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

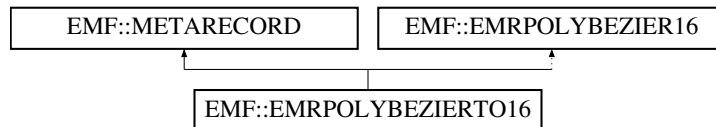
- libemf.h

4.28 EMF::EMRPOLYBEZIERTO16 Class Reference

EMF PolyBezierTo16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYBEZIERTO16:



Public Member Functions

- [EMRPOLYBEZIERTO16](#) (const RECTL *bounds, const POINT16 *points, INT n)
- [EMRPOLYBEZIERTO16](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYBEZIERTO16](#) (DATASTREAM &ds)
- [~EMRPOLYBEZIERTO16](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.28.1 Detailed Description

EMF PolyBezierTo16.

Draw a polygonal Bezier curve to (what?) using 16-bit points

4.28.2 Constructor & Destructor Documentation

4.28.2.1 EMRPOLYBEZIERTO16() [1/3] `EMF::EMRPOLYBEZIERTO16::EMRPOLYBEZIERTO16 (`
`const RECTL * bounds,`
`const POINT16 * points,`
`INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIERTO16\(\)](#).

Referenced by [EMRPOLYBEZIERTO16\(\)](#).

4.28.2.2 EMRPOLYBEZIERTO16() [2/3] `EMF::EMRPOLYBEZIERTO16::EMRPOLYBEZIERTO16 (`
`const RECTL * bounds,`
`const POINT * points,`
`INT n) [inline]`

Convenience constructor with POINTs.

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYBEZIERTO16\(\)](#).

4.28.2.3 EMRPOLYBEZIERTO16() [3/3] `EMF::EMRPOLYBEZIERTO16::EMRPOLYBEZIERTO16 (`
`DATASTREAM & ds) [inline]`

Construct a PolyBezier record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.28.2.4 ~EMRPOLYBEZIERTO16() `EMF::EMRPOLYBEZIERTO16::~~EMRPOLYBEZIERTO16 () [inline]`

Destructor frees a copy of the points it buffered.

4.28.3 Member Function Documentation

4.28.3.1 execute() `void EMF::EMRPOLYBEZIERTO16::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.28.3.2 serialize() `bool EMF::EMRPOLYBEZIERTO16::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.28.3.3 size() `int EMF::EMRPOLYBEZIERTO16::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

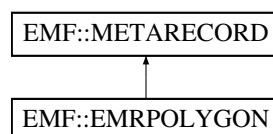
- libemf.h

4.29 EMF::EMRPOLYGON Class Reference

EMF Filled Polygon.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYGON:



Public Member Functions

- [EMRPOLYGON](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYGON](#) ([DATASTREAM](#) &ds)
- [~EMRPOLYGON](#) ()
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.29.1 Detailed Description

EMF Filled Polygon.

Draw a filled polygon.

4.29.2 Constructor & Destructor Documentation

4.29.2.1 EMRPOLYGON() [1/2] `EMF::EMRPOLYGON::EMRPOLYGON (`
 `const RECTL * bounds,`
 `const POINT * points,`
 `INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYGON\(\)](#).

Referenced by [EMRPOLYGON\(\)](#).

4.29.2.2 EMRPOLYGON() [2/2] `EMF::EMRPOLYGON::EMRPOLYGON (`
 `DATASTREAM & ds) [inline]`

Construct a Polygon record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.29.2.3 ~EMRPOLYGON() `EMF::EMRPOLYGON::~~EMRPOLYGON () [inline]`

Destructor frees a copy of the points it buffered.

4.29.3 Member Function Documentation

4.29.3.1 execute() `void EMF::EMRPOLYGON::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.29.3.2 serialize() `bool EMF::EMRPOLYGON::serialize (`
`DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.29.3.3 size() `int EMF::EMRPOLYGON::size (`
`void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

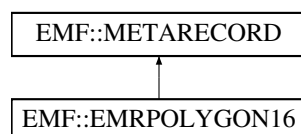
- libemf.h

4.30 EMF::EMRPOLYGON16 Class Reference

EMF Filled Polygon16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYGON16:



Public Member Functions

- [EMRPOLYGON16](#) (const RECTL *bounds, const POINT *points, INT16 n)
- [EMRPOLYGON16](#) (const RECTL *bounds, const POINT16 *points, INT16 n)
- [EMRPOLYGON16](#) (DATASTREAM &ds)
- [~EMRPOLYGON16](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.30.1 Detailed Description

EMF Filled Polygon16.

Draw a filled polygon (with 16-bit points).

4.30.2 Constructor & Destructor Documentation

4.30.2.1 [EMRPOLYGON16\(\)](#) [1/3] `EMF::EMRPOLYGON16::EMRPOLYGON16 (`
`const RECTL * bounds,`
`const POINT * points,`
`INT16 n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYGON16\(\)](#).

Referenced by [EMRPOLYGON16\(\)](#).

4.30.2.2 [EMRPOLYGON16\(\)](#) [2/3] `EMF::EMRPOLYGON16::EMRPOLYGON16 (`
`const RECTL * bounds,`
`const POINT16 * points,`
`INT16 n) [inline]`

Additional constructor which takes a POINT16 array.

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYGON16\(\)](#).

4.30.2.3 EMRPOLYGON16() [3/3] `EMF::EMRPOLYGON16::EMRPOLYGON16 (
 DATASTREAM & ds) [inline]`

Construct a Polygon record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.30.2.4 ~EMRPOLYGON16() `EMF::EMRPOLYGON16::~~EMRPOLYGON16 () [inline]`

Destructor frees a copy of the points it buffered.

4.30.3 Member Function Documentation

4.30.3.1 execute() `void EMF::EMRPOLYGON16::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.30.3.2 serialize() `bool EMF::EMRPOLYGON16::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

```
4.30.3.3 size() int EMF::EMRPOLYGON16::size (
    void ) const [inline], [virtual]
```

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

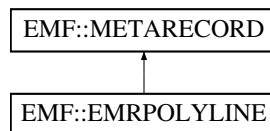
- libemf.h

4.31 EMF::EMRPOLYLINE Class Reference

EMF Polyline.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYLINE:



Public Member Functions

- [EMRPOLYLINE](#) (const RECTL *bounds, const POINT *points, INT n)
- [~EMRPOLYLINE](#) ()
- [EMRPOLYLINE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.31.1 Detailed Description

EMF Polyline.

Draw a series of connected lines.

4.31.2 Constructor & Destructor Documentation

```
4.31.2.1 EMRPOLYLINE() [1/2] EMF::EMRPOLYLINE::EMRPOLYLINE (
    const RECTL * bounds,
    const POINT * points,
    INT n ) [inline]
```

Parameters

<i>bounds</i>	overall bounding box of polyline.
<i>points</i>	array of polyline vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINE\(\)](#).

Referenced by [EMRPOLYLINE\(\)](#).

4.31.2.2 ~EMRPOLYLINE() `EMF::EMRPOLYLINE::~~EMRPOLYLINE () [inline]`

Destructor frees a copy of the points it buffered.

4.31.2.3 EMRPOLYLINE() [2/2] `EMF::EMRPOLYLINE::EMRPOLYLINE (DATASTREAM & ds) [inline]`

Construct a Polyline record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

References [EMRPOLYLINE\(\)](#).

4.31.3 Member Function Documentation

4.31.3.1 execute() `void EMF::EMRPOLYLINE::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.31.3.2 serialize() `bool EMF::EMRPOLYLINE::serialize (`
`DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.31.3.3 size() `int EMF::EMRPOLYLINE::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

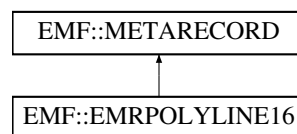
- libemf.h

4.32 EMF::EMRPOLYLINE16 Class Reference

EMF Polyline16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYLINE16:



Public Member Functions

- [EMRPOLYLINE16](#) (const RECTL *bounds, const POINT16 *points, INT n)
- [EMRPOLYLINE16](#) (const RECTL *bounds, const POINT *points, INT n)
- [~EMRPOLYLINE16](#) ()
- [EMRPOLYLINE16](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.32.1 Detailed Description

EMF Polyline16.

Draw a series of connected lines using 16-bit points.

4.32.2 Constructor & Destructor Documentation

4.32.2.1 EMRPOLYLINE16() [1/3] `EMF::EMRPOLYLINE16::EMRPOLYLINE16 (const RECTL * bounds, const POINT16 * points, INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polyline.
<i>points</i>	array of polyline vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINE16\(\)](#).

Referenced by [EMRPOLYLINE16\(\)](#).

4.32.2.2 EMRPOLYLINE16() [2/3] `EMF::EMRPOLYLINE16::EMRPOLYLINE16 (`
`const RECTL * bounds,`
`const POINT * points,`
`INT n) [inline]`

Constructor with POINTs.

Parameters

<i>bounds</i>	overall bounding box of polyline.
<i>points</i>	array of polyline vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINE16\(\)](#).

4.32.2.3 ~EMRPOLYLINE16() `EMF::EMRPOLYLINE16::~~EMRPOLYLINE16 () [inline]`

Destructor frees a copy of the points it buffered.

4.32.2.4 EMRPOLYLINE16() [3/3] `EMF::EMRPOLYLINE16::EMRPOLYLINE16 (`
`DATASTREAM & ds) [inline]`

Construct a Polyline record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

References [EMRPOLYLINE16\(\)](#).

4.32.3 Member Function Documentation

4.32.3.1 execute() `void EMF::EMRPOLYLINE16::execute (
METAFILEDEVICECONTEXT * source,
HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.32.3.2 serialize() `bool EMF::EMRPOLYLINE16::serialize (
DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.32.3.3 size() `int EMF::EMRPOLYLINE16::size (
void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

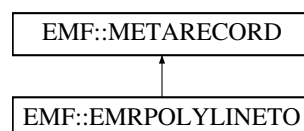
- libemf.h

4.33 EMF::EMRPOLYLINETO Class Reference

EMF PolylineTo.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYLINETO:



Public Member Functions

- [EMRPOLYLINETO](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYLINETO](#) (DATASTREAM &ds)
- [~EMRPOLYLINETO](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.33.1 Detailed Description

EMF PolylineTo.

Draw a polygonal line curve to (what?)

4.33.2 Constructor & Destructor Documentation

4.33.2.1 [EMRPOLYLINETO\(\)](#) [1/2] `EMF::EMRPOLYLINETO::EMRPOLYLINETO (`
`const RECTL * bounds,`
`const POINT * points,`
`INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINETO\(\)](#).

Referenced by [EMRPOLYLINETO\(\)](#).

4.33.2.2 [EMRPOLYLINETO\(\)](#) [2/2] `EMF::EMRPOLYLINETO::EMRPOLYLINETO (`
`DATASTREAM & ds) [inline]`

Construct a PolylineTo record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.33.2.3 [~EMRPOLYLINETO\(\)](#) `EMF::EMRPOLYLINETO::~~EMRPOLYLINETO () [inline]`

Destructor frees a copy of the points it buffered.

4.33.3 Member Function Documentation

4.33.3.1 execute() `void EMF::EMRPOLYLINETO::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.33.3.2 serialize() `bool EMF::EMRPOLYLINETO::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.33.3.3 size() `int EMF::EMRPOLYLINETO::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

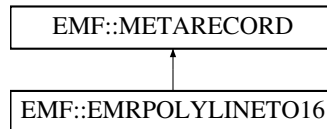
- libemf.h

4.34 EMF::EMRPOLYLINETO16 Class Reference

EMF PolylineTo16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYLINETO16:



Public Member Functions

- [EMRPOLYLINETO16](#) (const RECTL *bounds, const POINT16 *points, INT n)
- [EMRPOLYLINETO16](#) (const RECTL *bounds, const POINT *points, INT n)
- [EMRPOLYLINETO16](#) (DATASTREAM &ds)
- [~EMRPOLYLINETO16](#) ()
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.34.1 Detailed Description

EMF PolylineTo16.

Draw a polygonal line curve to (what?)

4.34.2 Constructor & Destructor Documentation

4.34.2.1 EMRPOLYLINETO16() [1/3] `EMF::EMRPOLYLINETO16::EMRPOLYLINETO16 (const RECTL * bounds, const POINT16 * points, INT n) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINETO16\(\)](#).

Referenced by [EMRPOLYLINETO16\(\)](#).

4.34.2.2 EMRPOLYLINETO16() [2/3] `EMF::EMRPOLYLINETO16::EMRPOLYLINETO16 (`
`const RECTL * bounds,`
`const POINT * points,`
`INT n) [inline]`

Convenience constructor with POINTs.

Parameters

<i>bounds</i>	overall bounding box of polybezier curve.
<i>points</i>	array of polybezier vertices.
<i>n</i>	number of vertices in points.

References [EMRPOLYLINETO16\(\)](#).

4.34.2.3 EMRPOLYLINETO16() [3/3] `EMF::EMRPOLYLINETO16::EMRPOLYLINETO16 (`
`DATASTREAM & ds) [inline]`

Construct a PolylineTo record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.34.2.4 ~EMRPOLYLINETO16() `EMF::EMRPOLYLINETO16::~~EMRPOLYLINETO16 () [inline]`

Destructor frees a copy of the points it buffered.

4.34.3 Member Function Documentation

4.34.3.1 execute() `void EMF::EMRPOLYLINETO16::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.34.3.2 serialize() `bool EMF::EMRPOLYLINETO16::serialize (`
`DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.34.3.3 size() `int EMF::EMRPOLYLINETO16::size (`
`void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

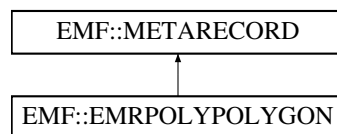
- libemf.h

4.35 EMF::EMRPOLYPOLYGON Class Reference

EMF Poly Polygon.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYPOLYGON:



Public Member Functions

- [EMRPOLYPOLYGON](#) (const RECTL *bounds, const POINT *points, const INT *counts, UINT polygons)
- [~EMRPOLYPOLYGON](#) ()
- [EMRPOLYPOLYGON](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.35.1 Detailed Description

EMF Poly Polygon.

Draw several filled polygons.

4.35.2 Constructor & Destructor Documentation

4.35.2.1 EMRPOLYPOLYGON() [1/2] `EMF::EMRPOLYPOLYGON::EMRPOLYPOLYGON (const RECTL * bounds, const POINT * points, const INT * counts, UINT polygons) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>counts</i>	array of number of vertices in each polygon.
<i>polygons</i>	number of polygons.

References [EMRPOLYPOLYGON\(\)](#).

Referenced by [EMRPOLYPOLYGON\(\)](#).

4.35.2.2 ~EMRPOLYPOLYGON() `EMF::EMRPOLYPOLYGON::~~EMRPOLYPOLYGON () [inline]`

Destructor frees a copy of the counts and points it buffered.

4.35.2.3 EMRPOLYPOLYGON() [2/2] `EMF::EMRPOLYPOLYGON::EMRPOLYPOLYGON (DATASTREAM & ds) [inline]`

Construct a Polygon record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.35.3 Member Function Documentation

4.35.3.1 execute() `void EMF::EMRPOLYPOLYGON::execute (`
`METAFILEDEVICECONTEXT * source,`
`HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.35.3.2 serialize() `bool EMF::EMRPOLYPOLYGON::serialize (`
`DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.35.3.3 size() `int EMF::EMRPOLYPOLYGON::size (`
`void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

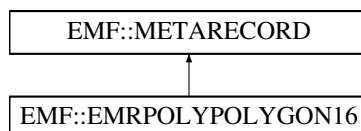
- libemf.h

4.36 EMF::EMRPOLYPOLYGON16 Class Reference

EMF Poly Polygon16.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRPOLYPOLYGON16:



Public Member Functions

- [EMRPOLYPOLYGON16](#) (const RECTL *bounds, const POINT *points, const INT *counts, UINT polygons)
- [EMRPOLYPOLYGON16](#) (const RECTL *bounds, const POINT16 *points, const INT *counts, UINT16 polygons)
- [~EMRPOLYPOLYGON16](#) ()
- [EMRPOLYPOLYGON16](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.36.1 Detailed Description

EMF Poly Polygon16.

Draw several filled polygons (with 16-bit points).

4.36.2 Constructor & Destructor Documentation

4.36.2.1 EMRPOLYPOLYGON16() [1/3] `EMF::EMRPOLYPOLYGON16::EMRPOLYPOLYGON16 (`
`const RECTL * bounds,`
`const POINT * points,`
`const INT * counts,`
`UINT polygons) [inline]`

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>counts</i>	array of number of vertices in each polygon.
<i>polygons</i>	number of polygons.

References [EMRPOLYPOLYGON16\(\)](#).

Referenced by [EMRPOLYPOLYGON16\(\)](#).

4.36.2.2 EMRPOLYPOLYGON16() [2/3] `EMF::EMRPOLYPOLYGON16::EMRPOLYPOLYGON16 (`
`const RECTL * bounds,`
`const POINT16 * points,`
`const INT * counts,`
`UINT16 polygons) [inline]`

Additional constructor which takes a POINT16 structure.

Parameters

<i>bounds</i>	overall bounding box of polygon.
<i>points</i>	array of polygon vertices.
<i>counts</i>	array of number of vertices in each polygon.
<i>polygons</i>	number of polygons.

References [EMRPOLYPOLYGON16\(\)](#).

4.36.2.3 `~EMRPOLYPOLYGON16()` `EMF::EMRPOLYPOLYGON16::~~EMRPOLYPOLYGON16 () [inline]`

Destructor frees a copy of the counts and points it buffered.

4.36.2.4 `EMRPOLYPOLYGON16()` `[3/3] EMF::EMRPOLYPOLYGON16::EMRPOLYPOLYGON16 (
 DATASTREAM & ds) [inline]`

Construct a Polygon record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.36.3 Member Function Documentation

4.36.3.1 `execute()` `void EMF::EMRPOLYPOLYGON16::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.36.3.2 `serialize()` `bool EMF::EMRPOLYPOLYGON16::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.36.3.3 size() `int EMF::EMRPOLYPOLYGON16::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

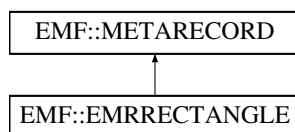
- libemf.h

4.37 EMF::EMRRECTANGLE Class Reference

EMF Rectangle.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRRECTANGLE:



Public Member Functions

- [EMRRECTANGLE](#) (INT left, INT top, INT right, INT bottom)
- [EMRRECTANGLE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.37.1 Detailed Description

EMF Rectangle.

Draw a rectangle.

4.37.2 Constructor & Destructor Documentation

4.37.2.1 EMRRECTANGLE() [1/2] `EMF::EMRRECTANGLE::EMRRECTANGLE (INT left, INT top, INT right, INT bottom) [inline]`

Parameters

<i>left</i>	x position of left side of rectangle.
<i>top</i>	y position of top side of rectangle.
<i>right</i>	x position of right edge of rectangle.
<i>bottom</i>	y position of bottom edge of rectangle.

References [EMRRECTANGLE\(\)](#).

Referenced by [EMRRECTANGLE\(\)](#).

4.37.2.2 EMRRECTANGLE() [2/2] `EMF::EMRRECTANGLE::EMRRECTANGLE (
 DATASTREAM & ds) [inline]`

Construct a Rectangle record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.37.3 Member Function Documentation

4.37.3.1 execute() `void EMF::EMRRECTANGLE::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.37.3.2 serialize() `bool EMF::EMRRECTANGLE::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

```
4.37.3.3 size() int EMF::EMRRECTANGLE::size (
    void ) const [inline], [virtual]
```

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

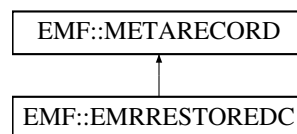
- libemf.h

4.38 EMF::EMRRESTOREDC Class Reference

EMF Restore DC.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRRESTOREDC:



Public Member Functions

- [EMRRESTOREDC](#) (INT n)
- [EMRRESTOREDC](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.38.1 Detailed Description

EMF Restore DC.

Use the stored device context in this context(?)

4.38.2 Constructor & Destructor Documentation

4.38.2.1 EMRSTOREDC() [1/2] `EMF::EMRSTOREDC::EMRSTOREDC (`
`INT n) [inline]`

Create a Restore DC record.

References [EMRSTOREDC\(\)](#).

Referenced by [EMRSTOREDC\(\)](#).

4.38.2.2 EMRSTOREDC() [2/2] `EMF::EMRSTOREDC::EMRSTOREDC (`
`DATASTREAM & ds) [inline]`

Construct an Restoredc record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.38.3 Member Function Documentation

4.38.3.1 execute() `void EMF::EMRRESTOREDC::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.38.3.2 serialize() `bool EMF::EMRRESTOREDC::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.38.3.3 size() `int EMF::EMRRESTOREDC::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

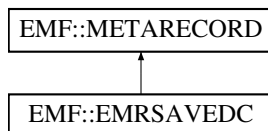
- libemf.h

4.39 EMF::EMRSAVE DC Class Reference

EMF Save DC.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSAVE DC:



Public Member Functions

- [EMRSAVE DC](#) (void)
- [EMRSAVE DC](#) ([DATASTREAM](#) &ds)
- bool [serialize](#) ([DATASTREAM](#) ds)
- int [size](#) (void) const
- void [execute](#) ([METAFILEDEVICECONTEXT](#) *source, HDC dc) const

4.39.1 Detailed Description

EMF Save DC.

Save the device context (i.e., push contents on a stack of some variety?)

4.39.2 Constructor & Destructor Documentation

4.39.2.1 EMRSAVE DC() [1/2] `EMF::EMRSAVE DC::EMRSAVE DC (void) [inline]`

Create a Save DC record.

References [EMRSAVE DC\(\)](#).

Referenced by [EMRSAVE DC\(\)](#).

4.39.2.2 EMRSAVE DC() [2/2] `EMF::EMRSAVE DC::EMRSAVE DC (DATASTREAM & ds) [inline]`

Construct an Savedc record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.39.3 Member Function Documentation

4.39.3.1 execute() void EMF::EMRSAVEDC::execute (
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.39.3.2 serialize() bool EMF::EMRSAVEDC::serialize (
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.39.3.3 size() int EMF::EMRSAVEDC::size (
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

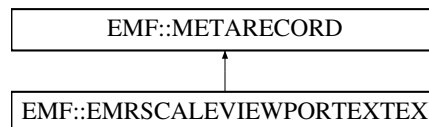
- libemf.h

4.40 EMF::EMRSCALEVIEWPORTEXTEX Class Reference

EMF Scale Viewport Extents (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSCALEVIEWPORTEXTEX:



Public Member Functions

- [EMRSCALEVIEWPORTEXTEX](#) (LONG x_num, LONG x_den, LONG y_num, LONG y_den)
- [EMRSCALEVIEWPORTEXTEX](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.40.1 Detailed Description

EMF Scale Viewport Extents (ex)

The viewport extent is the device coordinate (i.e. pixels) size of the viewport. Scale the viewport extents by the ratios of the given values. (OpenOffice accepts this, but not SETVIEWPORTEXT(?))

4.40.2 Constructor & Destructor Documentation

4.40.2.1 EMRSCALEVIEWPORTEXTEX() [1/2] `EMF::EMRSCALEVIEWPORTEXTEX::EMRSCALEVIEWPORTEXTEX (`
`LONG x_num,`
`LONG x_den,`
`LONG y_num,`
`LONG y_den) [inline]`

Parameters

<i>x_num</i>	numerator of x scale
<i>x_den</i>	denominator of x scale
<i>y_num</i>	numerator of y scale
<i>y_den</i>	denominator of y scale

References [EMRSCALEVIEWPORTEXTEX\(\)](#).

Referenced by [EMRSCALEVIEWPORTEXTEX\(\)](#).

4.40.2.2 EMRSCALEVIEWPORTEXTEx() [2/2] `EMF::EMRSCALEVIEWPORTEXTEx::EMRSCALEVIEWPORTEXTEx (DATASTREAM & ds) [inline]`

Construct a ScaleViewportExtEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.40.3 Member Function Documentation

4.40.3.1 execute() `void EMF::EMRSCALEVIEWPORTEXTEx::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.40.3.2 serialize() `bool EMF::EMRSCALEVIEWPORTEXTEx::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.40.3.3 size() `int EMF::EMRSCALEVIEWPORTEXTEx::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

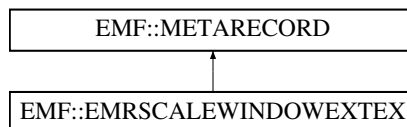
- libemf.h

4.41 EMF::EMRSCALEWINDOWEXTEx Class Reference

EMF Scale Window Extents (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSCALEWINDOWEXTEx:



Public Member Functions

- [EMRSCALEWINDOWEXTEx](#) (LONG x_num, LONG x_den, LONG y_num, LONG y_den)
- [EMRSCALEWINDOWEXTEx](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.41.1 Detailed Description

EMF Scale Window Extents (ex)

The window extent is the logical coordinate size of the window. Scale the window extents by the ratios of the given values.

4.41.2 Constructor & Destructor Documentation

4.41.2.1 EMRSCALEWINDOWEXTEx() [1/2] `EMF::EMRSCALEWINDOWEXTEx::EMRSCALEWINDOWEXTEx (`
 LONG x_num,
 LONG x_den,
 LONG y_num,
 LONG y_den) [inline]

Parameters

<i>x_num</i>	numerator of x scale
<i>x_den</i>	denominator of x scale
<i>y_num</i>	numerator of y scale
<i>y_den</i>	denominator of y scale

References [EMRSCALEWINDOWEXTEx\(\)](#).

Referenced by [EMRSCALEWINDOWEXTEx\(\)](#).

4.41.2.2 EMRSCALEWINDOWEXTEx() [2/2] `EMF::EMRSCALEWINDOWEXTEx::EMRSCALEWINDOWEXTEx (DATASTREAM & ds) [inline]`

Construct a ScaleWindowExtEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.41.3 Member Function Documentation

4.41.3.1 execute() `void EMF::EMRSCALEWINDOWEXTEx::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.41.3.2 serialize() `bool EMF::EMRSCALEWINDOWEXTEx::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.41.3.3 size() `int EMF::EMRSCALEWINDOWEXTEx::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

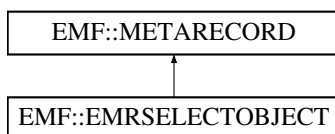
- libemf.h

4.42 EMF::EMRSELECTOBJECT Class Reference

EMF Select Object.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSELECTOBJECT:



Public Member Functions

- [EMRSELECTOBJECT](#) (HGDIOBJ object)
- [EMRSELECTOBJECT](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.42.1 Detailed Description

EMF Select Object.

Activate (make current) the given object, such as a pen, brush or font.

4.42.2 Constructor & Destructor Documentation

4.42.2.1 EMRSELECTOBJECT() [1/2] EMF::EMRSELECTOBJECT::EMRSELECTOBJECT (HGDIOBJ *object*) [inline]

Parameters

<i>object</i>	the object to make active.
---------------	----------------------------

References [EMRSELECTOBJECT\(\)](#).

Referenced by [EMRSELECTOBJECT\(\)](#).

4.42.2.2 EMRSELECTOBJECT() [2/2] EMF::EMRSELECTOBJECT::EMRSELECTOBJECT (DATASTREAM & *ds*) [inline]

Construct a SelectObject record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.42.3 Member Function Documentation

4.42.3.1 execute() `void EMF::EMRSELECTOBJECT::execute (
METAFILEDEVICECONTEXT * source,
HDC dc) const [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

References [EMF::METAFILEDEVICECONTEXT::emf_handles](#).

4.42.3.2 serialize() `bool EMF::EMRSELECTOBJECT::serialize (
DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.42.3.3 size() `int EMF::EMRSELECTOBJECT::size (
void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following files:

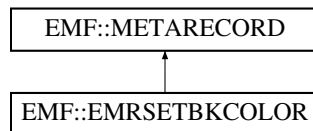
- libemf.h
- libemf.cpp

4.43 EMF::EMRSETBKCOLOR Class Reference

EMF Set Background Color.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETBKCOLOR:



Public Member Functions

- [EMRSETBKCOLOR](#) (COLORREF color)
- [EMRSETBKCOLOR](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.43.1 Detailed Description

EMF Set Background Color.

Sets the background color.

4.43.2 Constructor & Destructor Documentation

4.43.2.1 EMRSETBKCOLOR() [1/2] EMF::EMRSETBKCOLOR::EMRSETBKCOLOR (COLORREF *color*) [inline]

Parameters

<i>color</i>	background color
--------------	------------------

References [EMRSETBKCOLOR\(\)](#).

Referenced by [EMRSETBKCOLOR\(\)](#).

4.43.2.2 EMRSETBKCOLOR() [2/2] EMF::EMRSETBKCOLOR::EMRSETBKCOLOR (DATASTREAM & *ds*) [inline]

Construct a SetBkColor record from the input datastream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.43.3 Member Function Documentation

4.43.3.1 execute() void EMF::EMRSETBKCOLOR::execute (
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.43.3.2 serialize() bool EMF::EMRSETBKCOLOR::serialize (
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.43.3.3 size() int EMF::EMRSETBKCOLOR::size (
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

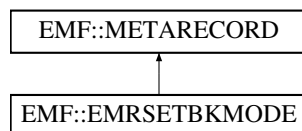
- libemf.h

4.44 EMF::EMRSETBKMODE Class Reference

EMF Set Background Mode.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETBKMODE:



Public Member Functions

- [EMRSETBKMODE](#) (DWORD mode)
- [EMRSETBKMODE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.44.1 Detailed Description

EMF Set Background Mode.

Set the background mode: transparent or opaque. Seems to be ignored by StarOffice. (Appears to work for text, though.)

4.44.2 Constructor & Destructor Documentation

4.44.2.1 EMRSETBKMODE() [1/2] EMF::EMRSETBKMODE::EMRSETBKMODE (
 DWORD *mode*) [inline]

Parameters

<i>mode</i>	background mode.
-------------	------------------

References [EMRSETBKMODE\(\)](#).

Referenced by [EMRSETBKMODE\(\)](#).

4.44.2.2 EMRSETBKMODE() [2/2] EMF::EMRSETBKMODE::EMRSETBKMODE (
 DATASTREAM & *ds*) [inline]

Construct a SetBkMode record from the input datastream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.44.3 Member Function Documentation

4.44.3.1 execute() void EMF::EMRSETBKMODE::execute (
[METAFILEDEVICECONTEXT](#) * *source*,
HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.44.3.2 serialize() bool EMF::EMRSETBKMODE::serialize (
[DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.44.3.3 size() int EMF::EMRSETBKMODE::size (
void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

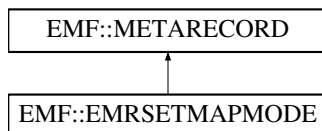
- libemf.h

4.45 EMF::EMRSETMAPMODE Class Reference

EMF Set Mapping Mode.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETMAPMODE:



Public Member Functions

- [EMRSETMAPMODE](#) (DWORD mode)
- [EMRSETMAPMODE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.45.1 Detailed Description

EMF Set Mapping Mode.

Set the mapping mode: HI (X style), LO (OpenGL style). Totally ignored by StarOffice as near as I can tell.

4.45.2 Constructor & Destructor Documentation

4.45.2.1 EMRSETMAPMODE() [1/2] EMF::EMRSETMAPMODE::EMRSETMAPMODE (
 DWORD mode) [inline]

Parameters

<i>mode</i>	window mapping mode
-------------	---------------------

References [EMRSETMAPMODE\(\)](#).

Referenced by [EMRSETMAPMODE\(\)](#).

4.45.2.2 EMRSETMAPMODE() [2/2] EMF::EMRSETMAPMODE::EMRSETMAPMODE (
 DATASTREAM & ds) [inline]

Construct a SetMapMode record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.45.3 Member Function Documentation

4.45.3.1 execute() void EMF::EMRSETMAPMODE::execute (
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.45.3.2 serialize() bool EMF::EMRSETMAPMODE::serialize (
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.45.3.3 size() int EMF::EMRSETMAPMODE::size (
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

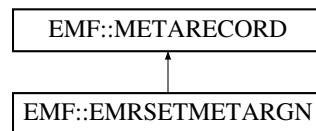
- libemf.h

4.46 EMF::EMRSETMETARGN Class Reference

EMF Set Meta Region.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETMETARGN:



Public Member Functions

- [EMRSETMETARGN](#) (void)
- [EMRSETMETARGN](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.46.1 Detailed Description

EMF Set Meta Region.

I really have no idea.

4.46.2 Constructor & Destructor Documentation

4.46.2.1 EMRSETMETARGN() [1/2] EMF::EMRSETMETARGN::EMRSETMETARGN (void) [inline]

Create a Set Meta Rgn record.

References [EMRSETMETARGN\(\)](#).

Referenced by [EMRSETMETARGN\(\)](#).

4.46.2.2 EMRSETMETARGN() [2/2] EMF::EMRSETMETARGN::EMRSETMETARGN (DATASTREAM & ds) [inline]

Construct an Setmetargn record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.46.3 Member Function Documentation

4.46.3.1 execute() void EMF::EMRSETMETARGN::execute (
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.46.3.2 serialize() bool EMF::EMRSETMETARGN::serialize (
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.46.3.3 size() int EMF::EMRSETMETARGN::size (
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

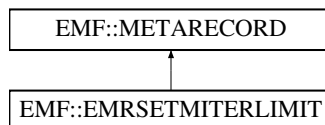
- libemf.h

4.47 EMF::EMRSETMITERLIMIT Class Reference

EMF SetMiterLimit.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETMITERLIMIT:



Public Member Functions

- [EMRSETMITERLIMIT](#) (FLOAT limit)
- [EMRSETMITERLIMIT](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.47.1 Detailed Description

EMF SetMiterLimit.

Sets the length limit for miter joins.

4.47.2 Constructor & Destructor Documentation

4.47.2.1 EMRSETMITERLIMIT() [1/2] EMF::EMRSETMITERLIMIT::EMRSETMITERLIMIT (
 FLOAT *limit*) [inline]

Parameters

<i>limit</i>	miter length limit.
--------------	---------------------

References [EMRSETMITERLIMIT\(\)](#).

Referenced by [EMRSETMITERLIMIT\(\)](#).

4.47.2.2 EMRSETMITERLIMIT() [2/2] EMF::EMRSETMITERLIMIT::EMRSETMITERLIMIT (
 DATASTREAM & *ds*) [inline]

Construct a SetMiterLimit record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.47.3 Member Function Documentation

4.47.3.1 execute() void EMF::EMRSETMITERLIMIT::execute (
 [METAFILEDEVICECONTEXT](#) * *source*,
 HDC *dc*) const [inline], [virtual]

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.47.3.2 serialize() bool EMF::EMRSETMITERLIMIT::serialize (
 [DATASTREAM](#) *ds*) [inline], [virtual]

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.47.3.3 size() int EMF::EMRSETMITERLIMIT::size (
 void) const [inline], [virtual]

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

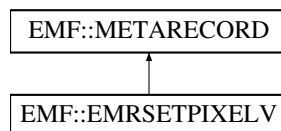
- libemf.h

4.48 EMF::EMRSETPIXELV Class Reference

EMF Set Pixel.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETPIXELV:



Public Member Functions

- [EMRSETPIXELV](#) (INT x, INT y, COLORREF color)
- [EMRSETPIXELV](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.48.1 Detailed Description

EMF Set Pixel.

Set the given pixel to the given color.

4.48.2 Constructor & Destructor Documentation

4.48.2.1 EMRSETPIXELV() [1/2] `EMF::EMRSETPIXELV::EMRSETPIXELV (`
`INT x,`
`INT y,`
`COLORREF color) [inline]`

Parameters

<i>x</i>	x position at which to draw pixel.
<i>y</i>	y position at which to draw pixel.
<i>color</i>	color of pixel.

References [EMRSETPIXELV\(\)](#).

Referenced by [EMRSETPIXELV\(\)](#).

4.48.2.2 EMRSETPIXELV() [2/2] `EMF::EMRSETPIXELV::EMRSETPIXELV (
 DATASTREAM & ds) [inline]`

Construct a SetPixelV record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.48.3 Member Function Documentation

4.48.3.1 execute() `void EMF::EMRSETPIXELV::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.48.3.2 serialize() `bool EMF::EMRSETPIXELV::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.48.3.3 size() `int EMF::EMRSETPIXELV::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

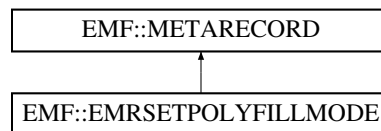
- libemf.h

4.49 EMF::EMRSETPOLYFILLMODE Class Reference

EMF Set the Polygon Fill Mode.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETPOLYFILLMODE:



Public Member Functions

- [EMRSETPOLYFILLMODE](#) (DWORD mode)
- [EMRSETPOLYFILLMODE](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.49.1 Detailed Description

EMF Set the Polygon Fill Mode.

Set the polygon fill mode: ALTERNATE or WINDING

4.49.2 Constructor & Destructor Documentation

4.49.2.1 EMRSETPOLYFILLMODE() [1/2] `EMF::EMRSETPOLYFILLMODE::EMRSETPOLYFILLMODE (
 DWORD mode) [inline]`

Parameters

<i>mode</i>	background mode.
-------------	------------------

References [EMRSETPOLYFILLMODE\(\)](#).

Referenced by [EMRSETPOLYFILLMODE\(\)](#).

4.49.2.2 EMRSETPOLYFILLMODE() [2/2] `EMF::EMRSETPOLYFILLMODE::EMRSETPOLYFILLMODE (
 DATASTREAM & ds) [inline]`

Construct a SetPolyFillMode record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.49.3 Member Function Documentation

4.49.3.1 execute() `void EMF::EMRSETPOLYFILLMODE::execute (
METAFILEDEVICECONTEXT * source,
HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.49.3.2 serialize() `bool EMF::EMRSETPOLYFILLMODE::serialize (
DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.49.3.3 size() `int EMF::EMRSETPOLYFILLMODE::size (
void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

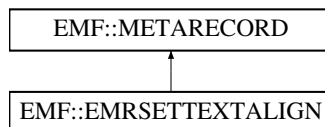
- libemf.h

4.50 EMF::EMRSETTEXTALIGN Class Reference

EMF Set Text Alignment.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETTEXTALIGN:



Public Member Functions

- [EMRSETTEXTALIGN](#) (UINT mode)
- [EMRSETTEXTALIGN](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.50.1 Detailed Description

EMF Set Text Alignment.

Determines the justification of the text with respect to its position.

4.50.2 Constructor & Destructor Documentation

4.50.2.1 EMRSETTEXTALIGN() [1/2] `EMF::EMRSETTEXTALIGN::EMRSETTEXTALIGN (
UINT mode) [inline]`

Parameters

<i>mode</i>	text alignment mode.
-------------	----------------------

References [EMRSETTEXTALIGN\(\)](#).

Referenced by [EMRSETTEXTALIGN\(\)](#).

4.50.2.2 EMRSETTEXTALIGN() [2/2] `EMF::EMRSETTEXTALIGN::EMRSETTEXTALIGN (
DATASTREAM & ds) [inline]`

Construct a SetTextAlign record from the input datastream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.50.3 Member Function Documentation

4.50.3.1 execute() `void EMF::EMRSETTEXTALIGN::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.50.3.2 serialize() `bool EMF::EMRSETTEXTALIGN::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.50.3.3 size() `int EMF::EMRSETTEXTALIGN::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

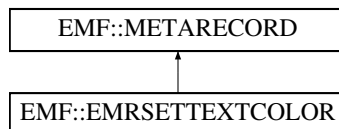
- libemf.h

4.51 EMF::EMRSETTEXTCOLOR Class Reference

EMF Set Text Color.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETTEXTCOLOR:



Public Member Functions

- [EMRSETTEXTCOLOR](#) (COLORREF color)
- [EMRSETTEXTCOLOR](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.51.1 Detailed Description

EMF Set Text Color.

Sets the foreground color of text.

4.51.2 Constructor & Destructor Documentation

4.51.2.1 EMRSETTEXTCOLOR() [1/2] EMF::EMRSETTEXTCOLOR::EMRSETTEXTCOLOR (COLORREF color) [inline]

Parameters

<i>color</i>	text foreground color
--------------	-----------------------

References [EMRSETTEXTCOLOR\(\)](#).

Referenced by [EMRSETTEXTCOLOR\(\)](#).

4.51.2.2 EMRSETTEXTCOLOR() [2/2] EMF::EMRSETTEXTCOLOR::EMRSETTEXTCOLOR (DATASTREAM & ds) [inline]

Construct a SetTextColor record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.51.3 Member Function Documentation

4.51.3.1 execute() `void EMF::EMRSETTEXTCOLOR::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.51.3.2 serialize() `bool EMF::EMRSETTEXTCOLOR::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.51.3.3 size() `int EMF::EMRSETTEXTCOLOR::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

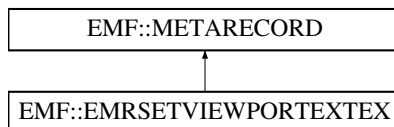
- libemf.h

4.52 EMF::EMRSETVIEWPORTEXX Class Reference

EMF Set Viewport Extents (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETVIEWPORTEXX:



Public Member Functions

- [EMRSETVIEWPORTEXX](#) (INT cx, INT cy)
- [EMRSETVIEWPORTEXX](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.52.1 Detailed Description

EMF Set Viewport Extents (ex)

The viewport extent is the device coordinate (i.e. pixels) size of the viewport. Since W32 doesn't do any clipping, the purpose of this is not clear.

4.52.2 Constructor & Destructor Documentation

4.52.2.1 EMRSETVIEWPORTEXX() [1/2] `EMF::EMRSETVIEWPORTEXX::EMRSETVIEWPORTEXX (`
 INT cx,
 INT cy) [inline]

Parameters

cx	width of viewport in device coordinates
cy	height of viewport in device coordinates

References [EMRSETVIEWPORTEXX\(\)](#).

Referenced by [EMRSETVIEWPORTEXX\(\)](#).

4.52.2.2 EMRSETVIEWPORTEXTEx() [2/2] `EMF::EMRSETVIEWPORTEXTEx::EMRSETVIEWPORTEXTEx (DATASTREAM & ds) [inline]`

Construct a SetViewportExtEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.52.3 Member Function Documentation

4.52.3.1 execute() `void EMF::EMRSETVIEWPORTEXTEx::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.52.3.2 serialize() `bool EMF::EMRSETVIEWPORTEXTEx::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.52.3.3 size() `int EMF::EMRSETVIEWPORTEXTEx::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

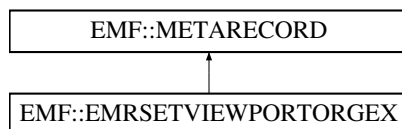
- libemf.h

4.53 EMF::EMRSETVIEWPORTORGEX Class Reference

EMF Set Viewport Origin (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETVIEWPORTORGEX:



Public Member Functions

- [EMRSETVIEWPORTORGEX](#) (INT x, INT y)
- [EMRSETVIEWPORTORGEX](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.53.1 Detailed Description

EMF Set Viewport Origin (ex)

The viewport origin is a point in device coordinates (i.e., pixels) where the viewport starts. (For example, if you want to put several different views on the same page, you might use different viewports.)

4.53.2 Constructor & Destructor Documentation

4.53.2.1 EMRSETVIEWPORTORGEX() [1/2] `EMF::EMRSETVIEWPORTORGEX::EMRSETVIEWPORTORGEX (`
`INT x,`
`INT y) [inline]`

Parameters

<i>x</i>	x position of the viewport in device coordinates
<i>y</i>	y position of the viewport in device coordinates

References [EMRSETVIEWPORTORGEX\(\)](#).

Referenced by [EMRSETVIEWPORTORGEX\(\)](#).

4.53.2.2 EMRSETVIEWPORTORGEX() [2/2] `EMF::EMRSETVIEWPORTORGEX::EMRSETVIEWPORTORGEX (DATASTREAM & ds) [inline]`

Construct a SetVieportOrgEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.53.3 Member Function Documentation

4.53.3.1 execute() `void EMF::EMRSETVIEWPORTORGEX::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.53.3.2 serialize() `bool EMF::EMRSETVIEWPORTORGEX::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.53.3.3 size() `int EMF::EMRSETVIEWPORTORGEX::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

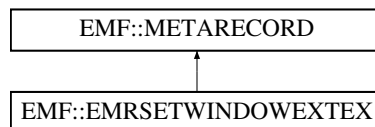
- libemf.h

4.54 EMF::EMRSETWINDOWEXTEx Class Reference

EMF Set Window Extent (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETWINDOWEXTEx:



Public Member Functions

- [EMRSETWINDOWEXTEx](#) (INT cx, INT cy)
- [EMRSETWINDOWEXTEx](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.54.1 Detailed Description

EMF Set Window Extent (ex)

The window extents define the scale of the logical coordinates. For example, if your XY plot is from [-10,-10] to [10,10], then the window extents are [20,20].

4.54.2 Constructor & Destructor Documentation

4.54.2.1 EMRSETWINDOWEXTEx() [1/2] `EMF::EMRSETWINDOWEXTEx::EMRSETWINDOWEXTEx (`
 INT cx,
 INT cy) `[inline]`

Parameters

<i>cx</i>	width of window in logical coordinates.
<i>cy</i>	height of window in logical coordinates.

References [EMRSETWINDOWEXTEx\(\)](#).

Referenced by [EMRSETWINDOWEXTEx\(\)](#).

4.54.2.2 EMRSETWINDOWEXTEx() [2/2] `EMF::EMRSETWINDOWEXTEx::EMRSETWINDOWEXTEx (DATASTREAM & ds) [inline]`

Construct a SetWindowExtEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.54.3 Member Function Documentation

4.54.3.1 execute() `void EMF::EMRSETWINDOWEXTEx::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.54.3.2 serialize() `bool EMF::EMRSETWINDOWEXTEx::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.54.3.3 size() `int EMF::EMRSETWINDOWEXTEx::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

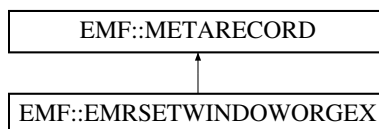
- libemf.h

4.55 EMF::EMRSETWINDOWORGEX Class Reference

EMF Set Window Origin (ex)

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETWINDOWORGEX:



Public Member Functions

- [EMRSETWINDOWORGEX](#) (INT x, INT y)
- [EMRSETWINDOWORGEX](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.55.1 Detailed Description

EMF Set Window Origin (ex)

The window origin specifies the logical (i.e., real) coordinates of the upper, left corner of the viewport. (For example, if you want your XY plot's axis origin to be in the middle of the viewport, you'd set the window origin to something like [-1,-1].)

4.55.2 Constructor & Destructor Documentation

4.55.2.1 EMRSETWINDOWORGEX() [1/2] `EMF::EMRSETWINDOWORGEX::EMRSETWINDOWORGEX (`
 INT x,
 INT y) [inline]

Parameters

x	x coordinate of window origin in logical coordinates
y	y coordinate of window origin in logical coordinates

References [EMRSETWINDOWORGEX\(\)](#).

Referenced by [EMRSETWINDOWORGEX\(\)](#).

4.55.2.2 EMRSETWINDOWORGEX() [2/2] `EMF::EMRSETWINDOWORGEX::EMRSETWINDOWORGEX (DATASTREAM & ds) [inline]`

Construct a SetWindowOrgEx record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.55.3 Member Function Documentation

4.55.3.1 execute() `void EMF::EMRSETWINDOWORGEX::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.55.3.2 serialize() `bool EMF::EMRSETWINDOWORGEX::serialize (DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.55.3.3 size() `int EMF::EMRSETWINDOWORGEX::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

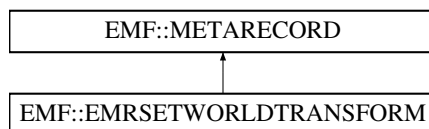
- libemf.h

4.56 EMF::EMRSETWORLDTRANSFORM Class Reference

EMF Set World Transform.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSETWORLDTRANSFORM:



Public Member Functions

- [EMRSETWORLDTRANSFORM](#) (const XFORM *transform)
- [EMRSETWORLDTRANSFORM](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.56.1 Detailed Description

EMF Set World Transform.

Enhanced metafiles have a Coordinate Transformation which allows the contents to be rotated and transformed. Does not appear to work properly in StarOffice (but it's also possible I don't understand how it's supposed to work either).

4.56.2 Constructor & Destructor Documentation

4.56.2.1 EMRSETWORLDTRANSFORM() [1/2] `EMF::EMRSETWORLDTRANSFORM::EMRSETWORLDTRANSFORM (const XFORM * transform) [inline]`

Parameters

<i>transform</i>	the new transformation
------------------	------------------------

References [EMRSETWORLDTRANSFORM\(\)](#).

Referenced by [EMRSETWORLDTRANSFORM\(\)](#).

4.56.2.2 EMRSETWORLDTRANSFORM() [2/2] `EMF::EMRSETWORLDTRANSFORM::EMRSETWORLDTRANSFORM (DATASTREAM & ds) [inline]`

Construct a SetWorldTransform record from the input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.56.3 Member Function Documentation

4.56.3.1 execute() `void EMF::EMRSETWORLDTRANSFORM::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.56.3.2 serialize() `bool EMF::EMRSETWORLDTRANSFORM::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.56.3.3 size() `int EMF::EMRSETWORLDTRANSFORM::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

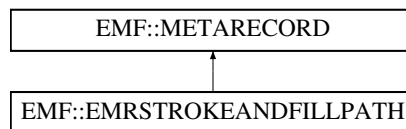
- libemf.h

4.57 EMF::EMRSTROKEANDFILLPATH Class Reference

EMF Stroke and Fill path.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSTROKEANDFILLPATH:



Public Member Functions

- [EMRSTROKEANDFILLPATH](#) (const RECTL *bounds)
- [EMRSTROKEANDFILLPATH](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.57.1 Detailed Description

EMF Stroke and Fill path.

Stroke and Fill the path.

4.57.2 Constructor & Destructor Documentation

4.57.2.1 EMRSTROKEANDFILLPATH() [1/2] EMF::EMRSTROKEANDFILLPATH::EMRSTROKEANDFILLPATH (const RECTL * *bounds*) [inline]

Parameters

<i>bounds</i>	overall bounding box of polygon.
---------------	----------------------------------

References [EMRSTROKEANDFILLPATH\(\)](#).

Referenced by [EMRSTROKEANDFILLPATH\(\)](#).

4.57.2.2 EMRSTROKEANDFILLPATH() [2/2] EMF::EMRSTROKEANDFILLPATH::EMRSTROKEANDFILLPATH (DATASTREAM & *ds*) [inline]

Create a StrokeandfillPath record from input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.57.3 Member Function Documentation

4.57.3.1 execute() `void EMF::EMRSTROKEANDFILLPATH::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.57.3.2 serialize() `bool EMF::EMRSTROKEANDFILLPATH::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.57.3.3 size() `int EMF::EMRSTROKEANDFILLPATH::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

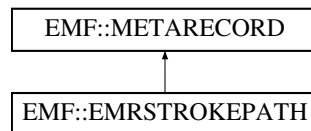
- libemf.h

4.58 EMF::EMRSTROKEPATH Class Reference

EMF Stroke path.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EMRSTROKEPATH:



Public Member Functions

- [EMRSTROKEPATH](#) (const RECTL *bounds)
- [EMRSTROKEPATH](#) (DATASTREAM &ds)
- bool [serialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.58.1 Detailed Description

EMF Stroke path.

Stroke the path.

4.58.2 Constructor & Destructor Documentation

4.58.2.1 EMRSTROKEPATH() [1/2] EMF::EMRSTROKEPATH::EMRSTROKEPATH (
 const RECTL * *bounds*) [inline]

Parameters

<i>bounds</i>	overall bounding box of polygon.
---------------	----------------------------------

References [EMRSTROKEPATH\(\)](#).

Referenced by [EMRSTROKEPATH\(\)](#).

4.58.2.2 EMRSTROKEPATH() [2/2] EMF::EMRSTROKEPATH::EMRSTROKEPATH (
 DATASTREAM & *ds*) [inline]

Create a StrokePath record from input stream.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

4.58.3 Member Function Documentation

4.58.3.1 execute() `void EMF::EMRSTROKEPATH::execute (
 METAFILEDEVICECONTEXT * source,
 HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.58.3.2 serialize() `bool EMF::EMRSTROKEPATH::serialize (
 DATASTREAM ds) [inline], [virtual]`

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.58.3.3 size() `int EMF::EMRSTROKEPATH::size (
 void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

The documentation for this class was generated from the following file:

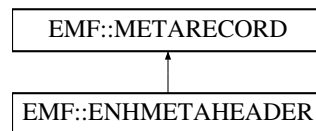
- libemf.h

4.59 EMF::ENHMETAHEADER Class Reference

Enhanced Metafile Header Record.

```
#include <libemf.h>
```

Inheritance diagram for EMF::ENHMETAHEADER:



Public Member Functions

- [ENHMETAHEADER](#) (LPCWSTR description=0)
- [~ENHMETAHEADER](#) ()
- bool [serialize](#) (DATASTREAM ds)
- bool [unserialize](#) (DATASTREAM ds)
- int [size](#) (void) const
- void [execute](#) (METAFILEDEVICECONTEXT *source, HDC dc) const

4.59.1 Detailed Description

Enhanced Metafile Header Record.

The [ENHMETAHEADER](#) serves two purposes in this library: it keeps track of the size of the metafile (in physical dimensions) and number of records and handles that are ultimately to be written to the disk file. It is also a real record that must be written out.

4.59.2 Constructor & Destructor Documentation

4.59.2.1 ENHMETAHEADER() `EMF::ENHMETAHEADER::ENHMETAHEADER (LPCWSTR description = 0) [inline]`

Parameters

<i>description</i>	an optional description argument is a UNICODE-like string with the following format: "some text\0some more text\0\0". The W32 interface defines UNICODE characters to be two-byte (unsigned short strings). The constructor makes a copy of the argument.
--------------------	---

References [ENHMETAHEADER\(\)](#).

Referenced by [ENHMETAHEADER\(\)](#), and [unserialize\(\)](#).

4.59.2.2 `~ENHMETAHEADER()` `EMF::ENHMETAHEADER::~~ENHMETAHEADER () [inline]`

Destructor deletes memory allocated for description.

4.59.3 Member Function Documentation

4.59.3.1 `execute()` `void EMF::ENHMETAHEADER::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [inline], [virtual]`

Execute this record in the context of the given device context.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	device context for execute.

Implements [EMF::METARECORD](#).

4.59.3.2 `serialize()` `bool EMF::ENHMETAHEADER::serialize (DATASTREAM ds) [inline], [virtual]`

Serializing the header is an example of an extended record.

Parameters

<i>ds</i>	Metafile datastream.
-----------	----------------------

Implements [EMF::METARECORD](#).

4.59.3.3 `size()` `int EMF::ENHMETAHEADER::size (void) const [inline], [virtual]`

Internally computed size of this record.

Implements [EMF::METARECORD](#).

4.59.3.4 `unserialize()` `bool EMF::ENHMETAHEADER::unserialize (DATASTREAM ds) [inline]`

Read a header record from the datastream.

References [ENHMETAHEADER\(\)](#).

The documentation for this class was generated from the following file:

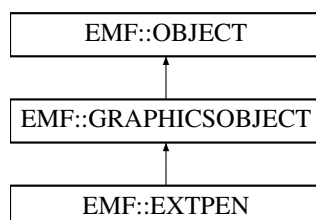
- libemf.h

4.60 EMF::EXTPEN Class Reference

Extended Graphics Pen.

```
#include <libemf.h>
```

Inheritance diagram for EMF::EXTPEN:



Public Member Functions

- [EXTPEN](#) (const EXTLOGPEN *lpen)
- OBJECTTYPE [getType](#) (void) const
- METARECORD * [newEMR](#) (HDC dc, HGDIOBJ emf_handle)

Additional Inherited Members

4.60.1 Detailed Description

Extended Graphics Pen.

Pens are used for drawing lines, arc, rectangles, etc.

4.60.2 Constructor & Destructor Documentation

4.60.2.1 EXT PEN() `EMF::EXTPEN::EXTPEN (const EXTLOGPEN * lpen) [inline]`

Parameters

<i>lpen</i>	the (logical?) pen definition.
-------------	--------------------------------

4.60.3 Member Function Documentation

4.60.3.1 `getType()` `OBJECTTYPE EMF::EXTPEN::getType (`
`void) const [inline], [virtual]`

Return the type of this object (could probably do better with `RTTI()`).

Implements [EMF::OBJECT](#).

4.60.3.2 `newEMR()` `METARECORD * EMF::EXTPEN::newEMR (`
`HDC dc,`
`HGDIOBJ emf_handle) [inline], [virtual]`

Return a new metarecord for this object. And record its selection into the given device context.

Parameters

<i>dc</i>	handle of device context into which this object is being selected.
<i>emf_handle</i>	the EMF handle associated with the PEN .

Implements [EMF::GRAPHICSOBJECT](#).

References [EMF::GRAPHICSOBJECT::contexts](#).

The documentation for this class was generated from the following file:

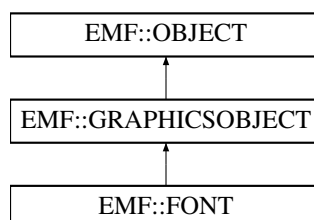
- `libemf.h`

4.61 EMF::FONT Class Reference

Graphics Font.

```
#include <libemf.h>
```

Inheritance diagram for `EMF::FONT`:



Public Member Functions

- [FONT](#) (`const LOGFONTW *lfont`)
- `OBJECTTYPE` [getType](#) (`void`) `const`
- [METARECORD](#) * [newEMR](#) (`HDC dc`, `HGDIOBJ emf_handle`)

Additional Inherited Members

4.61.1 Detailed Description

Graphics Font.

Fonts are used for drawing text (obviously).

4.61.2 Constructor & Destructor Documentation

4.61.2.1 FONT() `EMF::FONT::FONT (const LOGFONTW * lfont) [inline]`

Parameters

<i>lfont</i>	the (logical?) font definition.
--------------	---------------------------------

4.61.3 Member Function Documentation

4.61.3.1 getType() `OBJECTTYPE EMF::FONT::getType (void) const [inline], [virtual]`

Return the type of this object (could probably do better with RTTI()).

Implements [EMF::OBJECT](#).

4.61.3.2 newEMR() `METARECORD * EMF::FONT::newEMR (HDC dc, HGDIOBJ emf_handle) [inline], [virtual]`

Return a new metarecord for this object. And record its selection into the given device context.

Parameters

<i>dc</i>	handle of device context into which this object is being selected.
<i>emf_handle</i>	the EMF handle associated with the FONT .

Implements [EMF::GRAPHICSOBJECT](#).

References [EMF::GRAPHICSOBJECT::contexts](#).

The documentation for this class was generated from the following file:

- libemf.h

4.62 EMF::GLOBALOBJECTS Class Reference

```
#include <libemf.h>
```

Public Member Functions

- HGDIOBJ [add](#) (OBJECT *object)
- OBJECT * [find](#) (const HGDIOBJ handle)
- void [remove](#) (const OBJECT *object)
- auto [begin](#) (void) const
- auto [end](#) (void) const
- METARECORDCTOR [newRecord](#) (DWORD iType) const

Static Public Member Functions

- static EMF::METARECORD * [new_eof](#) (DATASTREAM &ds)
Create a new EMREOF record.
- static EMF::METARECORD * [new_setviewportorgex](#) (DATASTREAM &ds)
Create a new EMRSETVIEWPORTORGEX record.
- static EMF::METARECORD * [new_setwindoworgex](#) (DATASTREAM &ds)
Create a new EMRSETWINDOWORGEX record.
- static EMF::METARECORD * [new_setviewporttex](#) (DATASTREAM &ds)
Create a new EMRSETVIEWPORTEXTEX record.
- static EMF::METARECORD * [new_setwindowextex](#) (DATASTREAM &ds)
Create a new EMRSETWINDOWEXTEX record.
- static EMF::METARECORD * [new_scaleviewporttex](#) (DATASTREAM &ds)
Create a new SCALEVIEWPORTEXTEX record.
- static EMF::METARECORD * [new_scalewindowextex](#) (DATASTREAM &ds)
Create a new SCALEWINDOWEXTEX record.
- static EMF::METARECORD * [new_modifyworldtransform](#) (DATASTREAM &ds)
Create a new MODIFYWORLDTRANSFORM record.
- static EMF::METARECORD * [new_setworldtransform](#) (DATASTREAM &ds)
Create a new SETWORLDTRANSFORM record.
- static EMF::METARECORD * [new_settextalign](#) (DATASTREAM &ds)
Create a new SETTEXTALIGN record.
- static EMF::METARECORD * [new_settextcolor](#) (DATASTREAM &ds)
Create a new SETTEXTCOLOR record.
- static EMF::METARECORD * [new_setbkcolor](#) (DATASTREAM &ds)
Create a new SETBKCOLOR record.
- static EMF::METARECORD * [new_setbkmode](#) (DATASTREAM &ds)
Create a new SETBKMODE record.
- static EMF::METARECORD * [new_setpolyfillmode](#) (DATASTREAM &ds)
Create a new SETPOLYFILLMODE record.
- static EMF::METARECORD * [new_setmapmode](#) (DATASTREAM &ds)
Create a new SETMAPMODE record.

- static [EMF::METARECORD](#) * [new_selectobject](#) ([DATASTREAM](#) &ds)
Create a new SELECTOBJECT record.
- static [EMF::METARECORD](#) * [new_deleteobject](#) ([DATASTREAM](#) &ds)
Create a new DELETEOBJECT record.
- static [EMF::METARECORD](#) * [new_movetoex](#) ([DATASTREAM](#) &ds)
Create a new MOVETOEX record.
- static [EMF::METARECORD](#) * [new_lineto](#) ([DATASTREAM](#) &ds)
Create a new LINETO record.
- static [EMF::METARECORD](#) * [new_arc](#) ([DATASTREAM](#) &ds)
Create a new ARC record.
- static [EMF::METARECORD](#) * [new_arcto](#) ([DATASTREAM](#) &ds)
Create a new ARCTO record.
- static [EMF::METARECORD](#) * [new_rectangle](#) ([DATASTREAM](#) &ds)
Create a new RECTANGLE record.
- static [EMF::METARECORD](#) * [new_ellipse](#) ([DATASTREAM](#) &ds)
Create a new ELLIPSE record.
- static [EMF::METARECORD](#) * [new_polyline](#) ([DATASTREAM](#) &ds)
Create a new POLYLINE record.
- static [EMF::METARECORD](#) * [new_polyline16](#) ([DATASTREAM](#) &ds)
Create a new POLYLINE16 record.
- static [EMF::METARECORD](#) * [new_polygon](#) ([DATASTREAM](#) &ds)
Create a new POLYGON record.
- static [EMF::METARECORD](#) * [new_polygon16](#) ([DATASTREAM](#) &ds)
Create a new POLYGON16 record.
- static [EMF::METARECORD](#) * [new_polypolygon](#) ([DATASTREAM](#) &ds)
Create a new POLYPOLYGON record.
- static [EMF::METARECORD](#) * [new_polypolygon16](#) ([DATASTREAM](#) &ds)
Create a new POLYPOLYGON16 record.
- static [EMF::METARECORD](#) * [new_polybezier](#) ([DATASTREAM](#) &ds)
Create a new POLYBEZIER record.
- static [EMF::METARECORD](#) * [new_polybezier16](#) ([DATASTREAM](#) &ds)
Create a new POLYBEZIER16 record.
- static [EMF::METARECORD](#) * [new_polybezierto](#) ([DATASTREAM](#) &ds)
Create a new POLYBEZIERTO record.
- static [EMF::METARECORD](#) * [new_polybezierto16](#) ([DATASTREAM](#) &ds)
Create a new POLYBEZIERTO16 record.
- static [EMF::METARECORD](#) * [new_polylineto](#) ([DATASTREAM](#) &ds)
Create a new POLYLINETO record.
- static [EMF::METARECORD](#) * [new_polylineto16](#) ([DATASTREAM](#) &ds)
Create a new POLYLINETO16 record.
- static [EMF::METARECORD](#) * [new_exttextouta](#) ([DATASTREAM](#) &ds)
Create a new EXTTEXTOUTA record.
- static [EMF::METARECORD](#) * [new_exttextoutw](#) ([DATASTREAM](#) &ds)
Create a new EXTTEXTOUTW record.
- static [EMF::METARECORD](#) * [new_setpixelv](#) ([DATASTREAM](#) &ds)
Create a new SETPIXELV record.
- static [EMF::METARECORD](#) * [new_createpen](#) ([DATASTREAM](#) &ds)
Create a new CREATEPEN record.
- static [EMF::METARECORD](#) * [new_extcreatepen](#) ([DATASTREAM](#) &ds)
Create a new EXTCREATEPEN record.
- static [EMF::METARECORD](#) * [new_createbrushindirect](#) ([DATASTREAM](#) &ds)

- Create a new *CREATEBRUSHINDIRECT* record.
 • static `EMF::METARECORD * new_extcreatefontindirectw (DATASTREAM &ds)`
 Create a new *EXTCREATEFONTINDIRECTW* record.
- static `EMF::METARECORD * new_fillpath (DATASTREAM &ds)`
 Create a new *FILLPATH* record.
- static `EMF::METARECORD * new_strokepath (DATASTREAM &ds)`
 Create a new *STROKEPATH* record.
- static `EMF::METARECORD * new_strokeandfillpath (DATASTREAM &ds)`
 Create a new *STROKEANDFILLPATH* record.
- static `EMF::METARECORD * new_beginpath (DATASTREAM &ds)`
 Create a new *BEGINPATH* record.
- static `EMF::METARECORD * new_endpath (DATASTREAM &ds)`
 Create a new *ENDPATH* record.
- static `EMF::METARECORD * new_closefigure (DATASTREAM &ds)`
 Create a new *CLOSEFIGURE* record.
- static `EMF::METARECORD * new_savedc (DATASTREAM &ds)`
 Create a new *SAVEDC* record.
- static `EMF::METARECORD * new_restoredc (DATASTREAM &ds)`
 Create a new *RESTOREDC* record.
- static `EMF::METARECORD * new_setmetargn (DATASTREAM &ds)`
 Create a new *SETMETARGN* record.
- static `EMF::METARECORD * new_setmiterlimit (DATASTREAM &ds)`
 Create a new *SETMITERLIMIT* record.

4.62.1 Detailed Description

Stores all the objects in a single database within a process.

4.62.2 Member Function Documentation

4.62.2.1 add() `HGDIOBJ EMF::GLOBALOBJECTS::add (`
 `OBJECT * object)`

Add an object to the global vector. The object's handle is simply its index in the global object vector, which is computed by the very interesting "difference between two iterators" method.

Parameters

<i>object</i>	pointer to a real instance of an object, not its handle.
---------------	--

4.62.2.2 begin() `auto EMF::GLOBALOBJECTS::begin (`
 `void) const [inline]`

Returns

an iterator pointing to the first global object.

4.62.2.3 end() `auto EMF::GLOBALOBJECTS::end (`
`void) const [inline]`

Returns

an iterator pointing to (one past) the final global object.

4.62.2.4 find() `OBJECT * EMF::GLOBALOBJECTS::find (`
`const HGDIOBJ handle)`

Look up a object by handle in the global object vector. Note: Stock objects (like a gray brush or the black pen) have their high order bit set, so this has to be masked out when using their handles.

Parameters

<i>handle</i>	the object's handle.
---------------	----------------------

Returns

pointer to object.

4.62.2.5 newRecord() `METARECORDCTOR EMF::GLOBALOBJECTS::newRecord (`
`DWORD iType) const`

See if we have a constructor for a record of the given type.

Parameters

<i>iType</i>	metarecord type.
--------------	------------------

Returns

pointer to "virtual" constructor.

4.62.2.6 remove() `void EMF::GLOBALOBJECTS::remove (`
`const OBJECT * object)`

A call to the metafile function `DeleteObject()` allows a particular object's handle to be reused, so some care has to be taken to erase it.

Parameters

<i>object</i>	pointer to object to delete.
---------------	------------------------------

The documentation for this class was generated from the following files:

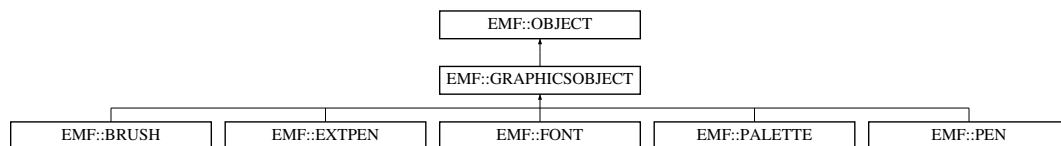
- libemf.h
- libemf.cpp

4.63 EMF::GRAPHICSOBJECT Class Reference

A global graphics object.

```
#include <libemf.h>
```

Inheritance diagram for EMF::GRAPHICSOBJECT:



Public Member Functions

- virtual `~GRAPHICSOBJECT()`
GRAPHICSOBJECTs has a virtual destructor.
- virtual `METARECORD * newEMR (HDC dc, HGDIOBJ handle)=0`

Data Fields

- `std::map< HDC, HGDIOBJ > contexts`

4.63.1 Detailed Description

A global graphics object.

Graphics objects have some additional properties: When an object is Select'ed into a device context, the handle for that context is added to the list of context's in which this object is used.

4.63.2 Member Function Documentation

4.63.2.1 newEMR() `virtual METARECORD * EMF::GRAPHICSOBJECT::newEMR (HDC dc, HGDIOBJ handle) [pure virtual]`

Create a new metarecord which describes this object.

Parameters

<i>dc</i>	the handle to the device context.
<i>handle</i>	(appears not to used. Note the handle is really assigned at serialization time.)

Implemented in [EMF::PEN](#), [EMF::EXTPEN](#), [EMF::BRUSH](#), [EMF::FONT](#), and [EMF::PALETTE](#).

4.63.3 Field Documentation**4.63.3.1 contexts** `std::map< HDC, HGDIOBJ > EMF::GRAPHICSOBJECT::contexts`

A set of all the contexts into which this object has been selected and the associated metafile handle for the object.

Referenced by [EMF::PEN::newEMR\(\)](#), [EMF::EXTPEN::newEMR\(\)](#), [EMF::BRUSH::newEMR\(\)](#), [EMF::FONT::newEMR\(\)](#), and [EMF::PALETTE::newEMR\(\)](#).

The documentation for this class was generated from the following file:

- `libemf.h`

4.64 EMF::INTARRAY Struct Reference

Represent an array of integers in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [INTARRAY](#) (`INT *const ints, const DWORD n`)

Data Fields

- `INT *const ints_`
Array of ints.
- `const DWORD n_`
Number of ints in array.

4.64.1 Detailed Description

Represent an array of integers in a simple way.

Allow an array of INT's to be written out at once.

4.64.2 Constructor & Destructor Documentation**4.64.2.1 INTARRAY()** `EMF::INTARRAY::INTARRAY (`

```
    INT *const ints,
    const DWORD n ) [inline]
```

simple constructor.

Parameters

<i>ints</i>	pointer to ints.
<i>n</i>	number ints in array.

The documentation for this struct was generated from the following file:

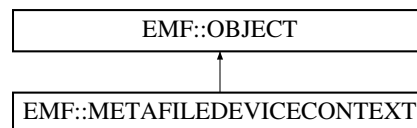
- libemf.h

4.65 EMF::METAFILEDEVICECONTEXT Class Reference

Graphics Device Context.

```
#include <libemf.h>
```

Inheritance diagram for EMF::METAFILEDEVICECONTEXT:



Public Member Functions

- [METAFILEDEVICECONTEXT](#) (FILE *fp_, const RECT *size, LPCWSTR description_w)
- virtual [~METAFILEDEVICECONTEXT](#) ()
- OBJECTTYPE [getType](#) (void) const
- DWORD [nextHandle](#) (void)
- void [clearHandle](#) (DWORD handle)
- void [appendRecord](#) (METARECORD *record)
- void [appendHandle](#) (METARECORD *record)
- void [deleteMetafile](#) (void)
- void [mergePoint](#) (const LONG &x, const LONG &y)
- void [mergePoint](#) (const POINT &p)

Data Fields

- ::FILE * [fp](#)
- DATASTREAM [ds](#)
- ENHMETAHEADER * [header](#)
- std::vector< [EMF::METARECORD](#) * > [records](#)
- SIZEL [resolution](#)
The resolution in DPI of the reference DC.
- SIZEL [viewport_ext](#)
The extent of the viewport.
- POINT [viewport_org](#)
The origin of the viewport.
- SIZEL [window_ext](#)

- The extent of the window.*
 - POINT **window_org**
- The origin of the window.*
 - bool **update_frame**
- Update the frame automatically?*
 - POINT **min_device_point**
- The lft/top-most painted point in device units.*
 - POINT **max_device_point**
- The rgt/btm-most painted point in device units.*
 - POINT **point**
- The current point.*
 - **PEN** * **pen**
- The current pen.*
 - **BRUSH** * **brush**
- The current brush.*
 - **FONT** * **font**
- The current font.*
 - **PALETTE** * **palette**
- The current palette.*
 - UINT **text_alignment**
- The current text alignment.*
 - COLORREF **text_color**
- The current text foreground color.*
 - COLORREF **bk_color**
- The current background color.*
 - INT **bk_mode**
- The current background mode.*
 - INT **polyfill_mode**
- The current polygon fill mode.*
 - INT **map_mode**
- The current mapping mode.*
 - FLOAT **miter_limit**
- The current miter length limit.*
 - std::vector< bool > [handles](#)
 - std::map< HGDIOBJ, HGDIOBJ > [emf_handles](#)

4.65.1 Detailed Description

Graphics Device Context.

Almost all GDI graphics calls require a device context (except those which create graphics objects such as pens and fonts). This is a specific context which renders to a metafile. There is a one-to-one correspondence between the device context and the metafile.

4.65.2 Constructor & Destructor Documentation

4.65.2.1 METAFILEDEVICECONTEXT() `EMF::METAFILEDEVICECONTEXT::METAFILEDEVICECONTEXT (FILE * fp_, const RECT * size, LPCWSTR description_w) [inline]`

Most graphics programs seem to want to handle the opening and closing of files themselves, so this is an extension to the w32 interface.

Parameters

<i>fp_</i>	stdio pointer to an open file. May be null.
<i>size</i>	the rectangle describing the position and size of the metafile on the "page". May be null.
<i>description↵ _w</i>	a UNICODE string describing the metafile. The format must be "some text\0some more text\0\0". May be null.

4.65.2.2 ~METAFILEDEVICECONTEXT() `virtual EMF::METAFILEDEVICECONTEXT::~~METAFILEDEVICECONTEXT () [inline], [virtual]`

Destructor frees all the graphics objects which may have been allocated. Now, it also frees any metarecords which it might hold, too.

References [deleteMetafile\(\)](#), and [records](#).

4.65.3 Member Function Documentation

4.65.3.1 appendHandle() `void EMF::METAFILEDEVICECONTEXT::appendHandle (METARECORD * record) [inline]`

Add this record to the metafile.

Parameters

<i>record</i>	this record is an object so it increments the handle count as well.
---------------	---

References [header](#), [records](#), and [EMF::METARECORD::size\(\)](#).

4.65.3.2 appendRecord() `void EMF::METAFILEDEVICECONTEXT::appendRecord (METARECORD * record) [inline]`

Add this record to the metafile.

Parameters

<i>record</i>	standard graphics record
---------------	--------------------------

References [header](#), [records](#), and [EMF::METARECORD::size\(\)](#).

4.65.3.3 clearHandle() `void EMF::METAFILEDEVICECONTEXT::clearHandle (`
`DWORD handle) [inline]`

Clear the usage of this handle

References [EMF::OBJECT::handle](#), and [handles](#).

4.65.3.4 deleteMetafile() `void EMF::METAFILEDEVICECONTEXT::deleteMetafile (`
`void) [inline]`

Delete all the records from the metafile. This would seem to include deleting the header record as well.

References [records](#).

Referenced by [~METAFILEDEVICECONTEXT\(\)](#).

4.65.3.5 getType() `OBJECTTYPE EMF::METAFILEDEVICECONTEXT::getType (`
`void) const [inline], [virtual]`

Return the type of this object (could probably do better with RTTI()).

Implements [EMF::OBJECT](#).

4.65.3.6 mergePoint() [1/2] `void EMF::METAFILEDEVICECONTEXT::mergePoint (`
`const LONG & x,`
`const LONG & y) [inline]`

Somewhat superfluous, except checker doesn't understand the initialization of automatic structures in the declaration.

References [mergePoint\(\)](#).

Referenced by [mergePoint\(\)](#).

4.65.3.7 mergePoint() [2/2] `void EMF::METAFILEDEVICECONTEXT::mergePoint (`
`const POINT & p) [inline]`

Take the given point and determine if it enlarges the "painted" area of the device.

References [header](#), [max_device_point](#), [min_device_point](#), [update_frame](#), [viewport_ext](#), [viewport_org](#), [window_ext](#), and [window_org](#).

4.65.3.8 nextHandle() `DWORD EMF::METAFILEDEVICECONTEXT::nextHandle (void) [inline]`

Scan the bit vector of used handles and return the index of the first free bit as this objects metafile handle.

References [handles](#), and [header](#).

4.65.4 Field Documentation

4.65.4.1 ds `DATASTREAM EMF::METAFILEDEVICECONTEXT::ds`

All i/o to the metafile is wrapped by this class so that byte swapping on big-endian machines is transparent.

4.65.4.2 emf_handles `std::map< HGDIOBJ, HGDIOBJ > EMF::METAFILEDEVICECONTEXT::emf_handles`

This map holds the *current* mapping between EMF handles and global object handles as a metafile is played back (with PlayEnhMetaFile).

Referenced by [EMF::EMRSELECTOBJECT::execute\(\)](#), [EMF::EMRDELETEOBJECT::execute\(\)](#), [EMF::EMRCREATEPEN::execute\(\)](#), [EMF::EMREXTCREATEPEN::execute\(\)](#), [EMF::EMRCREATEBRUSHINDIRECT::execute\(\)](#), and [EMF::EMREXTCREATEFONTINDIRECT::execute\(\)](#).

4.65.4.3 fp `::FILE* EMF::METAFILEDEVICECONTEXT::fp`

If it is a file-based metafile, then this pointer is not null.

4.65.4.4 handles `std::vector< bool > EMF::METAFILEDEVICECONTEXT::handles`

For compatibility, it appears that metafile handles are reused as objects are deleted. Attempt to emulate that behavior with a bit vector of used metafile handles.

Referenced by [clearHandle\(\)](#), and [nextHandle\(\)](#).

4.65.4.5 header `ENHMETAHEADER* EMF::METAFILEDEVICECONTEXT::header`

Serves double duty as the physical device description.

Referenced by [appendHandle\(\)](#), [appendRecord\(\)](#), [mergePoint\(\)](#), and [nextHandle\(\)](#).

Referenced by [appendHandle\(\)](#), [appendRecord\(\)](#), [deleteMetafile\(\)](#), and [~METAFILEDEVICECONTEXT\(\)](#).

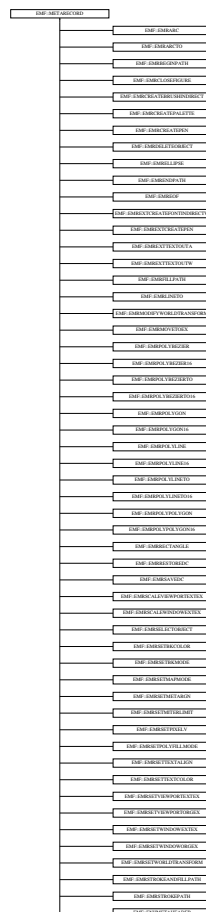
The documentation for this class was generated from the following file:

- libemf.h

4.66 EMF::METARECORD Class Reference

```
#include <libemf.h>
```

Inheritance diagram for EMF::METARECORD:



- virtual void **execute** (**METAFILEDVICECONTEXT** *source, HDC dc) const =0
- virtual bool **serialize** (**DATASTREAM** ds)=0
- virtual int **size** (void) const =0
- virtual **~METARECORD** ()

4.66.1 Detailed Description

The base class of all metafile records.

A metafile consists off a sequence of graphics records "executed" in order. This is a common base class that allows each, different, record to be stored in a common list. An interface is specified for each record to write itself to a file.

4.66.2 Constructor & Destructor Documentation

4.66.2.1 `~METARECORD()` `virtual EMF::METARECORD::~METARECORD () [inline], [virtual]`

The virtual destructor allows records which allocated additional memory to release it when they are deleted. Simple records just use the default destructor defined here.

4.66.3 Member Function Documentation

4.66.3.1 `execute()` `virtual void EMF::METARECORD::execute (METAFILEDEVICECONTEXT * source, HDC dc) const [pure virtual]`

Execute the graphics command in the given context. Used by PlayEnhMetaFile to "copy" one metafile into another.

Parameters

<i>source</i>	the device context from which this record is taken.
<i>dc</i>	the destination context.

Implemented in [EMF::ENHMETAHEADER](#), [EMF::EMREOF](#), [EMF::EMRSETVIEWPORTORGEX](#), [EMF::EMRSETWINDOWORGEX](#), [EMF::EMRSETVIEWPORTEXTEX](#), [EMF::EMRSCALEVIEWPORTEXTEX](#), [EMF::EMRSETWINDOWEXTEX](#), [EMF::EMRSCALEWINDOWEXTEX](#), [EMF::EMRMODIFYWORLDTRANSFORM](#), [EMF::EMRSETWORLDTRANSFORM](#), [EMF::EMRSETTEXTALIGN](#), [EMF::EMRSETTEXTCOLOR](#), [EMF::EMRSETBKCOLOR](#), [EMF::EMRSETBKMODE](#), [EMF::EMRSETPOLYFILLMODE](#), [EMF::EMRSETMAPMODE](#), [EMF::EMRSELECTOBJECT](#), [EMF::EMRDELETEOBJECT](#), [EMF::EMRMOVETOEX](#), [EMF::EMRLINETO](#), [EMF::EMRARC](#), [EMF::EMRARCTO](#), [EMF::EMRRECTANGLE](#), [EMF::EMRELLIPSE](#), [EMF::EMRPOLYLINE](#), [EMF::EMRPOLYLINE16](#), [EMF::EMRPOLYGON](#), [EMF::EMRPOLYGON16](#), [EMF::EMRPOLYPOLYGON](#), [EMF::EMRPOLYPOLYGON16](#), [EMF::EMRPOLYBEZIER](#), [EMF::EMRPOLYBEZIER16](#), [EMF::EMRPOLYBEZIERTO](#), [EMF::EMRPOLYBEZIERTO16](#), [EMF::EMRPOLYLINETO](#), [EMF::EMRPOLYLINETO16](#), [EMF::EMREXTTEXTOUTA](#), [EMF::EMREXTTEXTOUTW](#), [EMF::EMRSETPIXELV](#), [EMF::EMRCREATEPEN](#), [EMF::EMREXTCREATEPEN](#), [EMF::EMRCREATEBRUSHINDIRECT](#), [EMF::EMREXTCREATEFONTINDIRECTW](#), [EMF::EMRCREATEPALETTE](#), [EMF::EMRFILLPATH](#), [EMF::EMRSTROKEPATH](#), [EMF::EMRSTROKEANDFILLPATH](#), [EMF::EMRBEGINPATH](#), [EMF::EMRENDPATH](#), [EMF::EMRCLOSEFIGURE](#), [EMF::EMRSAVEDC](#), [EMF::EMRSTOREDC](#), [EMF::EMRSETMETARGN](#), and [EMF::EMRSETMITERLIMIT](#).

4.66.3.2 serialize() `virtual bool EMF::METARECORD::serialize (
 DATASTREAM ds) [pure virtual]`

Write yourself to the given file. This is virtual since some records are of arbitrary length and need to write additional information after their EMR structure.

Parameters

<code>ds</code>	the datastream to write oneself to.
-----------------	-------------------------------------

Implemented in [EMF::ENHMETAHEADER](#), [EMF::EMREOF](#), [EMF::EMRSETVIEWPORTORGEX](#), [EMF::EMRSETWINDOWORGEX](#), [EMF::EMRSETVIEWPORTEXT](#), [EMF::EMRSCALEVIEWPORTEXT](#), [EMF::EMRSETWINDOWEXT](#), [EMF::EMRSCALEWINDOWEXT](#), [EMF::EMRMODIFYWORLDTRANSFORM](#), [EMF::EMRSETWORLDTRANSFORM](#), [EMF::EMRSETTEXTALIGN](#), [EMF::EMRSETTEXTCOLOR](#), [EMF::EMRSETBKCOLOR](#), [EMF::EMRSETBKMODE](#), [EMF::EMRSETPOLYFILLMODE](#), [EMF::EMRSETMAPMODE](#), [EMF::EMRSELECTOBJECT](#), [EMF::EMRDELETEOBJECT](#), [EMF::EMRMOVETOEX](#), [EMF::EMRLINETO](#), [EMF::EMRARC](#), [EMF::EMRARCTO](#), [EMF::EMRRECTANGLE](#), [EMF::EMRELLIPSE](#), [EMF::EMRPOLYLINE](#), [EMF::EMRPOLYLINE16](#), [EMF::EMRPOLYGON](#), [EMF::EMRPOLYGON16](#), [EMF::EMRPOLYPOLYGON](#), [EMF::EMRPOLYPOLYGON16](#), [EMF::EMRPOLYBEZIER](#), [EMF::EMRPOLYBEZIER16](#), [EMF::EMRPOLYBEZIERTO](#), [EMF::EMRPOLYBEZIERTO16](#), [EMF::EMRPOLYLINETO](#), [EMF::EMRPOLYLINETO16](#), [EMF::EMREXTTEXTOUTA](#), [EMF::EMREXTTEXTOUTW](#), [EMF::EMRSETPIXELV](#), [EMF::EMRCREATEPEN](#), [EMF::EMREXTCREATEPEN](#), [EMF::EMRCREATEBRUSHINDIRECT](#), [EMF::EMREXTCREATEFONTINDIRECTW](#), [EMF::EMRCREATEPALETTE](#), [EMF::EMRFILLPATH](#), [EMF::EMRSTROKEPATH](#), [EMF::EMRSTROKEANDFILLPATH](#), [EMF::EMRBEGINPATH](#), [EMF::EMRENDPATH](#), [EMF::EMRCLOSEFIGURE](#), [EMF::EMRSAVEDC](#), [EMF::EMRSTOREDC](#), [EMF::EMRSETMETARGN](#), and [EMF::EMRSETMITERLIMIT](#).

4.66.3.3 size() `virtual int EMF::METARECORD::size (
 void) const [pure virtual]`

The header record of a metafile records the total size of the metafile in bytes, so as each record is added to the list, it updates the total size.

Implemented in [EMF::ENHMETAHEADER](#), [EMF::EMREOF](#), [EMF::EMRSETVIEWPORTORGEX](#), [EMF::EMRSETWINDOWORGEX](#), [EMF::EMRSETVIEWPORTEXT](#), [EMF::EMRSCALEVIEWPORTEXT](#), [EMF::EMRSETWINDOWEXT](#), [EMF::EMRSCALEWINDOWEXT](#), [EMF::EMRMODIFYWORLDTRANSFORM](#), [EMF::EMRSETWORLDTRANSFORM](#), [EMF::EMRSETTEXTALIGN](#), [EMF::EMRSETTEXTCOLOR](#), [EMF::EMRSETBKCOLOR](#), [EMF::EMRSETBKMODE](#), [EMF::EMRSETPOLYFILLMODE](#), [EMF::EMRSETMAPMODE](#), [EMF::EMRSELECTOBJECT](#), [EMF::EMRDELETEOBJECT](#), [EMF::EMRMOVETOEX](#), [EMF::EMRLINETO](#), [EMF::EMRARC](#), [EMF::EMRARCTO](#), [EMF::EMRRECTANGLE](#), [EMF::EMRELLIPSE](#), [EMF::EMRPOLYLINE](#), [EMF::EMRPOLYLINE16](#), [EMF::EMRPOLYGON](#), [EMF::EMRPOLYGON16](#), [EMF::EMRPOLYPOLYGON](#), [EMF::EMRPOLYPOLYGON16](#), [EMF::EMRPOLYBEZIER](#), [EMF::EMRPOLYBEZIER16](#), [EMF::EMRPOLYBEZIERTO](#), [EMF::EMRPOLYBEZIERTO16](#), [EMF::EMRPOLYLINETO](#), [EMF::EMRPOLYLINETO16](#), [EMF::EMREXTTEXTOUTA](#), [EMF::EMREXTTEXTOUTW](#), [EMF::EMRSETPIXELV](#), [EMF::EMRCREATEPEN](#), [EMF::EMREXTCREATEPEN](#), [EMF::EMRCREATEBRUSHINDIRECT](#), [EMF::EMREXTCREATEFONTINDIRECTW](#), [EMF::EMRCREATEPALETTE](#), [EMF::EMRFILLPATH](#), [EMF::EMRSTROKEPATH](#), [EMF::EMRSTROKEANDFILLPATH](#), [EMF::EMRBEGINPATH](#), [EMF::EMRENDPATH](#), [EMF::EMRCLOSEFIGURE](#), [EMF::EMRSAVEDC](#), [EMF::EMRSTOREDC](#), [EMF::EMRSETMETARGN](#), and [EMF::EMRSETMITERLIMIT](#).

Referenced by [EMF::METAFILEDEVICECONTEXT::appendHandle\(\)](#), and [EMF::METAFILEDEVICECONTEXT::appendRecord\(\)](#).

The documentation for this class was generated from the following file:

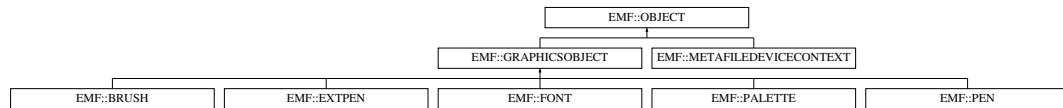
- [libemf.h](#)

4.67 EMF::OBJECT Class Reference

Global GDI object.

```
#include <libemf.h>
```

Inheritance diagram for EMF::OBJECT:



Public Member Functions

- virtual **~OBJECT** ()
OBJECTs have a virtual destructor.
- **OBJECT** (void)
- virtual OBJECTTYPE **getType** (void) const =0

Data Fields

- HGDIOBJ **handle**

4.67.1 Detailed Description

Global GDI object.

The GDI interface defines objects in terms of handles (rather than pointers). In order to emulate this, each object is placed into a global list and the index in that list becomes their handle.

4.67.2 Constructor & Destructor Documentation

4.67.2.1 OBJECT() `EMF::OBJECT::OBJECT (void) [inline]`

Create a new object. It's up to a subclass to create a real handle for this object.

4.67.3 Member Function Documentation

4.67.3.1 **getType()** `virtual OBJECTTYPE EMF::OBJECT::getType (`
`void) const [pure virtual]`

Return the type of the object.

Implemented in [EMF::PEN](#), [EMF::EXTPEN](#), [EMF::BRUSH](#), [EMF::FONT](#), [EMF::PALETTE](#), and [EMF::METAFILEDEVICECONTEXT](#).

4.67.4 Field Documentation

4.67.4.1 **handle** `HGDIOBJ EMF::OBJECT::handle`

The handle of a GDI object.

Referenced by [EMF::METAFILEDEVICECONTEXT::clearHandle\(\)](#).

The documentation for this class was generated from the following file:

- [libemf.h](#)

4.68 EMF::PADDING Struct Reference

All metafile records must be padded out to a multiple of 4 bytes.

```
#include <libemf.h>
```

Public Member Functions

- [PADDING](#) (const int size)

Data Fields

- const int **size_**
Number of bytes of padding.

Static Public Attributes

- static const char **padding_**[4] = { 0, 0, 0, 0 }
Pad with '0's.

4.68.1 Detailed Description

All metafile records must be padded out to a multiple of 4 bytes.

Write out a few bytes of padding if necessary.

4.68.2 Constructor & Destructor Documentation

4.68.2.1 **PADDING()** `EMF::PADDING::PADDING (`
`const int size) [inline]`

simple constructor.

Parameters

<i>size</i>	number of bytes of padding to use.
-------------	------------------------------------

The documentation for this struct was generated from the following files:

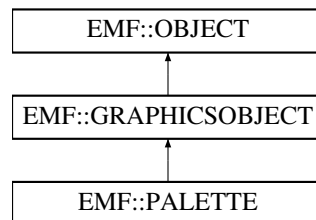
- libemf.h
- libemf.cpp

4.69 EMF::PALETTE Class Reference

Graphics Palette.

```
#include <libemf.h>
```

Inheritance diagram for EMF::PALETTE:



Public Member Functions

- [PALETTE](#) (const LOGPALETTE *lpalette)
- OBJECTTYPE [getType](#) (void) const
- [METARECORD](#) * [newEMR](#) (HDC dc, HGDIOBJ emf_handle)

Additional Inherited Members

4.69.1 Detailed Description

Graphics Palette.

Not entirely sure how palettes are used in general.

4.69.2 Constructor & Destructor Documentation

4.69.2.1 PALETTE() EMF::PALETTE::PALETTE (
const LOGPALETTE * lpalette) [inline]

Parameters

<i>palette</i>	the (logical?) palette definition.
----------------	------------------------------------

4.69.3 Member Function Documentation

4.69.3.1 `getType()` `OBJECTTYPE EMF::PALETTE::getType (void) const [inline], [virtual]`

Return the type of this object (could probably do better with `RTTI()`).

Implements [EMF::OBJECT](#).

4.69.3.2 `newEMR()` `METARECORD * EMF::PALETTE::newEMR (HDC dc, HGDIOBJ emf_handle) [inline], [virtual]`

Return a new metarecord for this object. And record its selection into the given device context.

Parameters

<i>dc</i>	handle of device context into which this object is being selected.
<i>emf_handle</i>	the EMF handle associated with the FONT .

Implements [EMF::GRAPHICSOBJECT](#).

References [EMF::GRAPHICSOBJECT::contexts](#).

The documentation for this class was generated from the following file:

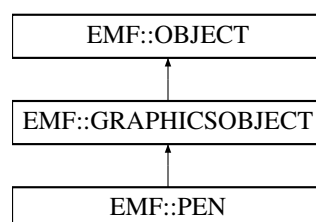
- `libemf.h`

4.70 EMF::PEN Class Reference

Graphics Pen.

```
#include <libemf.h>
```

Inheritance diagram for `EMF::PEN`:



Public Member Functions

- [PEN](#) (const LOGPEN *lpen)
- OBJECTTYPE [getType](#) (void) const
- [METARECORD](#) * [newEMR](#) (HDC dc, HGDIOBJ emf_handle)

Additional Inherited Members

4.70.1 Detailed Description

Graphics Pen.

Pens are used for drawing lines, arc, rectangles, etc.

4.70.2 Constructor & Destructor Documentation

4.70.2.1 PEN() `EMF::PEN::PEN (const LOGPEN * lpen) [inline]`

Parameters

<i>lpen</i>	the (logical?) pen definition.
-------------	--------------------------------

4.70.3 Member Function Documentation

4.70.3.1 getType() `OBJECTTYPE EMF::PEN::getType (void) const [inline], [virtual]`

Return the type of this object (could probably do better with RTTI()).

Implements [EMF::OBJECT](#).

4.70.3.2 newEMR() `METARECORD * EMF::PEN::newEMR (HDC dc, HGDIOBJ emf_handle) [inline], [virtual]`

Return a new metarecord for this object. And record its selection into the given device context.

Parameters

<i>dc</i>	handle of device context into which this object is being selected.
<i>emf_handle</i>	the EMF handle associated with the PEN .

Implements [EMF::GRAPHICSOBJECT](#).

References [EMF::GRAPHICSOBJECT::contexts](#).

The documentation for this class was generated from the following file:

- libemf.h

4.71 EMF::POINT16ARRAY Struct Reference

Represent an array of 16-bit point in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [POINT16ARRAY](#) (POINT16 *const points, const DWORD n)

Data Fields

- POINT16 *const **points_**
Array of POINT16s.
- const DWORD **n_**
Number of POINT16s in array.

4.71.1 Detailed Description

Represent an array of 16-bit point in a simple way.

Allow an array of POINT16's to be written out at once.

4.71.2 Constructor & Destructor Documentation

4.71.2.1 POINT16ARRAY() `EMF::POINT16ARRAY::POINT16ARRAY (`
POINT16 *const *points*,
const DWORD *n*) [inline]

Simple constructor.

Parameters

<i>points</i>	pointer to array of POINT16s.
<i>n</i>	number POINT16s in array.

The documentation for this struct was generated from the following file:

- libemf.h

4.72 EMF::POINTLARRAY Struct Reference

Represent an array of points in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [POINTLARRAY](#) (POINTL *const points, const DWORD n)

Data Fields

- POINTL *const **points_**
Array of POINTLs.
- const DWORD **n_**
Number of POINTLs in array.

4.72.1 Detailed Description

Represent an array of points in a simple way.

Allow an array of POINTL's to be written out at once.

4.72.2 Constructor & Destructor Documentation

4.72.2.1 POINTLARRAY() EMF::POINTLARRAY::POINTLARRAY (
POINTL *const *points*,
const DWORD *n*) [inline]

Simple constructor.

Parameters

<i>points</i>	pointer to array of POINTLs.
<i>n</i>	number POINTLs in array.

The documentation for this struct was generated from the following file:

- libemf.h

4.73 EMF::WCHARSTR Struct Reference

Represent a wide (UNICODE) character string in a simple way.

```
#include <libemf.h>
```

Public Member Functions

- [WCHARSTR](#) (WCHAR *const string, const int length)

Data Fields

- WCHAR *const **string_**
String of WCHARs.
- const int **length_**
Number of WCHARs in string.

4.73.1 Detailed Description

Represent a wide (UNICODE) character string in a simple way.

Even (wchar) strings have to be byte swapped. This structure allows us to provide a uniform stream-like interface for writing out all the components of metafiles.

4.73.2 Constructor & Destructor Documentation

4.73.2.1 WCHARSTR() `EMF::WCHARSTR::WCHARSTR (`
 WCHAR *const *string*,
 const int *length*) `[inline]`

Simple constructor.

Parameters

<i>string</i>	pointer to string of WCHARs.
<i>length</i>	number of WCHARs in string.

The documentation for this struct was generated from the following file:

- libemf.h

5 File Documentation

5.1 emf.h

```

1 /*
2  * EMF: A library for generating ECMA-234 Enhanced Metafiles
3  * Copyright (C) 2002 lignum Computing, Inc. <dallenbarnett@users.sourceforge.net>
4  * $Id: emf.h 93 2020-04-18 13:30:11Z dallenbarnett $
5  *
6  * This library is free software; you can redistribute it and/or
7  * modify it under the terms of the GNU Lesser General Public
8  * License as published by the Free Software Foundation; either
9  * version 2.1 of the License, or (at your option) any later version.
10 *
11 * This library is distributed in the hope that it will be useful,
12 * but WITHOUT ANY WARRANTY; without even the implied warranty of
13 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
14 * Lesser General Public License for more details.
15 *
16 * You should have received a copy of the GNU Lesser General Public
17 * License along with this library; if not, write to the Free Software
18 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
19 *
20 */
21 #ifndef _EMF_H
22 #define _EMF_H
23
24 #include <stdio.h>
25 #include <string.h>
26
27 #include <libEMF/wine/wine.h>
28 #include <libEMF/wine/winbase.h>
29 #include <libEMF/wine/wingdi.h>
30 #include <libEMF/wine/winuser.h>
31 #include <libEMF/wine/winerror.h>
32
33 #ifdef __cplusplus
34 extern "C" {
35 #endif
36 /*
37  * Here are additional, non-"standard" routines which the author deems useful.
38  */
39 HDC CreateEnhMetaFileWithFILEA( HDC context, FILE* fp, const RECT* size,
40                                LPCSTR description );
41 HDC CreateEnhMetaFileWithFILEW( HDC context, FILE* fp, const RECT* size,
42                                LPCWSTR description );
43 HENHMETAFILE CloseEnhMetaFileWithFILE( HDC context );
44 /*
45  * This function will only produce output if the library has been compiled with
46  * editing enabled (e.g., ./configure --enable-editing).
47  */
48 void WINAPI EditEnhMetaFile ( HENHMETAFILE metafile );
49 #ifdef __cplusplus
50 }
51 #endif
52
53 #endif /* _EMF_H */

```

5.2 basetsd.h

```

1 /*
2  * Compilers that uses ILP32, LP64 or P64 type models
3  * for both Win32 and Win64 are supported by this file.
4  */
5
6 #ifndef __WINE_BASETSD_H
7 #define __WINE_BASETSD_H
8
9 #ifdef __cplusplus
10 extern "C" {
11 #endif /* defined(__cplusplus) */
12
13 /*
14  * Win32 was easy to implement under Unix since most (all?) 32-bit
15  * Unices uses the same type model (ILP32) as Win32, where int, long
16  * and pointer are 32-bit.
17  *
18  * Win64, however, will cause some problems when implemented under Unix.
19  * Linux/(Alpha, Sparc64) and most (all?) other 64-bit Unices uses
20  * the LP64 type model where int is 32-bit and long and pointer are
21  * 64-bit. Win64 on the other hand uses the P64 (sometimes called LLP64)
22  * type model where int and long are 32 bit and pointer is 64-bit.

```

```

23 */
24
25 /* Type model indepent typedefs */
26
27 typedef char      __int8;
28 typedef unsigned char __uint8;
29
30 typedef short     __int16;
31 typedef unsigned short __uint16;
32
33 typedef int       __int32;
34 typedef unsigned int __uint32;
35
36 // wogl
37 //typedef long long      __int64;
38 //typedef unsigned long long __uint64;
39 typedef long           __int64;
40 typedef unsigned long __uint64;
41
42 #if defined(_WIN64)
43
44 typedef __uint32 __ptr32;
45 typedef void     *__ptr64;
46
47 #else /* FIXME: defined(_WIN32) */
48
49 typedef void     *__ptr32;
50 typedef __uint64 __ptr64;
51
52 #endif
53
54 /* Always signed and 32 bit wide */
55
56 typedef __int32 LONG32;
57 typedef __int32 INT32;
58
59 typedef LONG32 *PLONG32;
60 typedef INT32  *PINT32;
61
62 /* Always unsigned and 32 bit wide */
63
64 typedef __uint32 ULONG32;
65 typedef __uint32 DWORD32;
66 typedef __uint32 UINT32;
67
68 typedef ULONG32 *PULONG32;
69 typedef DWORD32 *PDWORD32;
70 typedef UINT32  *PUINT32;
71
72 /* Always signed and 64 bit wide */
73
74 typedef __int64 LONG64;
75 typedef __int64 INT64;
76
77 typedef LONG64 *PLONG64;
78 typedef INT64  *PINT64;
79
80 /* Always unsigned and 64 bit wide */
81
82 typedef __uint64 ULONG64;
83 typedef __uint64 DWORD64;
84 typedef __uint64 UINT64;
85
86 typedef ULONG64 *PULONG64;
87 typedef DWORD64 *PDWORD64;
88 typedef UINT64  *PUINT64;
89
90 /* Win32 or Win64 dependent typedef/defines. */
91
92 #ifdef _WIN64
93
94 typedef __int64 INT_PTR, *PINT_PTR;
95 typedef __uint64 UINT_PTR, *PUINT_PTR;
96
97 #define MAXINT_PTR 0x7fffffffffffffff
98 #define MININT_PTR 0x8000000000000000
99 #define MAXUINT_PTR 0xffffffffffffffff
100
101 typedef __int32 HALF_PTR, *PHALF_PTR;
102 typedef __int32 UHALF_PTR, *PUHALF_PTR;
103
104 #define MAXHALF_PTR 0x7fffffff
105 #define MINHALF_PTR 0x80000000
106 #define MAXUHALF_PTR 0xffffffff
107
108 typedef __int64 LONG_PTR, *PLONG_PTR;
109 typedef __uint64 ULONG_PTR, *PULONG_PTR;

```

```

110 typedef __uint64 DWORD_PTR, *PDWORD_PTR;
111
112 #else /* FIXME: defined(_WIN32) */
113
114 typedef __int32 INT_PTR, *PINT_PTR;
115 typedef __uint32 UINT_PTR, *PUINT_PTR;
116
117 #define MAXINT_PTR 0x7fffffff
118 #define MININT_PTR 0x80000000
119 #define MAXUINT_PTR 0xffffffff
120
121 typedef __int16 HALF_PTR, *PHALF_PTR;
122 typedef __uint16 UHALF_PTR, *PUHALF_PTR;
123
124 #define MAXUHALF_PTR 0xffff
125 #define MAXHALF_PTR 0x7fff
126 #define MINHALF_PTR 0x8000
127
128 typedef __int32 LONG_PTR, *PLONG_PTR;
129 typedef __uint32 ULONG_PTR, *PULONG_PTR;
130 typedef __uint32 DWORD_PTR, *PDWORD_PTR;
131
132 #endif /* defined(_WIN64) || defined(_WIN32) */
133
134 typedef INT_PTR SSIZE_T, *PSSIZE_T;
135 typedef UINT_PTR SIZE_T, *PSIZE_T;
136
137 #ifdef __cplusplus
138 } /* extern "C" */
139 #endif /* defined(__cplusplus) */
140
141 #endif /* !defined(__WINE_BASSETSD_H) */
142
143
144

```

5.3 guiddef.h

```

1 #ifndef GUID_DEFINED
2 #define GUID_DEFINED
3 typedef struct _GUID
4 {
5     unsigned long Data1;
6     unsigned short Data2;
7     unsigned short Data3;
8     unsigned char Data4[ 8 ];
9 } GUID;
10 #endif
11
12 #undef DEFINE_GUID
13
14 #ifdef INITGUID
15 #define DEFINE_GUID(name, l, w1, w2, b1, b2, b3, b4, b5, b6, b7, b8) \
16 const GUID name = \
17 { l, w1, w2, { b1, b2, b3, b4, b5, b6, b7, b8 } }
18 #else
19 #define DEFINE_GUID(name, l, w1, w2, b1, b2, b3, b4, b5, b6, b7, b8) \
20 extern const GUID name
21 #endif
22
23 #define DEFINE_OLEGUID(name, l, w1, w2) \
24 DEFINE_GUID(name, l, w1, w2, 0xC0,0,0,0,0,0,0,0,0,0,0,0)
25
26 #ifndef _GUIDDEF_H_
27 #define _GUIDDEF_H_
28
29 typedef GUID *LPGUID;
30 typedef GUID CLSID,*LPCLSID;
31 typedef GUID IID,*LPIID;
32 typedef GUID FMTID,*LPFMTID;
33
34 #if defined(__cplusplus) && !defined(CINTERFACE)
35 #define REFGUID const GUID &
36 #define REFCLSID const CLSID &
37 #define REFIID const IID &
38 #define REFFMTID const FMTID &
39 #else /* !defined(__cplusplus) && !defined(CINTERFACE) */
40 #define REFGUID const GUID* const
41 #define REFCLSID const CLSID* const
42 #define REFIID const IID* const
43 #define REFFMTID const FMTID* const
44 #endif /* !defined(__cplusplus) && !defined(CINTERFACE) */
45
46 #if defined(__cplusplus) && !defined(CINTERFACE)

```

```

47 #define IsEqualGUID(rguid1, rguid2) (!memcmp(&(rguid1), &(rguid2), sizeof(GUID)))
48 #else /* defined(__cplusplus) && !defined(CINTERFACE) */
49 #define IsEqualGUID(rguid1, rguid2) (!memcmp(rguid1, rguid2, sizeof(GUID)))
50 #endif /* defined(__cplusplus) && !defined(CINTERFACE) */
51 #define IsEqualIID(riid1, riid2) IsEqualGUID(riid1, riid2)
52 #define IsEqualCLSID(rclsid1, rclsid2) IsEqualGUID(rclsid1, rclsid2)
53
54 #if defined(__cplusplus) && !defined(CINTERFACE)
55 #include <string.h>
56 inline bool operator==(const GUID& guidOne, const GUID& guidOther)
57 {
58     return !memcmp(&guidOne, &guidOther, sizeof(GUID));
59 }
60 inline bool operator!=(const GUID& guidOne, const GUID& guidOther)
61 {
62     return !(guidOne == guidOther);
63 }
64 #endif
65
66 extern const IID GUID_NULL;
67 #define IID_NULL GUID_NULL
68 #define CLSID_NULL GUID_NULL
69 #define FMTID_NULL GUID_NULL
70
71 #endif /* _GUIDDEF_H_ */

```

5.4 poppack.h

```

1 #if defined(__WINE_PSHPACK_H3)
2 #   ifndef __WINE_INTERNAL_POPPACK
3 #       undef __WINE_PSHPACK_H3
4 #   endif
5 /* Depth == 3 */
6
7 #   if defined(__GNUC__) || defined(__SUNPRO_C) || defined(__SUNPRO_CC)
8 #       if __WINE_PSHPACK_H2 == 1
9 #           pragma pack(1)
10 #       elif __WINE_PSHPACK_H2 == 2
11 #           pragma pack(2)
12 #       elif __WINE_PSHPACK_H2 == 8
13 #           pragma pack(8)
14 #       else
15 #           pragma pack(4)
16 #       endif
17 #       elif !defined(RC_INVOKED)
18 #           error "Adjusting the alignment is not supported with this compiler"
19 #       endif
20
21 #   elif defined(__WINE_PSHPACK_H2)
22 #       ifndef __WINE_INTERNAL_POPPACK
23 #           undef __WINE_PSHPACK_H2
24 #       endif
25 /* Depth == 2 */
26
27 #       if defined(__GNUC__) || defined(__SUNPRO_C) || defined(__SUNPRO_CC)
28 #           if __WINE_PSHPACK_H == 1
29 #               pragma pack(1)
30 #           elif __WINE_PSHPACK_H == 2
31 #               pragma pack(2)
32 #           elif __WINE_PSHPACK_H == 8
33 #               pragma pack(8)
34 #           else
35 #               pragma pack(4)
36 #           endif
37 #       elif !defined(RC_INVOKED)
38 #           error "Adjusting the alignment is not supported with this compiler"
39 #       endif
40
41 #   elif defined(__WINE_PSHPACK_H)
42 #       ifndef __WINE_INTERNAL_POPPACK
43 #           undef __WINE_PSHPACK_H
44 #       endif
45 /* Depth == 1 */
46
47 #       if defined(__GNUC__) || defined(__SUNPRO_C)
48 #           pragma pack()
49 #       elif defined(__SUNPRO_CC)
50 /*#       warning "Assuming a default alignment of 4"*/
51 #           pragma pack(4)
52 #       elif !defined(RC_INVOKED)
53 #           error "Adjusting the alignment is not supported with this compiler"
54 #       endif
55
56 #   else

```



```

57 /* Depth == 0 ! */
58
59 #error "Popping alignment isn't possible since no alignment has been pushed"
60
61 #endif
62
63 #undef __WINE_INTERNAL_POPPACK

```

5.5 pshpack2.h

```

1 #if defined(__WINE_PSHPACK_H3)
2
3 /* Depth > 3 */
4 # error "Alignment nesting > 3 is not supported"
5
6 #else
7
8 # if !defined(__WINE_PSHPACK_H)
9 #   define __WINE_PSHPACK_H 2
10 #   /* Depth == 1 */
11 #   elif !defined(__WINE_PSHPACK_H2)
12 #   define __WINE_PSHPACK_H2 2
13 #   /* Depth == 2 */
14 #   define __WINE_INTERNAL_POPPACK
15 #   include "poppack.h"
16 #   elif !defined(__WINE_PSHPACK_H3)
17 #   define __WINE_PSHPACK_H3 2
18 #   /* Depth == 3 */
19 #   define __WINE_INTERNAL_POPPACK
20 #   include "poppack.h"
21 #   endif
22
23 # if defined(__GNUC__) || defined(__SUNPRO_C) || defined(__SUNPRO_CC)
24 #   pragma pack(2)
25 #   elif !defined(RC_INVOKED)
26 #     error "Adjusting the alignment is not supported with this compiler"
27 #   endif
28
29 #endif

```

5.6 pshpack4.h

```

1 #if defined(__WINE_PSHPACK_H3)
2
3 /* Depth > 3 */
4 # error "Alignment nesting > 3 is not supported"
5
6 #else
7
8 # if !defined(__WINE_PSHPACK_H)
9 #   define __WINE_PSHPACK_H 4
10 #   /* Depth == 1 */
11 #   elif !defined(__WINE_PSHPACK_H2)
12 #   define __WINE_PSHPACK_H2 4
13 #   /* Depth == 2 */
14 #   define __WINE_INTERNAL_POPPACK
15 #   include "poppack.h"
16 #   elif !defined(__WINE_PSHPACK_H3)
17 #   define __WINE_PSHPACK_H3 4
18 #   /* Depth == 3 */
19 #   define __WINE_INTERNAL_POPPACK
20 #   include "poppack.h"
21 #   endif
22
23 # if defined(__GNUC__) || defined(__SUNPRO_C) || defined(__SUNPRO_CC)
24 #   pragma pack(4)
25 #   elif !defined(RC_INVOKED)
26 #     error "Adjusting the alignment is not supported with this compiler"
27 #   endif
28
29 #endif

```

5.7 w16.h

```

1 /*
2 * These are some left-over definitions which are not supported
3 * in WINE any more, but still show up in metafiles. They are
4 * not exposed as API.

```

```

5 */
6 #ifndef W16_H
7 #define W16_H
8
9 #ifdef __cplusplus
10 extern "C" {
11 #endif
12
13 /* Standard data types */
14
15 typedef short          INT16;
16 typedef unsigned short UINT16;
17 typedef unsigned short BOOL16;
18
19 typedef HDC            HDC16;
20
21 /* The POINT structure */
22
23 typedef struct
24 {
25     INT16  x;
26     INT16  y;
27 } POINT16, *PPOINT16, *LPPOINT16;
28
29 typedef struct {
30     EMR      emr;
31     RECTL    rclBounds;
32     DWORD    cpts;
33     POINT16  apts[1];
34 } EMRPOLYLINE16, *PEMRPOLYLINE16,
35 EMRPOLYBEZIER16, *PEMRPOLYBEZIER16,
36 EMRPOLYGON16, *PEMRPOLYGON16,
37 EMRPOLYBEZIERTO16, *PEMRPOLYBEZIERTO16,
38 EMRPOLYLINETO16, *PEMRPOLYLINETO16;
39
40 typedef struct {
41     EMR      emr;
42     RECTL    rclBounds;
43     DWORD    nPolys;
44     DWORD    cpts;
45     DWORD    aPolyCounts[1];
46     POINT16  apts[1];
47 } EMRPOLYPOLYLINE16, *PEMRPOLYPOLYLINE16,
48 EMRPOLYPOLYGON16, *PEMRPOLYPOLYGON16;
49
50 BOOL      WINAPI PolyBezier16(HDC16, const POINT16*, INT16);
51 BOOL      WINAPI PolyBezierTo16(HDC16, const POINT16*, INT16);
52 BOOL      WINAPI Polyline16(HDC16, const POINT16*, INT16);
53 BOOL      WINAPI PolylineTo16(HDC16, const POINT16*, INT16);
54 BOOL      WINAPI Polygon16(HDC16, const POINT16*, INT16);
55 BOOL      WINAPI PolyPolygon16(HDC16, const POINT16*, const INT*, UINT16);
56 #ifdef __cplusplus
57 }
58 #endif
59
60
61 #endif /* W16_H */

```

5.8 winbase.h

```

1 #ifndef __WINE_WINBASE_H
2 #define __WINE_WINBASE_H
3
4 #ifndef RC_INVOKED
5 #include <stdarg.h>
6 #endif
7
8 #include "basetsd.h"
9 #include "windef.h"
10
11 #ifndef RC_INVOKED
12 #include <stdarg.h>
13 #endif
14
15 #ifdef __cplusplus
16 extern "C" {
17 #endif
18
19 /* Windows Exit Procedure flag values */
20 #define WEP_FREE_DLL      0
21 #define WEP_SYSTEM_EXIT  1
22
23 typedef DWORD CALLBACK (*LPTHREAD_START_ROUTINE) (LPVOID);
24

```

```

25 typedef VOID /*WINAPI*/ (*PFIBER_START_ROUTINE)( LPVOID lpFiberParameter );
26 typedef PFIBER_START_ROUTINE LPFIBER_START_ROUTINE;
27
28 typedef RTL_CRITICAL_SECTION CRITICAL_SECTION;
29 typedef PRTL_CRITICAL_SECTION PCRITICAL_SECTION;
30 typedef PRTL_CRITICAL_SECTION LPCRITICAL_SECTION;
31
32 typedef RTL_CRITICAL_SECTION_DEBUG CRITICAL_SECTION_DEBUG;
33 typedef PRTL_CRITICAL_SECTION_DEBUG PCRITICAL_SECTION_DEBUG;
34 typedef PRTL_CRITICAL_SECTION_DEBUG LPCRITICAL_SECTION_DEBUG;
35
36
37 #define EXCEPTION_DEBUG_EVENT 1
38 #define CREATE_THREAD_DEBUG_EVENT 2
39 #define CREATE_PROCESS_DEBUG_EVENT 3
40 #define EXIT_THREAD_DEBUG_EVENT 4
41 #define EXIT_PROCESS_DEBUG_EVENT 5
42 #define LOAD_DLL_DEBUG_EVENT 6
43 #define UNLOAD_DLL_DEBUG_EVENT 7
44 #define OUTPUT_DEBUG_STRING_EVENT 8
45 #define RIP_EVENT 9
46
47 typedef struct _EXCEPTION_DEBUG_INFO {
48     EXCEPTION_RECORD ExceptionRecord;
49     DWORD dwFirstChance;
50 } EXCEPTION_DEBUG_INFO;
51
52 typedef struct _CREATE_THREAD_DEBUG_INFO {
53     HANDLE hThread;
54     LPVOID lpThreadLocalBase;
55     LPTHREAD_START_ROUTINE lpStartAddress;
56 } CREATE_THREAD_DEBUG_INFO;
57
58 typedef struct _CREATE_PROCESS_DEBUG_INFO {
59     HANDLE hFile;
60     HANDLE hProcess;
61     HANDLE hThread;
62     LPVOID lpBaseOfImage;
63     DWORD dwDebugInfoFileOffset;
64     DWORD nDebugInfoSize;
65     LPVOID lpThreadLocalBase;
66     LPTHREAD_START_ROUTINE lpStartAddress;
67     LPVOID lpImageName;
68     WORD fUnicode;
69 } CREATE_PROCESS_DEBUG_INFO;
70
71 typedef struct _EXIT_THREAD_DEBUG_INFO {
72     DWORD dwExitCode;
73 } EXIT_THREAD_DEBUG_INFO;
74
75 typedef struct _EXIT_PROCESS_DEBUG_INFO {
76     DWORD dwExitCode;
77 } EXIT_PROCESS_DEBUG_INFO;
78
79 typedef struct _LOAD_DLL_DEBUG_INFO {
80     HANDLE hFile;
81     LPVOID lpBaseOfDll;
82     DWORD dwDebugInfoFileOffset;
83     DWORD nDebugInfoSize;
84     LPVOID lpImageName;
85     WORD fUnicode;
86 } LOAD_DLL_DEBUG_INFO;
87
88 typedef struct _UNLOAD_DLL_DEBUG_INFO {
89     LPVOID lpBaseOfDll;
90 } UNLOAD_DLL_DEBUG_INFO;
91
92 typedef struct _OUTPUT_DEBUG_STRING_INFO {
93     LPSTR lpDebugStringData;
94     WORD fUnicode;
95     WORD nDebugStringLength;
96 } OUTPUT_DEBUG_STRING_INFO;
97
98 typedef struct _RIP_INFO {
99     DWORD dwError;
100     DWORD dwType;
101 } RIP_INFO;
102
103 typedef struct _DEBUG_EVENT {
104     DWORD dwDebugEventCode;
105     DWORD dwProcessId;
106     DWORD dwThreadId;
107     union u11 {
108         EXCEPTION_DEBUG_INFO Exception;
109         CREATE_THREAD_DEBUG_INFO CreateThread;
110         CREATE_PROCESS_DEBUG_INFO CreateProcessInfo;
111         EXIT_THREAD_DEBUG_INFO ExitThread;

```

```

112     EXIT_PROCESS_DEBUG_INFO    ExitProcess;
113     LOAD_DLL_DEBUG_INFO        LoadDll;
114     UNLOAD_DLL_DEBUG_INFO      UnloadDll;
115     OUTPUT_DEBUG_STRING_INFO    DebugString;
116     RIP_INFO                    RipInfo;
117 } u;
118 } DEBUG_EVENT, *LPDEBUG_EVENT;
119
120 typedef PCONTEXT LPCONTEXT;
121 typedef PEXCEPTION_RECORD LPEXCEPTION_RECORD;
122 typedef PEXCEPTION_POINTERS LPEXCEPTION_POINTERS;
123
124 #define OFS_MAXPATHNAME 128
125 typedef struct
126 {
127     BYTE cBytes;
128     BYTE fFixedDisk;
129     WORD nErrCode;
130     BYTE reserved[4];
131     BYTE szPathName[OFS_MAXPATHNAME];
132 } OFSTRUCT, *POFSTRUCT, *LPOFSTRUCT;
133
134 #define OF_READ                0x0000
135 #define OF_WRITE                0x0001
136 #define OF_READWRITE            0x0002
137 #define OF_SHARE_COMPAT         0x0000
138 #define OF_SHARE_EXCLUSIVE     0x0010
139 #define OF_SHARE_DENY_WRITE    0x0020
140 #define OF_SHARE_DENY_READ     0x0030
141 #define OF_SHARE_DENY_NONE     0x0040
142 #define OF_PARSE                0x0100
143 #define OF_DELETE               0x0200
144 #define OF_VERIFY               0x0400 /* Used with OF_REOPEN */
145 #define OF_SEARCH               0x0400 /* Used without OF_REOPEN */
146 #define OF_CANCEL               0x0800
147 #define OF_CREATE               0x1000
148 #define OF_PROMPT               0x2000
149 #define OF_EXIST               0x4000
150 #define OF_REOPEN               0x8000
151
152 /* SetErrorMode values */
153 #define SEM_FAILCRITICALERRORS 0x0001
154 #define SEM_NOGPFAULTERRORBOX  0x0002
155 #define SEM_NOALIGNMENTFAULTEXCEPT 0x0004
156 #define SEM_NOOPENFILEERRORBOX 0x8000
157
158 /* CopyFileEx flags */
159 #define COPY_FILE_FAIL_IF_EXISTS 0x00000001
160 #define COPY_FILE_RESTARTABLE    0x00000002
161 #define COPY_FILE_OPEN_SOURCE_FOR_WRITE 0x00000004
162
163 /* GetTempFileName() Flags */
164 #define TF_FORCEDRIVE            0x80
165
166 #define DRIVE_UNKNOWN           0
167 #define DRIVE_NO_ROOT_DIR      1
168 #define DRIVE_REMOVABLE         2
169 #define DRIVE_FIXED             3
170 #define DRIVE_REMOTE            4
171 /* Win32 additions */
172 #define DRIVE_CDROM             5
173 #define DRIVE_RAMDISK           6
174
175 /* The security attributes structure */
176 typedef struct _SECURITY_ATTRIBUTES
177 {
178     DWORD    nLength;
179     LPVOID    lpSecurityDescriptor;
180     BOOL      bInheritHandle;
181 } SECURITY_ATTRIBUTES, *PSECURITY_ATTRIBUTES, *LPSECURITY_ATTRIBUTES;
182
183 #ifndef _FILETIME_
184 #define _FILETIME_
185 /* 64 bit number of 100 nanoseconds intervals since January 1, 1601 */
186 typedef struct
187 {
188     DWORD    dwLowDateTime;
189     DWORD    dwHighDateTime;
190 } FILETIME, *PFILETIME, *LPFILETIME;
191 #endif /* _FILETIME_ */
192
193 /* Find* structures */
194 typedef struct
195 {
196     DWORD    dwFileAttributes;
197     FILETIME ftCreationTime;
198     FILETIME ftLastAccessTime;

```

```

199     FILETIME    ftLastWriteTime;
200     DWORD       nFileSizeHigh;
201     DWORD       nFileSizeLow;
202     DWORD       dwReserved0;
203     DWORD       dwReserved1;
204     CHAR        cFileName[260];
205     CHAR        cAlternateFileName[14];
206 } WIN32_FIND_DATA, *PWIN32_FIND_DATA, *LPWIN32_FIND_DATA;
207
208 typedef struct
209 {
210     DWORD       dwFileAttributes;
211     FILETIME    ftCreationTime;
212     FILETIME    ftLastAccessTime;
213     FILETIME    ftLastWriteTime;
214     DWORD       nFileSizeHigh;
215     DWORD       nFileSizeLow;
216     DWORD       dwReserved0;
217     DWORD       dwReserved1;
218     WCHAR       cFileName[260];
219     WCHAR       cAlternateFileName[14];
220 } WIN32_FIND_DATAW, *PWIN32_FIND_DATAW, *LPWIN32_FIND_DATAW;
221
222 DECL_WINELIB_TYPE_AW(WIN32_FIND_DATA)
223 DECL_WINELIB_TYPE_AW(PWIN32_FIND_DATA)
224 DECL_WINELIB_TYPE_AW(LPWIN32_FIND_DATA)
225
226 typedef enum _FINDEX_INFO_LEVELS
227 {
228     FindExInfoStandard,
229     FindExInfoMaxInfoLevel
230 } FINDEX_INFO_LEVELS;
231
232 typedef enum _FINDEX_SEARCH_OPS
233 {
234     FindExSearchNameMatch,
235     FindExSearchLimitToDirectories,
236     FindExSearchLimitToDevices,
237     FindExSearchMaxSearchOp
238 } FINDEX_SEARCH_OPS;
239
240 typedef struct
241 {
242     LPVOID lpData;
243     DWORD  cbData;
244     BYTE   cbOverhead;
245     BYTE   iRegionIndex;
246     WORD   wFlags;
247     union u21 {
248         struct {
249             HANDLE hMem;
250             DWORD  dwReserved[3];
251         } Block;
252         struct {
253             DWORD  dwCommittedSize;
254             DWORD  dwUnCommittedSize;
255             LPVOID lpFirstBlock;
256             LPVOID lpLastBlock;
257         } Region;
258     } DUMMYUNIONNAME;
259 } PROCESS_HEAP_ENTRY, *PPROCESS_HEAP_ENTRY, *LPPROCESS_HEAP_ENTRY;
260
261 #define PROCESS_HEAP_REGION                0x0001
262 #define PROCESS_HEAP_UNCOMMITTED_RANGE    0x0002
263 #define PROCESS_HEAP_ENTRY_BUSY           0x0004
264 #define PROCESS_HEAP_ENTRY_MOVEABLE       0x0010
265 #define PROCESS_HEAP_ENTRY_DDESHARE       0x0020
266
267 #define INVALID_HANDLE_VALUE ((HANDLE) -1)
268
269 #define TLS_OUT_OF_INDEXES ((DWORD) 0xFFFFFFFF)
270
271 #define SHUTDOWN_NORETRY 1
272
273 /* comm */
274
275 #define CBR_110 0xFF10
276 #define CBR_300 0xFF11
277 #define CBR_600 0xFF12
278 #define CBR_1200 0xFF13
279 #define CBR_2400 0xFF14
280 #define CBR_4800 0xFF15
281 #define CBR_9600 0xFF16
282 #define CBR_14400 0xFF17
283 #define CBR_19200 0xFF18
284 #define CBR_38400 0xFF1B
285 #define CBR_56000 0xFF1F

```

```

286 #define CBR_57600      0xFF20
287 #define CBR_115200     0xFF21
288 #define CBR_128000     0xFF23
289 #define CBR_256000     0xFF27
290
291 #define NOPARITY        0
292 #define ODDPARITY       1
293 #define EVENPARITY      2
294 #define MARKPARITY      3
295 #define SPACEPARITY     4
296 #define ONESTOPBIT      0
297 #define ONE5STOPBITS    1
298 #define TWOSTOPBITS     2
299
300 #define IGNORE          0
301 #define INFINITE        0xFFFFFFFF
302
303 #define CE_RXOVER       0x0001
304 #define CE_OVERRUN      0x0002
305 #define CE_RXPARITY     0x0004
306 #define CE_FRAME        0x0008
307 #define CE_BREAK        0x0010
308 #define CE_CTSTO        0x0020
309 #define CE_DSRTO        0x0040
310 #define CE_RLSDTO       0x0080
311 #define CE_TXFULL       0x0100
312 #define CE_PTO          0x0200
313 #define CE_IOE          0x0400
314 #define CE_DNS          0x0800
315 #define CE_OOP          0x1000
316 #define CE_MODE 0x8000
317
318 #define IE_BADID        -1
319 #define IE_OPEN -2
320 #define IE_NOPEN -3
321 #define IE_MEMORY -4
322 #define IE_DEFAULT -5
323 #define IE_HARDWARE -10
324 #define IE_BYTESIZE -11
325 #define IE_BAUDRATE -12
326
327 #define EV_RXCHAR        0x0001
328 #define EV_RXFLAG        0x0002
329 #define EV_TXEMPTY       0x0004
330 #define EV_CTS           0x0008
331 #define EV_DSR           0x0010
332 #define EV_RLSD         0x0020
333 #define EV_BREAK         0x0040
334 #define EV_ERR           0x0080
335 #define EV_RING          0x0100
336 #define EV_PERR          0x0200
337 #define EV_RX80FULL      0x0400
338 #define EV_EVENT1        0x0800
339 #define EV_EVENT2        0x1000
340
341 #define SETXOFF 1
342 #define SETXON 2
343 #define SETRTS 3
344 #define CLRRTS 4
345 #define SETDTR 5
346 #define CLRDRTR 6
347 #define RESETDEV 7
348 #define SETBREAK 8
349 #define CLRBREAK 9
350
351 /* Purge functions for Comm Port */
352 #define PURGE_TXABORT    0x0001 /* Kill the pending/current writes to the
353 comm port */
354 #define PURGE_RXABORT    0x0002 /*Kill the pending/current reads to
355 the comm port */
356 #define PURGE_TXCLEAR    0x0004 /* Kill the transmit queue if there*/
357 #define PURGE_RXCLEAR    0x0008 /* Kill the typeahead buffer if there*/
358
359
360 /* Modem Status Flags */
361 #define MS_CTS_ON         ((DWORD)0x0010)
362 #define MS_DSR_ON         ((DWORD)0x0020)
363 #define MS_RING_ON       ((DWORD)0x0040)
364 #define MS_RLSD_ON        ((DWORD)0x0080)
365
366 #define RTS_CONTROL_DISABLE 0
367 #define RTS_CONTROL_ENABLE 1
368 #define RTS_CONTROL_HANDSHAKE 2
369 #define RTS_CONTROL_TOGGLE 3
370
371 #define DTR_CONTROL_DISABLE 0
372 #define DTR_CONTROL_ENABLE 1

```

```

373 #define DTR_CONTROL_HANDSHAKE    2
374
375
376 #define LMEM_FIXED                0
377 #define LMEM_MOVEABLE             0x0002
378 #define LMEM_NOCOMPACT            0x0010
379 #define LMEM_NODISCARD            0x0020
380 #define LMEM_ZEROINIT             0x0040
381 #define LMEM_MODIFY               0x0080
382 #define LMEM_DISCARDABLE          0x0F00
383 #define LMEM_DISCARDED            0x4000
384 #define LMEM_LOCKCOUNT          0x00FF
385
386 #define LPTR (LMEM_FIXED | LMEM_ZEROINIT)
387 #define LHND (LMEM_MOVEABLE | LMEM_ZEROINIT)
388
389 #define NONZEROLHND               (LMEM_MOVEABLE)
390 #define NONZEROLPTR              (LMEM_FIXED)
391
392 #define GMEM_FIXED                0x0000
393 #define GMEM_MOVEABLE             0x0002
394 #define GMEM_NOCOMPACT            0x0010
395 #define GMEM_NODISCARD            0x0020
396 #define GMEM_ZEROINIT             0x0040
397 #define GMEM_MODIFY               0x0080
398 #define GMEM_DISCARDABLE          0x0100
399 #define GMEM_NOT_BANKED           0x1000
400 #define GMEM_SHARE                0x2000
401 #define GMEM_DDESHARE             0x2000
402 #define GMEM_NOTIFY               0x4000
403 #define GMEM_LOWER                GMEM_NOT_BANKED
404 #define GMEM_DISCARDED            0x4000
405 #define GMEM_LOCKCOUNT          0x00ff
406 #define GMEM_INVALID_HANDLE       0x8000
407
408 #define GHND                      (GMEM_MOVEABLE | GMEM_ZEROINIT)
409 #define GPTR                      (GMEM_FIXED | GMEM_ZEROINIT)
410
411 #define INVALID_ATOM              ((ATOM) 0)
412 #define MAXINTATOM                0xc000
413 #define MAKEINTATOMA(atom)        ((LPCSTR) ((ULONG_PTR) ((WORD) (atom))))
414 #define MAKEINTATOMW(atom)        ((LPCWSTR) ((ULONG_PTR) ((WORD) (atom))))
415 #define MAKEINTATOM               WINELIB_NAME_AW(MAKEINTATOM)
416
417 typedef struct tagMEMORYSTATUS
418 {
419     DWORD    dwLength;
420     DWORD    dwMemoryLoad;
421     DWORD    dwTotalPhys;
422     DWORD    dwAvailPhys;
423     DWORD    dwTotalPageFile;
424     DWORD    dwAvailPageFile;
425     DWORD    dwTotalVirtual;
426     DWORD    dwAvailVirtual;
427 } MEMORYSTATUS, *LPMEMORYSTATUS;
428
429
430 typedef struct {
431     WORD    wYear;
432     WORD    wMonth;
433     WORD    wDayOfWeek;
434     WORD    wDay;
435     WORD    wHour;
436     WORD    wMinute;
437     WORD    wSecond;
438     WORD    wMilliseconds;
439 } SYSTEMTIME, *PSYSTEMTIME, *LPSYSTEMTIME;
440
441 /* The 'overlapped' data structure used by async I/O functions.
442 */
443 typedef struct {
444     DWORD    Internal;
445     DWORD    InternalHigh;
446     DWORD    Offset;
447     DWORD    OffsetHigh;
448     HANDLE    hEvent;
449 } OVERLAPPED, *LPOVERLAPPED;
450
451 typedef VOID CALLBACK (*LPOVERLAPPED_COMPLETION_ROUTINE) (DWORD dwErrorCode, DWORD
    dwNumberOfBytesTransferred, LPOVERLAPPED lpOverlapped);
452
453 /* Process startup information.
454 */
455
456 /* STARTUPINFO.dwFlags */
457 #define STARTF_USESHOWWINDOW       0x00000001
458 #define STARTF_USESIZE              0x00000002

```

```

459 #define STARTF_USEPOSITION 0x00000004
460 #define STARTF_USECOUNTCHARS 0x00000008
461 #define STARTF_USEFILLATTRIBUTE 0x00000010
462 #define STARTF_RUNFULLSCREEN 0x00000020
463 #define STARTF_FORCEONFEEDBACK 0x00000040
464 #define STARTF_FORCEOFFFEEDBACK 0x00000080
465 #define STARTF_USESTDHANDLES 0x00000100
466 #define STARTF_USEHOTKEY 0x00000200
467
468 typedef struct {
469     DWORD cb; /* 00: size of struct */
470     LPSTR lpReserved; /* 04: */
471     LPSTR lpDesktop; /* 08: */
472     LPSTR lpTitle; /* 0c: */
473     DWORD dwX; /* 10: */
474     DWORD dwY; /* 14: */
475     DWORD dwXSize; /* 18: */
476     DWORD dwYSize; /* 1c: */
477     DWORD dwXCountChars; /* 20: */
478     DWORD dwYCountChars; /* 24: */
479     DWORD dwFillAttribute; /* 28: */
480     DWORD dwFlags; /* 2c: */
481     WORD wShowWindow; /* 30: */
482     WORD cbReserved2; /* 32: */
483     BYTE *lpReserved2; /* 34: */
484     HANDLE hStdInput; /* 38: */
485     HANDLE hStdOutput; /* 3c: */
486     HANDLE hStdError; /* 40: */
487 } STARTUPINFOA, *LPSTARTUPINFOA;
488
489 typedef struct {
490     DWORD cb;
491     LPWSTR lpReserved;
492     LPWSTR lpDesktop;
493     LPWSTR lpTitle;
494     DWORD dwX;
495     DWORD dwY;
496     DWORD dwXSize;
497     DWORD dwYSize;
498     DWORD dwXCountChars;
499     DWORD dwYCountChars;
500     DWORD dwFillAttribute;
501     DWORD dwFlags;
502     WORD wShowWindow;
503     WORD cbReserved2;
504     BYTE *lpReserved2;
505     HANDLE hStdInput;
506     HANDLE hStdOutput;
507     HANDLE hStdError;
508 } STARTUPINFOW, *LPSTARTUPINFOW;
509
510 DECL_WINELIB_TYPE_AW(STARTUPINFO)
511 DECL_WINELIB_TYPE_AW(LPSTARTUPINFO)
512
513 typedef struct {
514     HANDLE hProcess;
515     HANDLE hThread;
516     DWORD dwProcessId;
517     DWORD dwThreadId;
518 } PROCESS_INFORMATION, *PPROCESS_INFORMATION, *LPPROCESS_INFORMATION;
519
520 typedef struct {
521     LONG Bias;
522     WCHAR StandardName[32];
523     SYSTEMTIME StandardDate;
524     LONG StandardBias;
525     WCHAR DaylightName[32];
526     SYSTEMTIME DaylightDate;
527     LONG DaylightBias;
528 } TIME_ZONE_INFORMATION, *PTIME_ZONE_INFORMATION, *LPTIME_ZONE_INFORMATION;
529
530 #define TIME_ZONE_ID_INVALID ((DWORD)0xFFFFFFFF)
531 #define TIME_ZONE_ID_UNKNOWN 0
532 #define TIME_ZONE_ID_STANDARD 1
533 #define TIME_ZONE_ID_DAYLIGHT 2
534
535 /* CreateProcess: dwCreationFlag values
536 */
537 #define DEBUG_PROCESS 0x00000001
538 #define DEBUG_ONLY_THIS_PROCESS 0x00000002
539 #define CREATE_SUSPENDED 0x00000004
540 #define DETACHED_PROCESS 0x00000008
541 #define CREATE_NEW_CONSOLE 0x00000010
542 #define NORMAL_PRIORITY_CLASS 0x00000020
543 #define IDLE_PRIORITY_CLASS 0x00000040
544 #define HIGH_PRIORITY_CLASS 0x00000080
545 #define REALTIME_PRIORITY_CLASS 0x00000100

```



```

546 #define CREATE_NEW_PROCESS_GROUP    0x00000200
547 #define CREATE_UNICODE_ENVIRONMENT 0x00000400
548 #define CREATE_SEPARATE_WOW_VDM     0x00000800
549 #define CREATE_SHARED_WOW_VDM       0x00001000
550 #define CREATE_DEFAULT_ERROR_MODE    0x04000000
551 #define CREATE_NO_WINDOW             0x08000000
552 #define PROFILE_USER                 0x10000000
553 #define PROFILE_KERNEL               0x20000000
554 #define PROFILE_SERVER               0x40000000
555
556
557 /* File object type definitions
558 */
559 #define FILE_TYPE_UNKNOWN            0
560 #define FILE_TYPE_DISK              1
561 #define FILE_TYPE_CHAR               2
562 #define FILE_TYPE_PIPE              3
563 #define FILE_TYPE_REMOTE            32768
564
565 /* File creation flags
566 */
567 #define FILE_FLAG_WRITE_THROUGH      0x80000000UL
568 #define FILE_FLAG_OVERLAPPED         0x40000000L
569 #define FILE_FLAG_NO_BUFFERING       0x20000000L
570 #define FILE_FLAG_RANDOM_ACCESS     0x10000000L
571 #define FILE_FLAG_SEQUENTIAL_SCAN   0x08000000L
572 #define FILE_FLAG_DELETE_ON_CLOSE    0x04000000L
573 #define FILE_FLAG_BACKUP_SEMANTICS  0x02000000L
574 #define FILE_FLAG_POSIX_SEMANTICS    0x01000000L
575 #define CREATE_NEW                   1
576 #define CREATE_ALWAYS                2
577 #define OPEN_EXISTING                3
578 #define OPEN_ALWAYS                  4
579 #define TRUNCATE_EXISTING            5
580
581 /* Standard handle identifiers
582 */
583 #define STD_INPUT_HANDLE              ((DWORD) -10)
584 #define STD_OUTPUT_HANDLE             ((DWORD) -11)
585 #define STD_ERROR_HANDLE              ((DWORD) -12)
586
587 typedef struct
588 {
589     DWORD dwFileAttributes;
590     FILETIME ftCreationTime;
591     FILETIME ftLastAccessTime;
592     FILETIME ftLastWriteTime;
593     DWORD dwVolumeSerialNumber;
594     DWORD nFileSizeHigh;
595     DWORD nFileSizeLow;
596     DWORD nNumberOfLinks;
597     DWORD nFileIndexHigh;
598     DWORD nFileIndexLow;
599 } BY_HANDLE_FILE_INFORMATION, *PBY_HANDLE_FILE_INFORMATION, *LPBY_HANDLE_FILE_INFORMATION ;
600
601 #define PIPE_ACCESS_INBOUND 1
602 #define PIPE_ACCESS_OUTBOUND 2
603 #define PIPE_ACCESS_DUPLEX 3
604
605 #define PIPE_TYPE_BYTE 0
606 #define PIPE_TYPE_MESSAGE 4
607
608 #define PIPE_READMODE_BYTE 0
609 #define PIPE_READMODE_MESSAGE 2
610
611 #define PIPE_WAIT 0
612 #define PIPE_NOWAIT 1
613
614 #define PIPE_UNLIMITED_INSTANCES 255
615
616 #define NMPWAIT_WAIT_FOREVER 0xffffffff
617 #define NMPWAIT_NOWAIT 0x00000001
618 #define NMPWAIT_USE_DEFAULT_WAIT 0x00000000
619
620 typedef struct _SYSTEM_POWER_STATUS
621 {
622     BYTE    ACLineStatus;
623     BYTE    BatteryFlag;
624     BYTE    BatteryLifePercent;
625     BYTE    reserved;
626     DWORD    BatteryLifeTime;
627     DWORD    BatteryFullLifeTime;
628 } SYSTEM_POWER_STATUS, *LPSYSTEM_POWER_STATUS;
629
630
631 typedef struct tagSYSTEM_INFO
632 {

```

```

633     union u3 {
634         DWORD    dwOemId; /* Obsolete field - do not use */
635     struct splits {
636         WORD    wProcessorArchitecture;
637         WORD    wReserved;
638     } DUMMYSTRUCTNAME;
639     } DUMMYUNIONNAME;
640     DWORD    dwPageSize;
641     LPVOID    lpMinimumApplicationAddress;
642     LPVOID    lpMaximumApplicationAddress;
643     DWORD    dwActiveProcessorMask;
644     DWORD    dwNumberOfProcessors;
645     DWORD    dwProcessorType;
646     DWORD    dwAllocationGranularity;
647     WORD    wProcessorLevel;
648     WORD    wProcessorRevision;
649 } SYSTEM_INFO, *LPSYSTEM_INFO;
650
651 typedef BOOL CALLBACK (*ENUMRESTYPEPROCA) (HMODULE, LPSTR, LONG);
652 typedef BOOL CALLBACK (*ENUMRESTYPEPROCW) (HMODULE, LPWSTR, LONG);
653 typedef BOOL CALLBACK (*ENUMRESNAMEPROCA) (HMODULE, LPCSTR, LPSTR, LONG);
654 typedef BOOL CALLBACK (*ENUMRESNAMEPROCW) (HMODULE, LPCWSTR, LPWSTR, LONG);
655 typedef BOOL CALLBACK (*ENUMRESLANGPROCA) (HMODULE, LPCSTR, LPCSTR, WORD, LONG);
656 typedef BOOL CALLBACK (*ENUMRESLANGPROCW) (HMODULE, LPCWSTR, LPCWSTR, WORD, LONG);
657
658 DECL_WINELIB_TYPE_AW (ENUMRESTYPEPROC)
659 DECL_WINELIB_TYPE_AW (ENUMRESNAMEPROC)
660 DECL_WINELIB_TYPE_AW (ENUMRESLANGPROC)
661
662 /* flags that can be passed to LoadLibraryEx */
663 #define DONT_RESOLVE_DLL_REFERENCES 0x00000001
664 #define LOAD_LIBRARY_AS_DATAFILE 0x00000002
665 #define LOAD_WITH_ALTERED_SEARCH_PATH 0x00000008
666
667 /* ifdef _x86_ ... */
668 typedef struct _LDT_ENTRY {
669     WORD    LimitLow;
670     WORD    BaseLow;
671     union u4 {
672     struct {
673         BYTE    BaseMid;
674         BYTE    Flags1; /* Declare as bytes to avoid alignment problems */
675         BYTE    Flags2;
676         BYTE    BaseHi;
677     } Bytes;
678     struct {
679         unsigned    BaseMid    : 8;
680         unsigned    Type      : 5;
681         unsigned    Dpl       : 2;
682         unsigned    Pres      : 1;
683         unsigned    LimitHi   : 4;
684         unsigned    Sys       : 1;
685         unsigned    Reserved_0 : 1;
686         unsigned    Default_Big : 1;
687         unsigned    Granularity : 1;
688         unsigned    BaseHi    : 8;
689     } Bits;
690     } HighWord;
691 } LDT_ENTRY, *LPLDT_ENTRY;
692
693
694 typedef enum _GET_FILEEX_INFO_LEVELS {
695     GetFileExInfoStandard
696 } GET_FILEEX_INFO_LEVELS;
697
698 typedef struct _WIN32_FILE_ATTRIBUTES_DATA {
699     DWORD    dwFileAttributes;
700     FILETIME ftCreationTime;
701     FILETIME ftLastAccessTime;
702     FILETIME ftLastWriteTime;
703     DWORD    nFileSizeHigh;
704     DWORD    nFileSizeLow;
705 } WIN32_FILE_ATTRIBUTE_DATA, *LPWIN32_FILE_ATTRIBUTE_DATA;
706
707 /*
708 * This one seems to be a Win32 only definition. It also is defined with
709 * WINAPI instead of CALLBACK in the windows headers.
710 */
711 typedef DWORD CALLBACK (*LPPROGRESS_ROUTINE) (LARGE_INTEGER, LARGE_INTEGER, LARGE_INTEGER,
712                                             LARGE_INTEGER, DWORD, DWORD, HANDLE,
713                                             HANDLE, LPVOID);
714
715
716 #define WAIT_FAILED 0xffffffff
717 #define WAIT_OBJECT_0 0
718 #define WAIT_ABANDONED STATUS_ABANDONED_WAIT_0
719 #define WAIT_ABANDONED_0 STATUS_ABANDONED_WAIT_0

```

```

720 #define WAIT_IO_COMPLETION    STATUS_USER_APC
721 #define WAIT_TIMEOUT           STATUS_TIMEOUT
722 #define STILL_ACTIVE           STATUS_PENDING
723
724 #define FILE_BEGIN              0
725 #define FILE_CURRENT            1
726 #define FILE_END                2
727
728 #define FILE_MAP_COPY           0x00000001
729 #define FILE_MAP_WRITE          0x00000002
730 #define FILE_MAP_READ           0x00000004
731 #define FILE_MAP_ALL_ACCESS     0x000f001f
732
733 #define MOVEFILE_REPLACE_EXISTING 0x00000001
734 #define MOVEFILE_COPY_ALLOWED    0x00000002
735 #define MOVEFILE_DELAY_UNTIL_REBOOT 0x00000004
736
737 #define FS_CASE_SENSITIVE        FILE_CASE_SENSITIVE_SEARCH
738 #define FS_CASE_IS_PRESERVED     FILE_CASE_PRESERVED_NAMES
739 #define FS_UNICODE_STORED_ON_DISK FILE_UNICODE_ON_DISK
740 #define FS_PERSISTENT_ACLS       FILE_PERSISTENT_ACLS
741 #define FS_VOL_IS_COMPRESSED     FILE_VOLUME_IS_COMPRESSED
742 #define FS_FILE_COMPRESSION      FILE_FILE_COMPRESSION
743
744 #define EXCEPTION_ACCESS_VIOLATION    STATUS_ACCESS_VIOLATION
745 #define EXCEPTION_DATATYPE_MISALIGNMENT STATUS_DATATYPE_MISALIGNMENT
746 #define EXCEPTION_BREAKPOINT          STATUS_BREAKPOINT
747 #define EXCEPTION_SINGLE_STEP          STATUS_SINGLE_STEP
748 #define EXCEPTION_ARRAY_BOUNDS_EXCEEDED STATUS_ARRAY_BOUNDS_EXCEEDED
749 #define EXCEPTION_FLT_DENORMAL_OPERAND STATUS_FLOAT_DENORMAL_OPERAND
750 #define EXCEPTION_FLT_DIVIDE_BY_ZERO   STATUS_FLOAT_DIVIDE_BY_ZERO
751 #define EXCEPTION_FLT_INEXACT_RESULT   STATUS_FLOAT_INEXACT_RESULT
752 #define EXCEPTION_FLT_INVALID_OPERATION STATUS_FLOAT_INVALID_OPERATION
753 #define EXCEPTION_FLT_OVERFLOW         STATUS_FLOAT_OVERFLOW
754 #define EXCEPTION_FLT_STACK_CHECK      STATUS_FLOAT_STACK_CHECK
755 #define EXCEPTION_FLT_UNDERFLOW        STATUS_FLOAT_UNDERFLOW
756 #define EXCEPTION_INT_DIVIDE_BY_ZERO   STATUS_INTEGER_DIVIDE_BY_ZERO
757 #define EXCEPTION_INT_OVERFLOW         STATUS_INTEGER_OVERFLOW
758 #define EXCEPTION_PRIV_INSTRUCTION     STATUS_PRIVILEGED_INSTRUCTION
759 #define EXCEPTION_IN_PAGE_ERROR        STATUS_IN_PAGE_ERROR
760 #define EXCEPTION_ILLEGAL_INSTRUCTION  STATUS_ILLEGAL_INSTRUCTION
761 #define EXCEPTION_NONCONTINUABLE_EXCEPTION STATUS_NONCONTINUABLE_EXCEPTION
762 #define EXCEPTION_STACK_OVERFLOW        STATUS_STACK_OVERFLOW
763 #define EXCEPTION_INVALID_DISPOSITION  STATUS_INVALID_DISPOSITION
764 #define EXCEPTION_GUARD_PAGE           STATUS_GUARD_PAGE_VIOLATION
765 #define EXCEPTION_INVALID_HANDLE       STATUS_INVALID_HANDLE
766 #define CONTROL_C_EXIT                 STATUS_CONTROL_C_EXIT
767
768 /* Wine extension; Windows doesn't have a name for this code */
769 #define EXCEPTION_CRITICAL_SECTION_WAIT 0xc0000194
770
771 #define DUPLICATE_CLOSE_SOURCE 0x00000001
772 #define DUPLICATE_SAME_ACCESS  0x00000002
773
774 #define HANDLE_FLAG_INHERIT 0x00000001
775 #define HANDLE_FLAG_PROTECT_FROM_CLOSE 0x00000002
776
777 #define HINSTANCE_ERROR 32
778
779 #define THREAD_PRIORITY_LOWEST    THREAD_BASE_PRIORITY_MIN
780 #define THREAD_PRIORITY_BELOW_NORMAL (THREAD_PRIORITY_LOWEST+1)
781 #define THREAD_PRIORITY_NORMAL    0
782 #define THREAD_PRIORITY_HIGHEST    THREAD_BASE_PRIORITY_MAX
783 #define THREAD_PRIORITY_ABOVE_NORMAL (THREAD_PRIORITY_HIGHEST-1)
784 #define THREAD_PRIORITY_ERROR_RETURN (0x7fffffff)
785 #define THREAD_PRIORITY_TIME_CRITICAL THREAD_BASE_PRIORITY_LOWRT
786 #define THREAD_PRIORITY_IDLE        THREAD_BASE_PRIORITY_IDLE
787
788 /* flags to FormatMessage */
789 #define FORMAT_MESSAGE_ALLOCATE_BUFFER 0x00000100
790 #define FORMAT_MESSAGE_IGNORE_INSERTS 0x00000200
791 #define FORMAT_MESSAGE_FROM_STRING    0x00000400
792 #define FORMAT_MESSAGE_FROM_HMODULE   0x00000800
793 #define FORMAT_MESSAGE_FROM_SYSTEM    0x00001000
794 #define FORMAT_MESSAGE_ARGUMENT_ARRAY 0x00002000
795 #define FORMAT_MESSAGE_MAX_WIDTH_MASK 0x000000ff
796
797 #ifdef __WINE__
798 #define CRITICAL_SECTION_INIT(name) { (void *) (__FILE__ " : " name), -1, 0, 0, 0, 0 }
799 #endif
800
801 typedef struct {
802     DWORD dwOSVersionInfoSize;
803     DWORD dwMajorVersion;
804     DWORD dwMinorVersion;
805     DWORD dwBuildNumber;
806     DWORD dwPlatformId;

```

```

807     CHAR szCSDVersion[128];
808 } OSVERSIONINFOA, *POSVERSIONINFOA, *LPOSVERSIONINFOA;
809
810 typedef struct {
811     DWORD dwOSVersionInfoSize;
812     DWORD dwMajorVersion;
813     DWORD dwMinorVersion;
814     DWORD dwBuildNumber;
815     DWORD dwPlatformId;
816     WCHAR szCSDVersion[128];
817 } OSVERSIONINFOW, *POSVERSIONINFOW, *LPOSVERSIONINFOW;
818
819 DECL_WINELIB_TYPE_AW(OSVERSIONINFO)
820 DECL_WINELIB_TYPE_AW(POSVERSIONINFO)
821 DECL_WINELIB_TYPE_AW(LPOSVERSIONINFO)
822
823 #define VER_PLATFORM_WIN32s 0
824 #define VER_PLATFORM_WIN32_WINDOWS 1
825 #define VER_PLATFORM_WIN32_NT 2
826
827 typedef struct tagCOMSTAT
828 {
829     DWORD status;
830     DWORD cbInQue;
831     DWORD cbOutQue;
832 } COMSTAT, *LPCOMSTAT;
833
834 typedef struct tagDCB
835 {
836     DWORD DCBlength;
837     DWORD BaudRate;
838     unsigned fBinary :1;
839     unsigned fParity :1;
840     unsigned fOutxCtsFlow :1;
841     unsigned fOutxDsrFlow :1;
842     unsigned fDtrControl :2;
843     unsigned fDsrSensitivity :1;
844     unsigned fTXContinueOnXoff :1;
845     unsigned fOutX :1;
846     unsigned fInX :1;
847     unsigned fErrorChar :1;
848     unsigned fNull :1;
849     unsigned fRtsControl :2;
850     unsigned fAbortOnError :1;
851     unsigned fDummy2 :17;
852     WORD wReserved;
853     WORD XonLim;
854     WORD XoffLim;
855     BYTE ByteSize;
856     BYTE Parity;
857     BYTE StopBits;
858     char XonChar;
859     char XoffChar;
860     char ErrorChar;
861     char EofChar;
862     char EvtChar;
863 } DCB, *LPDCB;
864
865 typedef struct tagCOMMCONFIG {
866     DWORD dwSize;
867     WORD wVersion;
868     WORD wReserved;
869     DCB dcb;
870     DWORD dwProviderSubType;
871     DWORD dwProviderOffset;
872     DWORD dwProviderSize;
873     DWORD wcProviderData[1];
874 } COMMCONFIG, *LPCOMMCONFIG;
875
876 typedef struct tagCOMMPROP {
877     WORD wPacketLength;
878     WORD wPacketVersion;
879     DWORD dwServiceMask;
880     DWORD dwReserved1;
881     DWORD dwMaxTxQueue;
882     DWORD dwMaxRxQueue;
883     DWORD dwMaxBaud;
884     DWORD dwProvSubType;
885     DWORD dwProvCapabilities;
886     DWORD dwSettableParams;
887     DWORD dwSettableBaud;
888     WORD wSettableData;
889     WORD wSettableStopParity;
890     DWORD dwCurrentTxQueue;
891     DWORD dwCurrentRxQueue;
892     DWORD dwProvSpec1;
893     DWORD dwProvSpec2;

```

```
894     WCHAR wcProvChar[1];
895 } COMMPROP, *LPCOMMPROP;
896
897 #define SP_SERIALCOMM ((DWORD)1)
898
899 #define BAUD_075      ((DWORD)0x01)
900 #define BAUD_110      ((DWORD)0x02)
901 #define BAUD_134_5    ((DWORD)0x04)
902 #define BAUD_150      ((DWORD)0x08)
903 #define BAUD_300      ((DWORD)0x10)
904 #define BAUD_600      ((DWORD)0x20)
905 #define BAUD_1200     ((DWORD)0x40)
906 #define BAUD_1800     ((DWORD)0x80)
907 #define BAUD_2400     ((DWORD)0x100)
908 #define BAUD_4800     ((DWORD)0x200)
909 #define BAUD_7200     ((DWORD)0x400)
910 #define BAUD_9600     ((DWORD)0x800)
911 #define BAUD_14400    ((DWORD)0x1000)
912 #define BAUD_19200    ((DWORD)0x2000)
913 #define BAUD_38400    ((DWORD)0x4000)
914 #define BAUD_56K      ((DWORD)0x8000)
915 #define BAUD_57600    ((DWORD)0x40000)
916 #define BAUD_115200   ((DWORD)0x200000)
917 #define BAUD_128K     ((DWORD)0x100000)
918 #define BAUD_USER     ((DWORD)0x10000000)
919
920 #define PST_FAX        ((DWORD)0x21)
921 #define PST_LAT        ((DWORD)0x101)
922 #define PST_MODEM      ((DWORD)0x06)
923 #define PST_NETWORK_BRIDGE ((DWORD)0x100)
924 #define PST_PARALLEL_PORT ((DWORD)0x02)
925 #define PST_RS232      ((DWORD)0x01)
926 #define PST_RS442      ((DWORD)0x03)
927 #define PST_RS423      ((DWORD)0x04)
928 #define PST_RS449      ((DWORD)0x06)
929 #define PST_SCANNER    ((DWORD)0x22)
930 #define PST_TCPIP_TELNET ((DWORD)0x102)
931 #define PST_UNSPECIFIED ((DWORD)0x00)
932 #define PST_X25        ((DWORD)0x103)
933
934 #define PCF_16BITMODE  ((DWORD)0x200)
935 #define PCF_DTRDSR     ((DWORD)0x01)
936 #define PCF_INTTIMEOUTS ((DWORD)0x80)
937 #define PCF_PARITY_CHECK ((DWORD)0x08)
938 #define PCF_RLSD       ((DWORD)0x04)
939 #define PCF_RTSCTS     ((DWORD)0x02)
940 #define PCF_SETXCHAR    ((DWORD)0x20)
941 #define PCF_SPECIALCHARS ((DWORD)0x100)
942 #define PCF_TOTALTIMEOUTS ((DWORD)0x40)
943 #define PCF_XONXOFF    ((DWORD)0x10)
944
945 #define SP_BAUD        ((DWORD)0x02)
946 #define SP_DATABITS    ((DWORD)0x04)
947 #define SP_HANDSHAKING ((DWORD)0x10)
948 #define SP_PARITY      ((DWORD)0x01)
949 #define SP_PARITY_CHECK ((DWORD)0x20)
950 #define SP_RLSD        ((DWORD)0x40)
951 #define SP_STOPBITS    ((DWORD)0x08)
952
953 #define DATABITS_5     ((DWORD)0x01)
954 #define DATABITS_6     ((DWORD)0x02)
955 #define DATABITS_7     ((DWORD)0x04)
956 #define DATABITS_8     ((DWORD)0x08)
957 #define DATABITS_16    ((DWORD)0x10)
958 #define DATABITS_16X   ((DWORD)0x20)
959
960 #define STOPBITS_10 ((DWORD)1)
961 #define STOPBITS_15 ((DWORD)2)
962 #define STOPBITS_20 ((DWORD)4)
963
964 #define PARITY_NONE    ((DWORD)0x100)
965 #define PARITY_ODD     ((DWORD)0x200)
966 #define PARITY_EVEN    ((DWORD)0x400)
967 #define PARITY_MARK    ((DWORD)0x800)
968 #define PARITY_SPACE   ((DWORD)0x1000)
969
970 typedef struct tagCOMMTIMEOUTS {
971     DWORD    ReadIntervalTimeout;
972     DWORD    ReadTotalTimeoutMultiplier;
973     DWORD    ReadTotalTimeoutConstant;
974     DWORD    WriteTotalTimeoutMultiplier;
975     DWORD    WriteTotalTimeoutConstant;
976 } COMMTIMEOUTS, *LPCOMMTIMEOUTS;
977
978 typedef void CALLBACK (*PAPCFUNC) (ULONG_PTR);
979 typedef void CALLBACK (*PTIMERAPCROUTINE) (LPVOID, DWORD, DWORD);
980
```

```

981 /*DWORD WINAPI GetVersion( void );*/
982 BOOL WINAPI GetVersionExA(OSVERSIONINFOA*);
983 BOOL WINAPI GetVersionExW(OSVERSIONINFOW*);
984 #define GetVersionEx WINELIB_NAME_AW(GetVersionEx)
985
986 /*int WinMain(HINSTANCE, HINSTANCE prev, char *cmd, int show);*/
987
988 /* FIXME: need to use defines because we don't have proper imports everywhere yet */
989 #ifndef have_proper_imports
990 LONG WINAPI RtlEnterCriticalSection( CRITICAL_SECTION *crit );
991 LONG WINAPI RtlLeaveCriticalSection( CRITICAL_SECTION *crit );
992 LONG WINAPI RtlDeleteCriticalSection( CRITICAL_SECTION *crit );
993 BOOL WINAPI RtlTryEnterCriticalSection( CRITICAL_SECTION *crit );
994 PVOID WINAPI RtlAllocateHeap( HANDLE, ULONG, ULONG );
995 BOOLEAN WINAPI RtlFreeHeap( HANDLE, ULONG, PVOID );
996 PVOID WINAPI RtlReAllocateHeap( HANDLE, ULONG, PVOID, ULONG );
997 ULONG WINAPI RtlSizeHeap( HANDLE, ULONG, PVOID );
998 #define HeapAlloc(heap, flags, size) RtlAllocateHeap(heap, flags, size)
999 #define HeapFree(heap, flags, ptr) RtlFreeHeap(heap, flags, ptr)
1000 #define HeapReAlloc(heap, flags, ptr, size) RtlReAllocateHeap(heap, flags, ptr, size)
1001 #define HeapSize(heap, flags, ptr) RtlSizeHeap(heap, flags, ptr)
1002 #define EnterCriticalSection(crit) RtlEnterCriticalSection(crit)
1003 #define LeaveCriticalSection(crit) RtlLeaveCriticalSection(crit)
1004 #define DeleteCriticalSection(crit) RtlDeleteCriticalSection(crit)
1005 #define TryEnterCriticalSection(crit) RtlTryEnterCriticalSection(crit)
1006 #else
1007 LPVOID WINAPI HeapAlloc( HANDLE, DWORD, DWORD );
1008 BOOL WINAPI HeapFree( HANDLE, DWORD, LPVOID );
1009 LPVOID WINAPI HeapReAlloc( HANDLE, DWORD, LPVOID, DWORD );
1010 DWORD WINAPI HeapSize( HANDLE, DWORD, LPVOID );
1011 void WINAPI DeleteCriticalSection( CRITICAL_SECTION *lpCrit );
1012 void WINAPI EnterCriticalSection( CRITICAL_SECTION *lpCrit );
1013 BOOL WINAPI TryEnterCriticalSection( CRITICAL_SECTION *lpCrit );
1014 void WINAPI LeaveCriticalSection( CRITICAL_SECTION *lpCrit );
1015 #endif
1016
1017 void WINAPI InitializeCriticalSection( CRITICAL_SECTION *lpCrit );
1018 BOOL WINAPI InitializeCriticalSectionAndSpinCount( CRITICAL_SECTION *, DWORD );
1019 void WINAPI MakeCriticalSectionGlobal( CRITICAL_SECTION *lpCrit );
1020 BOOL WINAPI GetProcessWorkingSetSize( HANDLE, LPDWORD, LPDWORD );
1021 DWORD WINAPI QueueUserAPC( PAPCFUNC, HANDLE, ULONG_PTR );
1022 void WINAPI RaiseException( DWORD, DWORD, DWORD, const LPDWORD );
1023 BOOL WINAPI SetProcessWorkingSetSize( HANDLE, DWORD, DWORD );
1024 BOOL WINAPI TerminateProcess( HANDLE, DWORD );
1025 BOOL WINAPI TerminateThread( HANDLE, DWORD );
1026 BOOL WINAPI GetExitCodeThread( HANDLE, LPDWORD );
1027
1028 /* GetBinaryType return values.
1029 */
1030
1031 #define SCS_32BIT_BINARY 0
1032 #define SCS_DOS_BINARY 1
1033 #define SCS_WOW_BINARY 2
1034 #define SCS_PIF_BINARY 3
1035 #define SCS_POSIX_BINARY 4
1036 #define SCS_OS216_BINARY 5
1037
1038 BOOL WINAPI GetBinaryTypeA( LPCSTR lpApplicationName, LPDWORD lpBinaryType );
1039 BOOL WINAPI GetBinaryTypeW( LPCWSTR lpApplicationName, LPDWORD lpBinaryType );
1040 #define GetBinaryType WINELIB_NAME_AW(GetBinaryType)
1041
1042 /* Declarations for functions that exist only in Win32 */
1043
1044 BOOL WINAPI AddAccessAllowedAce( PACL, DWORD, DWORD, PSID );
1045 BOOL WINAPI AttachThreadInput( DWORD, DWORD, BOOL );
1046 BOOL WINAPI
    AccessCheck( PSECURITY_DESCRIPTOR, HANDLE, DWORD, PGENERIC_MAPPING, PPRIVILEGE_SET, LPDWORD, LPDWORD, LPBOOL );
1047 BOOL WINAPI AdjustTokenPrivileges( HANDLE, BOOL, LPVOID, DWORD, LPVOID, LPDWORD );
1048 BOOL WINAPI
    AllocateAndInitializeSid( PSID_IDENTIFIER_AUTHORITY, BYTE, DWORD, DWORD, DWORD, DWORD, DWORD, DWORD, DWORD, PSID
    * );
1049 BOOL WINAPI AllocateLocallyUniqueId( PLUID );
1050 BOOL WINAPI AreFileApisANSI( void );
1051 BOOL WINAPI BackupEventLogA( HANDLE, LPCSTR );
1052 BOOL WINAPI BackupEventLogW( HANDLE, LPCWSTR );
1053 #define BackupEventLog WINELIB_NAME_AW(BackupEventLog)
1054 BOOL WINAPI BackupRead( HANDLE, LPBYTE, DWORD, LPDWORD, BOOL, BOOL, LPVOID* );
1055 BOOL WINAPI BackupSeek( HANDLE, DWORD, DWORD, LPDWORD, LPDWORD, LPVOID* );
1056 BOOL WINAPI BackupWrite( HANDLE, LPBYTE, DWORD, LPDWORD, BOOL, BOOL, LPVOID* );
1057 BOOL WINAPI Beep( DWORD, DWORD );
1058 BOOL WINAPI BuildCommDCBA( LPCSTR, LPDCB );
1059 BOOL WINAPI BuildCommDCBW( LPCWSTR, LPDCB );
1060 #define BuildCommDCB WINELIB_NAME_AW(BuildCommDCB)
1061 BOOL WINAPI BuildCommDCBAndTimeoutsA( LPCSTR, LPDCB, LPCOMMTIMEOUTS );
1062 BOOL WINAPI BuildCommDCBAndTimeoutsW( LPCWSTR, LPDCB, LPCOMMTIMEOUTS );
1063 #define BuildCommDCBAndTimeouts WINELIB_NAME_AW(BuildCommDCBAndTimeouts)
1064 BOOL WINAPI CancelIo( HANDLE );

```

```

1065 BOOL        WINAPI CancelWaitableTimer (HANDLE);
1066 BOOL        WINAPI ClearCommBreak (HANDLE);
1067 BOOL        WINAPI ClearCommError (HANDLE, LPDWORD, LPCOMSTAT);
1068 BOOL        WINAPI ClearEventLogA (HANDLE, LPCSTR);
1069 BOOL        WINAPI ClearEventLogW (HANDLE, LPCWSTR);
1070 #define      ClearEventLog WINELIB_NAME_AW (ClearEventLog)
1071 BOOL        WINAPI CloseEventLog (HANDLE);
1072 BOOL        WINAPI CloseHandle (HANDLE);
1073 BOOL        WINAPI CommConfigDialogA (LPCSTR, HANDLE, LPCOMMCONFIG);
1074 BOOL        WINAPI CommConfigDialogW (LPCWSTR, HANDLE, LPCOMMCONFIG);
1075 #define      CommConfigDialog WINELIB_NAME_AW (CommConfigDialog)
1076 BOOL        WINAPI ConnectNamedPipe (HANDLE, LPOVERLAPPED);
1077 BOOL        WINAPI ContinueDebugEvent (DWORD, DWORD, DWORD);
1078 HANDLE      WINAPI ConvertToGlobalHandle (HANDLE hSrc);
1079 BOOL        WINAPI CopyFileA (LPCSTR, LPCSTR, BOOL);
1080 BOOL        WINAPI CopyFileW (LPCWSTR, LPCWSTR, BOOL);
1081 #define      CopyFile WINELIB_NAME_AW (CopyFile)
1082 BOOL        WINAPI CopyFileExA (LPCSTR, LPCSTR, LPPROGRESS_ROUTINE, LPVOID, LPBOOL, DWORD);
1083 BOOL        WINAPI CopyFileExW (LPCWSTR, LPCWSTR, LPPROGRESS_ROUTINE, LPVOID, LPBOOL, DWORD);
1084 #define      CopyFileEx WINELIB_NAME_AW (CopyFileEx)
1085 BOOL        WINAPI CopySid (DWORD, PSID, PSID);
1086 INT         WINAPI CompareFileTime (LPFILETIME, LPFILETIME);
1087 HANDLE      WINAPI CreateEventA (LPSECURITY_ATTRIBUTES, BOOL, BOOL, LPCSTR);
1088 HANDLE      WINAPI CreateEventW (LPSECURITY_ATTRIBUTES, BOOL, BOOL, LPCWSTR);
1089 #define      CreateEvent WINELIB_NAME_AW (CreateEvent)
1090 HANDLE      WINAPI CreateFileA (LPCSTR, DWORD, DWORD, LPSECURITY_ATTRIBUTES,
1091                                DWORD, DWORD, HANDLE);
1092 HANDLE      WINAPI CreateFileW (LPCWSTR, DWORD, DWORD, LPSECURITY_ATTRIBUTES,
1093                                DWORD, DWORD, HANDLE);
1094 #define      CreateFile WINELIB_NAME_AW (CreateFile)
1095 HANDLE      WINAPI CreateFileMappingA (HANDLE, LPSECURITY_ATTRIBUTES, DWORD,
1096                                       DWORD, DWORD, LPCSTR);
1097 HANDLE      WINAPI CreateFileMappingW (HANDLE, LPSECURITY_ATTRIBUTES, DWORD,
1098                                       DWORD, DWORD, LPCWSTR);
1099 #define      CreateFileMapping WINELIB_NAME_AW (CreateFileMapping)
1100 HANDLE      WINAPI CreateMutexA (LPSECURITY_ATTRIBUTES, BOOL, LPCSTR);
1101 HANDLE      WINAPI CreateMutexW (LPSECURITY_ATTRIBUTES, BOOL, LPCWSTR);
1102 #define      CreateMutex WINELIB_NAME_AW (CreateMutex)
1103 HANDLE      WINAPI CreateNamedPipeA (LPCSTR, DWORD, DWORD, DWORD, DWORD, LPSECURITY_ATTRIBUTES);
1104 HANDLE      WINAPI CreateNamedPipeW (LPCWSTR, DWORD, DWORD, DWORD, DWORD, LPSECURITY_ATTRIBUTES);
1105 #define      CreateNamedPipe WINELIB_NAME_AW (CreateNamedPipe)
1106 BOOL        WINAPI CreatePipe (PHANDLE, PHANDLE, LPSECURITY_ATTRIBUTES, DWORD);
1107 BOOL        WINAPI CreateProcessA (LPCSTR, LPSTR, LPSECURITY_ATTRIBUTES,
1108                                    LPSECURITY_ATTRIBUTES, BOOL, DWORD, LPVOID, LPCSTR,
1109                                    LPSTARTUPINFOA, LPPROCESS_INFORMATION);
1110 BOOL        WINAPI CreateProcessW (LPCWSTR, LPWSTR, LPSECURITY_ATTRIBUTES,
1111                                    LPSECURITY_ATTRIBUTES, BOOL, DWORD, LPVOID, LPCWSTR,
1112                                    LPSTARTUPINFOW, LPPROCESS_INFORMATION);
1113 #define      CreateProcess WINELIB_NAME_AW (CreateProcess)
1114 HANDLE      WINAPI CreateRemoteThread (HANDLE, LPSECURITY_ATTRIBUTES, DWORD, LPTHREAD_START_ROUTINE, LPVOID, DWORD, LPDWORD);
1115 HANDLE      WINAPI CreateSemaphoreA (LPSECURITY_ATTRIBUTES, LONG, LONG, LPCSTR);
1116 HANDLE      WINAPI CreateSemaphoreW (LPSECURITY_ATTRIBUTES, LONG, LONG, LPCWSTR);
1117 #define      CreateSemaphore WINELIB_NAME_AW (CreateSemaphore)
1118 DWORD       WINAPI CreateTapePartition (HANDLE, DWORD, DWORD, DWORD);
1119 HANDLE      WINAPI CreateThread (LPSECURITY_ATTRIBUTES, DWORD, LPTHREAD_START_ROUTINE, LPVOID, DWORD, LPDWORD);
1120 HANDLE      WINAPI CreateWaitableTimerA (LPSECURITY_ATTRIBUTES, BOOL, LPCSTR);
1121 HANDLE      WINAPI CreateWaitableTimerW (LPSECURITY_ATTRIBUTES, BOOL, LPCWSTR);
1122 #define      CreateWaitableTimer WINELIB_NAME_AW (CreateWaitableTimer)
1123 BOOL        WINAPI DebugActiveProcess (DWORD);
1124 void        WINAPI DebugBreak (void);
1125 BOOL        WINAPI DeregisterEventSource (HANDLE);
1126 BOOL        WINAPI DeviceIoControl (HANDLE, DWORD, LPVOID, DWORD, LPVOID, DWORD, LPDWORD, LPOVERLAPPED);
1127 BOOL        WINAPI DisableThreadLibraryCalls (HMODULE);
1128 BOOL        WINAPI DosDateTimeToFileTime (WORD, WORD, LPFILETIME);
1129 BOOL        WINAPI DuplicateHandle (HANDLE, HANDLE, HANDLE, HANDLE*, DWORD, BOOL, DWORD);
1130 BOOL        WINAPI EscapeCommFunction (HANDLE, UINT);
1131 BOOL        WINAPI EnumResourceLanguagesA (HMODULE, LPCSTR, LPCSTR,
1132                                           ENUMRESLANGPROCA, LONG);
1133 BOOL        WINAPI EnumResourceLanguagesW (HMODULE, LPCWSTR, LPCWSTR,
1134                                           ENUMRESLANGPROCW, LONG);
1135 #define      EnumResourceLanguages WINELIB_NAME_AW (EnumResourceLanguages)
1136 BOOL        WINAPI EnumResourceNamesA (HMODULE, LPCSTR, ENUMRESNAMEPROCA,
1137                                       LONG);
1138 BOOL        WINAPI EnumResourceNamesW (HMODULE, LPCWSTR, ENUMRESNAMEPROCW,
1139                                       LONG);
1140 #define      EnumResourceNames WINELIB_NAME_AW (EnumResourceNames)
1141 BOOL        WINAPI EnumResourceTypesA (HMODULE, ENUMRESTYPEPROCA, LONG);
1142 BOOL        WINAPI EnumResourceTypesW (HMODULE, ENUMRESTYPEPROCW, LONG);
1143 #define      EnumResourceTypes WINELIB_NAME_AW (EnumResourceTypes)
1144 BOOL        WINAPI EqualSid (PSID, PSID);
1145 BOOL        WINAPI EqualPrefixSid (PSID, PSID);
1146 DWORD       WINAPI EraseTape (HANDLE, DWORD, BOOL);
1147 VOID        WINAPI ExitProcess (DWORD) WINE_NORETURN;
1148 VOID        WINAPI ExitThread (DWORD) WINE_NORETURN;
1149 DWORD       WINAPI ExpandEnvironmentStringsA (LPCSTR, LPSTR, DWORD);

```



```

1150 DWORD        WINAPI ExpandEnvironmentStringsW(LPCWSTR, LPWSTR, DWORD);
1151 #define        ExpandEnvironmentStrings WINELIB_NAME_AW(ExpandEnvironmentStrings)
1152 BOOL          WINAPI FileTimeToDosDateTime(const FILETIME*, LPWORD, LPWORD);
1153 BOOL          WINAPI FileTimeToLocalFileTime(const FILETIME*, LPFILETIME);
1154 BOOL          WINAPI FileTimeToSystemTime(const FILETIME*, LPSYSTEMTIME);
1155 HANDLE        WINAPI FindFirstChangeNotificationA(LPCSTR, BOOL, DWORD);
1156 HANDLE        WINAPI FindFirstChangeNotificationW(LPCWSTR, BOOL, DWORD);
1157 #define        FindFirstChangeNotification WINELIB_NAME_AW(FindFirstChangeNotification)
1158 BOOL          WINAPI FindNextChangeNotification(HANDLE);
1159 BOOL          WINAPI FindCloseChangeNotification(HANDLE);
1160 HRSRC          WINAPI FindResourceExA(HMODULE, LPCSTR, LPCSTR, WORD);
1161 HRSRC          WINAPI FindResourceExW(HMODULE, LPCWSTR, LPCWSTR, WORD);
1162 #define        FindResourceEx WINELIB_NAME_AW(FindResourceEx)
1163 BOOL          WINAPI FlushFileBuffers(HANDLE);
1164 BOOL          WINAPI FlushViewOfFile(LPCVOID, DWORD);
1165 DWORD          WINAPI FormatMessageA(DWORD, LPCVOID, DWORD, DWORD, LPSTR, DWORD, va_list*);
1166 DWORD          WINAPI FormatMessageW(DWORD, LPCVOID, DWORD, DWORD, LPWSTR, DWORD, va_list*);
1167 #define        FormatMessage WINELIB_NAME_AW(FormatMessage)
1168 BOOL          WINAPI FreeEnvironmentStringsA(LPSTR);
1169 BOOL          WINAPI FreeEnvironmentStringsW(LPWSTR);
1170 #define        FreeEnvironmentStrings WINELIB_NAME_AW(FreeEnvironmentStrings)
1171 VOID          WINAPI FreeLibraryAndExitThread(HINSTANCE, DWORD);
1172 PVOID          WINAPI FreeSid(PSID);
1173 BOOL          WINAPI GetCommConfig(HANDLE, LPCOMMCONFIG, LPDWORD);
1174 BOOL          WINAPI GetCommMask(HANDLE, LPDWORD);
1175 BOOL          WINAPI GetCommModemStatus(HANDLE, LPDWORD);
1176 BOOL          WINAPI GetCommProperties(HANDLE, LPCOMMPROP);
1177 BOOL          WINAPI GetCommState(HANDLE, LPPDCB);
1178 BOOL          WINAPI GetCommTimeouts(HANDLE, LPCOMMTIMEOUTS);
1179 LPSTR          WINAPI GetCommandLineA(void);
1180 LPWSTR          WINAPI GetCommandLineW(void);
1181 #define        GetCommandLine WINELIB_NAME_AW(GetCommandLine)
1182 BOOL          WINAPI GetComputerNameA(LPSTR, LPDWORD);
1183 BOOL          WINAPI GetComputerNameW(LPWSTR, LPDWORD);
1184 #define        GetComputerName WINELIB_NAME_AW(GetComputerName)
1185 HANDLE        WINAPI GetCurrentProcess(void);
1186 HANDLE        WINAPI GetCurrentThread(void);
1187 BOOL          WINAPI GetDefaultCommConfigA(LPCSTR, LPCOMMCONFIG, LPDWORD);
1188 BOOL          WINAPI GetDefaultCommConfigW(LPCWSTR, LPCOMMCONFIG, LPDWORD);
1189 #define        GetDefaultCommConfig WINELIB_NAME_AW(GetDefaultCommConfig)
1190 LPSTR          WINAPI GetEnvironmentStringsA(void);
1191 LPWSTR          WINAPI GetEnvironmentStringsW(void);
1192 #define        GetEnvironmentStrings WINELIB_NAME_AW(GetEnvironmentStrings)
1193 DWORD          WINAPI GetEnvironmentVariableA(LPCSTR, LPSTR, DWORD);
1194 DWORD          WINAPI GetEnvironmentVariableW(LPCWSTR, LPWSTR, DWORD);
1195 #define        GetEnvironmentVariable WINELIB_NAME_AW(GetEnvironmentVariable)
1196 BOOL          WINAPI GetFileAttributesExA(LPCSTR, GET_FILEEX_INFO_LEVELS, LPVOID);
1197 BOOL          WINAPI GetFileAttributesExW(LPCWSTR, GET_FILEEX_INFO_LEVELS, LPVOID);
1198 #define        GetFileAttributesEx WINELIB_NAME_AW(GetFileAttributesEx)
1199 DWORD          WINAPI GetFileInformationByHandle(HANDLE, BY_HANDLE_FILE_INFORMATION*);
1200 BOOL          WINAPI GetFileSecurityA(LPCSTR, SECURITY_INFORMATION, PSECURITY_DESCRIPTOR, DWORD, LPDWORD);
1201 BOOL          WINAPI GetFileSecurityW(LPCWSTR, SECURITY_INFORMATION, PSECURITY_DESCRIPTOR, DWORD, LPDWORD);
1202 #define        GetFileSecurity WINELIB_NAME_AW(GetFileSecurity)
1203 DWORD          WINAPI GetFileSize(HANDLE, LPDWORD);
1204 BOOL          WINAPI GetFileTime(HANDLE, LPFILETIME, LPFILETIME, LPFILETIME);
1205 DWORD          WINAPI GetFileType(HANDLE);
1206 DWORD          WINAPI GetFullPathNameA(LPCSTR, DWORD, LPSTR, LPSTR*);
1207 DWORD          WINAPI GetFullPathNameW(LPCWSTR, DWORD, LPWSTR, LPWSTR*);
1208 #define        GetFullPathName WINELIB_NAME_AW(GetFullPathName)
1209 BOOL          WINAPI GetHandleInformation(HANDLE, LPDWORD);
1210 DWORD          WINAPI GetLengthSid(PSID);
1211 VOID          WINAPI GetLocalTime(LPSYSTEMTIME);
1212 DWORD          WINAPI GetLogicalDrives(void);
1213 DWORD          WINAPI GetLongPathNameA(LPCSTR, LPSTR, DWORD);
1214 DWORD          WINAPI GetLongPathNameW(LPCWSTR, LPWSTR, DWORD);
1215 #define        GetLongPathName WINELIB_NAME_AW(GetLongPathName)
1216 BOOL          WINAPI GetNumberOfEventLogRecords(HANDLE, PDWORD);
1217 BOOL          WINAPI GetOldestEventLogRecord(HANDLE, PDWORD);
1218 DWORD          WINAPI GetPriorityClass(HANDLE);
1219 BOOL          WINAPI GetProcessTimes(HANDLE, LPFILETIME, LPFILETIME, LPFILETIME, LPFILETIME);
1220 DWORD          WINAPI GetProcessVersion(DWORD);
1221 BOOL          WINAPI GetSecurityDescriptorControl(PSECURITY_DESCRIPTOR, PSECURITY_DESCRIPTOR_CONTROL, LPDWORD);
1222 BOOL          WINAPI GetSecurityDescriptorDacl(PSECURITY_DESCRIPTOR, LPBOOL, PACL *, LPBOOL);
1223 BOOL          WINAPI GetSecurityDescriptorGroup(PSECURITY_DESCRIPTOR, PSID *, LPBOOL);
1224 DWORD          WINAPI GetSecurityDescriptorLength(PSECURITY_DESCRIPTOR);
1225 BOOL          WINAPI GetSecurityDescriptorOwner(PSECURITY_DESCRIPTOR, PSID *, LPBOOL);
1226 BOOL          WINAPI GetSecurityDescriptorSacl(PSECURITY_DESCRIPTOR, LPBOOL, PACL *, LPBOOL);
1227 PSID_IDENTIFIER_AUTHORITY WINAPI GetSidIdentifierAuthority(PSID);
1228 DWORD          WINAPI GetSidLengthRequired(BYTE);
1229 PDWORD          WINAPI GetSidSubAuthority(PSID, DWORD);
1230 PCHAR          WINAPI GetSidSubAuthorityCount(PSID);
1231 DWORD          WINAPI GetShortPathNameA(LPCSTR, LPSTR, DWORD);
1232 DWORD          WINAPI GetShortPathNameW(LPCWSTR, LPWSTR, DWORD);
1233 #define        GetShortPathName WINELIB_NAME_AW(GetShortPathName)
1234 HANDLE        WINAPI GetStdHandle(DWORD);
1235 VOID          WINAPI GetSystemInfo(LPSYSTEM_INFO);

```



```

1236 BOOL        WINAPI GetSystemPowerStatus (LPSYSTEM_POWER_STATUS);
1237 VOID        WINAPI GetSystemTime (LPSYSTEMTIME);
1238 VOID        WINAPI GetSystemTimeAsFileTime (LPFILETIME);
1239 DWORD        WINAPI GetTapeParameters (HANDLE, DWORD, LPDWORD, LPVOID);
1240 DWORD        WINAPI GetTapePosition (HANDLE, DWORD, LPDWORD, LPDWORD, LPDWORD);
1241 DWORD        WINAPI GetTapeStatus (HANDLE);
1242 DWORD        WINAPI GetTimeZoneInformation (LPTIME_ZONE_INFORMATION);
1243 BOOL        WINAPI GetThreadContext (HANDLE, CONTEXT *);
1244 INT          WINAPI GetThreadPriority (HANDLE);
1245 BOOL        WINAPI GetThreadPriorityBoost (HANDLE, PBOOL);
1246 BOOL        WINAPI GetThreadSelectorEntry (HANDLE, DWORD, LPLDT_ENTRY);
1247 BOOL        WINAPI GetThreadTimes (HANDLE, LPFILETIME, LPFILETIME, LPFILETIME, LPFILETIME);
1248 BOOL        WINAPI GetTokenInformation (HANDLE, TOKEN_INFORMATION_CLASS, LPVOID, DWORD, LPDWORD);
1249 BOOL        WINAPI GetUserNameA (LPSTR, LPDWORD);
1250 BOOL        WINAPI GetUserNameW (LPWSTR, LPDWORD);
1251 #define       GetUserInformation WINELIB_NAME_AW(GetUserName)
1252 VOID        WINAPI GlobalMemoryStatus (LPMEMORYSTATUS);
1253 DWORD        WINAPI HeapCompact (HANDLE, DWORD);
1254 HANDLE       WINAPI HeapCreate (DWORD, DWORD, DWORD);
1255 BOOL        WINAPI HeapDestroy (HANDLE);
1256 BOOL        WINAPI HeapLock (HANDLE);
1257 BOOL        WINAPI HeapUnlock (HANDLE);
1258 BOOL        WINAPI HeapValidate (HANDLE, DWORD, LPCVOID);
1259 BOOL        WINAPI HeapWalk (HANDLE, LPPROCESS_HEAP_ENTRY);
1260 DWORD        WINAPI InitializeAcl (PACL, DWORD, DWORD);
1261 BOOL        WINAPI InitializeSecurityDescriptor (PSECURITY_DESCRIPTOR, DWORD);
1262 BOOL        WINAPI InitializeSid (PSID, PSID_IDENTIFIER_AUTHORITY, BYTE);
1263 BOOL        WINAPI IsTextUnicode (CONST LPVOID lpBuffer, int cb, LPINT lpi);
1264 BOOL        WINAPI IsValidSecurityDescriptor (PSECURITY_DESCRIPTOR);
1265 BOOL        WINAPI IsValidSid (PSID);
1266 BOOL        WINAPI ImpersonateSelf (SECURITY_IMPERSONATION_LEVEL);
1267 BOOL        WINAPI IsProcessorFeaturePresent (DWORD);
1268 BOOL        WINAPI LookupAccountSida (LPCSTR, PSID, LPSTR, LPDWORD, LPSTR, LPDWORD, PSID_NAME_USE);
1269 BOOL        WINAPI LookupAccountSidW (LPCWSTR, PSID, LPWSTR, LPDWORD, LPWSTR, LPDWORD, PSID_NAME_USE);
1270 #define       LookupAccountSid WINELIB_NAME_AW(LookupAccountSid)
1271 BOOL        WINAPI LocalFileTimeToFileTime (const FILETIME*, LPFILETIME);
1272 BOOL        WINAPI LockFile (HANDLE, DWORD, DWORD, DWORD, DWORD);
1273 BOOL        WINAPI LockFileEx (HANDLE, DWORD, DWORD, DWORD, DWORD, LPOVERLAPPED);
1274 BOOL        WINAPI LookupPrivilegeValueA (LPCSTR, LPCSTR, LPVOID);
1275 BOOL        WINAPI LookupPrivilegeValueW (LPCWSTR, LPCWSTR, LPVOID);
1276 #define       LookupPrivilegeValue WINELIB_NAME_AW(LookupPrivilegeValue)
1277 BOOL        WINAPI MakeSelfRelativeSD (PSECURITY_DESCRIPTOR, PSECURITY_DESCRIPTOR, LPDWORD);
1278 HMODULE      WINAPI MapHModuleSL (WORD);
1279 WORD        WINAPI MapHModuleLS (HMODULE);
1280 LPVOID       WINAPI MapViewOfFile (HANDLE, DWORD, DWORD, DWORD, DWORD);
1281 LPVOID       WINAPI MapViewOfFileEx (HANDLE, DWORD, DWORD, DWORD, DWORD, LPVOID);
1282 BOOL        WINAPI MoveFileA (LPCSTR, LPCSTR);
1283 BOOL        WINAPI MoveFileW (LPCWSTR, LPCWSTR);
1284 #define       MoveFile WINELIB_NAME_AW(MoveFile)
1285 BOOL        WINAPI MoveFileExA (LPCSTR, LPCSTR, DWORD);
1286 BOOL        WINAPI MoveFileExW (LPCWSTR, LPCWSTR, DWORD);
1287 #define       MoveFileEx WINELIB_NAME_AW(MoveFileEx)
1288 BOOL        WINAPI NotifyChangeEventLog (HANDLE, HANDLE);
1289 HANDLE       WINAPI OpenBackupEventLogA (LPCSTR, LPCSTR);
1290 HANDLE       WINAPI OpenBackupEventLogW (LPCWSTR, LPCWSTR);
1291 #define       OpenBackupEventLog WINELIB_NAME_AW(OpenBackupEventLog)
1292 HANDLE       WINAPI OpenEventA (DWORD, BOOL, LPCSTR);
1293 HANDLE       WINAPI OpenEventW (DWORD, BOOL, LPCWSTR);
1294 #define       OpenEvent WINELIB_NAME_AW(OpenEvent)
1295 HANDLE       WINAPI OpenEventLogA (LPCSTR, LPCSTR);
1296 HANDLE       WINAPI OpenEventLogW (LPCWSTR, LPCWSTR);
1297 #define       OpenEventLog WINELIB_NAME_AW(OpenEventLog)
1298 HANDLE       WINAPI OpenFileMappingA (DWORD, BOOL, LPCSTR);
1299 HANDLE       WINAPI OpenFileMappingW (DWORD, BOOL, LPCWSTR);
1300 #define       OpenFileMapping WINELIB_NAME_AW(OpenFileMapping)
1301 HANDLE       WINAPI OpenMutexA (DWORD, BOOL, LPCSTR);
1302 HANDLE       WINAPI OpenMutexW (DWORD, BOOL, LPCWSTR);
1303 #define       OpenMutex WINELIB_NAME_AW(OpenMutex)
1304 HANDLE       WINAPI OpenProcess (DWORD, BOOL, DWORD);
1305 BOOL        WINAPI OpenProcessToken (HANDLE, DWORD, PHANDLE);
1306 HANDLE       WINAPI OpenSemaphoreA (DWORD, BOOL, LPCSTR);
1307 HANDLE       WINAPI OpenSemaphoreW (DWORD, BOOL, LPCWSTR);
1308 #define       OpenSemaphore WINELIB_NAME_AW(OpenSemaphore)
1309 BOOL        WINAPI OpenThreadToken (HANDLE, DWORD, BOOL, PHANDLE);
1310 HANDLE       WINAPI OpenWaitableTimerA (DWORD, BOOL, LPCSTR);
1311 HANDLE       WINAPI OpenWaitableTimerW (DWORD, BOOL, LPCWSTR);
1312 #define       OpenWaitableTimer WINELIB_NAME_AW(OpenWaitableTimer)
1313 DWORD        WINAPI PrepareTape (HANDLE, DWORD, BOOL);
1314 BOOL        WINAPI PulseEvent (HANDLE);
1315 BOOL        WINAPI PurgeComm (HANDLE, DWORD);
1316 DWORD        WINAPI QueryDosDeviceA (LPCSTR, LPSTR, DWORD);
1317 DWORD        WINAPI QueryDosDeviceW (LPCWSTR, LPWSTR, DWORD);
1318 #define       QueryDosDevice WINELIB_NAME_AW(QueryDosDevice)
1319 BOOL        WINAPI QueryPerformanceCounter (LARGE_INTEGER*);
1320 BOOL        WINAPI QueryPerformanceFrequency (LARGE_INTEGER*);
1321 BOOL        WINAPI ReadEventLogA (HANDLE, DWORD, DWORD, LPVOID, DWORD, DWORD *, DWORD *);
1322 BOOL        WINAPI ReadEventLogW (HANDLE, DWORD, DWORD, LPVOID, DWORD, DWORD *, DWORD *);

```

```

1323 #define      ReadEventLog WINELIB_NAME_AW(ReadEventLog)
1324 BOOL        WINAPI ReadFile(HANDLE, LPVOID, DWORD, LPDWORD, LPOVERLAPPED);
1325 BOOL        WINAPI ReadFileEx(HANDLE, LPVOID, DWORD, LPOVERLAPPED, LPOVERLAPPED_COMPLETION_ROUTINE);
1326 HANDLE      WINAPI RegisterEventSourceA(LPCSTR, LPCSTR);
1327 HANDLE      WINAPI RegisterEventSourceW(LPCWSTR, LPCWSTR);
1328 #define      RegisterEventSource WINELIB_NAME_AW(RegisterEventSource)
1329 BOOL        WINAPI ReleaseMutex(HANDLE);
1330 BOOL        WINAPI ReleaseSemaphore(HANDLE, LONG, LPLONG);
1331 BOOL        WINAPI ReportEventA(HANDLE, WORD, WORD, DWORD, PSID, WORD, DWORD, LPCSTR *, LPVOID);
1332 BOOL        WINAPI ReportEventW(HANDLE, WORD, WORD, DWORD, PSID, WORD, DWORD, LPCWSTR *, LPVOID);
1333 #define      ReportEvent WINELIB_NAME_AW(ReportEvent)
1334 BOOL        WINAPI ResetEvent(HANDLE);
1335 DWORD       WINAPI ResumeThread(HANDLE);
1336 BOOL        WINAPI RevertToSelf(void);
1337 DWORD       WINAPI SearchPathA(LPCSTR, LPCSTR, LPCSTR, DWORD, LPSTR, LPSTR*);
1338 DWORD       WINAPI SearchPathW(LPCWSTR, LPCWSTR, LPCWSTR, DWORD, LPWSTR, LPWSTR*);
1339 #define      SearchPath WINELIB_NAME_AW(SearchPath)
1340 BOOL        WINAPI SetCommConfig(HANDLE, LPCOMMCONFIG, DWORD);
1341 BOOL        WINAPI SetCommBreak(HANDLE);
1342 BOOL        WINAPI SetCommMask(HANDLE, DWORD);
1343 BOOL        WINAPI SetCommState(HANDLE, LPDCB);
1344 BOOL        WINAPI SetCommTimeouts(HANDLE, LPCOMMTIMEOUTS);
1345 BOOL        WINAPI SetComputerNameA(LPCSTR);
1346 BOOL        WINAPI SetComputerNameW(LPCWSTR);
1347 #define      SetComputerName WINELIB_NAME_AW(SetComputerName)
1348 BOOL        WINAPI SetDefaultCommConfigA(LPCSTR, LPCOMMCONFIG, DWORD);
1349 BOOL        WINAPI SetDefaultCommConfigW(LPCWSTR, LPCOMMCONFIG, DWORD);
1350 #define      SetDefaultCommConfig WINELIB_NAME_AW(SetDefaultCommConfig)
1351 BOOL        WINAPI SetEndOfFile(HANDLE);
1352 BOOL        WINAPI SetEnvironmentVariableA(LPCSTR, LPCSTR);
1353 BOOL        WINAPI SetEnvironmentVariableW(LPCWSTR, LPCWSTR);
1354 #define      SetEnvironmentVariable WINELIB_NAME_AW(SetEnvironmentVariable)
1355 BOOL        WINAPI SetEvent(HANDLE);
1356 VOID        WINAPI SetFileApisToANSI(void);
1357 VOID        WINAPI SetFileApisToOEM(void);
1358 DWORD       WINAPI SetFilePointer(HANDLE, LONG, LPLONG, DWORD);
1359 BOOL        WINAPI SetFileSecurityA(LPCSTR, SECURITY_INFORMATION, PSECURITY_DESCRIPTOR);
1360 BOOL        WINAPI SetFileSecurityW(LPCWSTR, SECURITY_INFORMATION, PSECURITY_DESCRIPTOR);
1361 #define      SetFileSecurity WINELIB_NAME_AW(SetFileSecurity)
1362 BOOL        WINAPI SetFileTime(HANDLE, const FILETIME*, const FILETIME*, const FILETIME*);
1363 BOOL        WINAPI SetHandleInformation(HANDLE, DWORD, DWORD);
1364 BOOL        WINAPI SetKernelObjectSecurity(HANDLE, SECURITY_INFORMATION, PSECURITY_DESCRIPTOR);
1365 BOOL        WINAPI SetPriorityClass(HANDLE, DWORD);
1366 BOOL        WINAPI SetLocalTime(const SYSTEMTIME*);
1367 BOOL        WINAPI SetSecurityDescriptorDacl(PSECURITY_DESCRIPTOR, BOOL, PACL, BOOL);
1368 BOOL        WINAPI SetSecurityDescriptorGroup(PSECURITY_DESCRIPTOR, PSID, BOOL);
1369 BOOL        WINAPI SetSecurityDescriptorOwner(PSECURITY_DESCRIPTOR, PSID, BOOL);
1370 BOOL        WINAPI SetSecurityDescriptorSacl(PSECURITY_DESCRIPTOR, BOOL, PACL, BOOL);
1371 BOOL        WINAPI SetStdHandle(DWORD, HANDLE);
1372 BOOL        WINAPI SetSystemPowerState(BOOL, BOOL);
1373 BOOL        WINAPI SetSystemTime(const SYSTEMTIME*);
1374 DWORD       WINAPI SetTapeParameters(HANDLE, DWORD, LPVOID);
1375 DWORD       WINAPI SetTapePosition(HANDLE, DWORD, DWORD, DWORD, DWORD, BOOL);
1376 DWORD       WINAPI SetThreadAffinityMask(HANDLE, DWORD);
1377 BOOL        WINAPI SetThreadContext(HANDLE, const CONTEXT *);
1378 DWORD       WINAPI SetThreadExecutionState(EXECUTION_STATE);
1379 BOOL        WINAPI SetThreadPriority(HANDLE, INT);
1380 BOOL        WINAPI SetThreadPriorityBoost(HANDLE, BOOL);
1381 BOOL        WINAPI SetThreadToken(PHANDLE, HANDLE);
1382 BOOL        WINAPI SetTimeZoneInformation(const LPTIME_ZONE_INFORMATION);
1383 BOOL        WINAPI SetWaitableTimer(HANDLE, const LARGE_INTEGER*, LONG, PTIMERAPCRoutine, LPVOID, BOOL);
1384 BOOL        WINAPI SetupComm(HANDLE, DWORD, DWORD);
1385 VOID        WINAPI Sleep(DWORD);
1386 DWORD       WINAPI SleepEx(DWORD, BOOL);
1387 DWORD       WINAPI SuspendThread(HANDLE);
1388 BOOL        WINAPI SystemTimeToFileTime(const SYSTEMTIME*, LPFILETIME);
1389 DWORD       WINAPI TlsAlloc(void);
1390 BOOL        WINAPI TlsFree(DWORD);
1391 LPVOID       WINAPI TlsGetValue(DWORD);
1392 BOOL        WINAPI TlsSetValue(DWORD, LPVOID);
1393 BOOL        WINAPI TransmitCommChar(HANDLE, CHAR);
1394 BOOL        WINAPI UnlockFile(HANDLE, DWORD, DWORD, DWORD, DWORD);
1395 BOOL        WINAPI UnmapViewOfFile(LPVOID);
1396 LPVOID       WINAPI VirtualAlloc(LPVOID, DWORD, DWORD, DWORD);
1397 LPVOID       WINAPI VirtualAllocEx(HANDLE, LPVOID, DWORD, DWORD, DWORD);
1398 BOOL        WINAPI VirtualFree(LPVOID, DWORD, DWORD);
1399 BOOL        WINAPI VirtualLock(LPVOID, DWORD);
1400 BOOL        WINAPI VirtualProtect(LPVOID, DWORD, DWORD, LPDWORD);
1401 BOOL        WINAPI VirtualProtectEx(HANDLE, LPVOID, DWORD, DWORD, LPDWORD);
1402 DWORD       WINAPI VirtualQuery(LPCVOID, LPMEMORY_BASIC_INFORMATION, DWORD);
1403 DWORD       WINAPI VirtualQueryEx(HANDLE, LPCVOID, LPMEMORY_BASIC_INFORMATION, DWORD);
1404 BOOL        WINAPI VirtualUnlock(LPVOID, DWORD);
1405 BOOL        WINAPI WaitCommEvent(HANDLE, LPDWORD, LPOVERLAPPED);
1406 BOOL        WINAPI WaitForDebugEvent(LPDEBUG_EVENT, DWORD);
1407 DWORD       WINAPI WaitForMultipleObjects(DWORD, const HANDLE*, BOOL, DWORD);
1408 DWORD       WINAPI WaitForMultipleObjectsEx(DWORD, const HANDLE*, BOOL, DWORD, BOOL);
1409 DWORD       WINAPI WaitForSingleObject(HANDLE, DWORD);

```

```

1410 DWORD      WINAPI WaitForSingleObjectEx (HANDLE, DWORD, BOOL);
1411 BOOL        WINAPI WaitNamedPipeA (LPCSTR, DWORD);
1412 BOOL        WINAPI WaitNamedPipeW (LPCWSTR, DWORD);
1413 #define      WaitNamedPipe WINELIB_NAME_AW(WaitNamedPipe)
1414 BOOL        WINAPI WriteFile (HANDLE, LPCVOID, DWORD, LPDWORD, LPOVERLAPPED);
1415 BOOL        WINAPI WriteFileEx (HANDLE, LPCVOID, DWORD, LPOVERLAPPED, LPOVERLAPPED_COMPLETION_ROUTINE);
1416 DWORD      WINAPI WriteTapemark (HANDLE, DWORD, DWORD, BOOL);
1417 ATOM        WINAPI AddAtomA (LPCSTR);
1418 ATOM        WINAPI AddAtomW (LPCWSTR);
1419 #define      AddAtom WINELIB_NAME_AW(AddAtom)
1420 BOOL        WINAPI CreateDirectoryA (LPCSTR, LPSECURITY_ATTRIBUTES);
1421 BOOL        WINAPI CreateDirectoryW (LPCWSTR, LPSECURITY_ATTRIBUTES);
1422 #define      CreateDirectory WINELIB_NAME_AW(CreateDirectory)
1423 BOOL        WINAPI CreateDirectoryExA (LPCSTR, LPCSTR, LPSECURITY_ATTRIBUTES);
1424 BOOL        WINAPI CreateDirectoryExW (LPCWSTR, LPCWSTR, LPSECURITY_ATTRIBUTES);
1425 #define      CreateDirectoryEx WINELIB_NAME_AW(CreateDirectoryEx)
1426 BOOL        WINAPI DefineDosDeviceA (DWORD, LPCSTR, LPCSTR);
1427 #define      DefineHandleTable(w) ((w), TRUE)
1428 ATOM        WINAPI DeleteAtom (ATOM);
1429 BOOL        WINAPI DeleteFileA (LPCSTR);
1430 BOOL        WINAPI DeleteFileW (LPCWSTR);
1431 #define      DeleteFile WINELIB_NAME_AW(DeleteFile)
1432 void        WINAPI FatalAppExitA (UINT, LPCSTR);
1433 void        WINAPI FatalAppExitW (UINT, LPCWSTR);
1434 #define      FatalAppExit WINELIB_NAME_AW(FatalAppExit)
1435 ATOM        WINAPI FindAtomA (LPCSTR);
1436 ATOM        WINAPI FindAtomW (LPCWSTR);
1437 #define      FindAtom WINELIB_NAME_AW(FindAtom)
1438 BOOL        WINAPI FindClose (HANDLE);
1439 HANDLE      WINAPI FindFirstFileA (LPCSTR, LPWIN32_FIND_DATA);
1440 HANDLE      WINAPI FindFirstFileW (LPCWSTR, LPWIN32_FIND_DATA);
1441 #define      FindFirstFile WINELIB_NAME_AW(FindFirstFile)
1442 HANDLE      WINAPI FindFirstFileExA (LPCSTR, FINDEX_INFO_LEVELS, LPVOID, FINDEX_SEARCH_OPS, LPVOID, DWORD);
1443 HANDLE      WINAPI FindFirstFileExW (LPCWSTR, FINDEX_INFO_LEVELS, LPVOID, FINDEX_SEARCH_OPS, LPVOID, DWORD);
1444 #define      FindFirstFileEx WINELIB_NAME_AW(FindFirstFileEx)
1445 BOOL        WINAPI FindNextFileA (HANDLE, LPWIN32_FIND_DATA);
1446 BOOL        WINAPI FindNextFileW (HANDLE, LPWIN32_FIND_DATA);
1447 #define      FindNextFile WINELIB_NAME_AW(FindNextFile)
1448 HRSRC       WINAPI FindResourceA (HMODULE, LPCSTR, LPCSTR);
1449 HRSRC       WINAPI FindResourceW (HMODULE, LPCWSTR, LPCWSTR);
1450 #define      FindResource WINELIB_NAME_AW(FindResource)
1451 BOOL        WINAPI FreeLibrary (HMODULE);
1452 #define      FreeModule(handle) FreeLibrary(handle)
1453 #define      FreeProcInstance(proc) /*nothing*/
1454 BOOL        WINAPI FreeResource (HGLOBAL);
1455 UINT        WINAPI GetAtomNameA (ATOM, LPSTR, INT);
1456 UINT        WINAPI GetAtomNameW (ATOM, LPWSTR, INT);
1457 #define      GetAtomName WINELIB_NAME_AW(GetAtomName)
1458 UINT        WINAPI GetCurrentDirectoryA (UINT, LPSTR);
1459 UINT        WINAPI GetCurrentDirectoryW (UINT, LPWSTR);
1460 #define      GetCurrentDirectory WINELIB_NAME_AW(GetCurrentDirectory)
1461 #define      GetCurrentTime() GetTickCount()
1462 BOOL        WINAPI GetDiskFreeSpaceA (LPCSTR, LPDWORD, LPDWORD, LPDWORD, LPDWORD);
1463 BOOL        WINAPI GetDiskFreeSpaceW (LPCWSTR, LPDWORD, LPDWORD, LPDWORD, LPDWORD);
1464 #define      GetDiskFreeSpace WINELIB_NAME_AW(GetDiskFreeSpace)
1465 BOOL        WINAPI GetDiskFreeSpaceExA (LPCSTR, PULARGE_INTEGER, PULARGE_INTEGER, PULARGE_INTEGER);
1466 BOOL        WINAPI GetDiskFreeSpaceExW (LPCWSTR, PULARGE_INTEGER, PULARGE_INTEGER, PULARGE_INTEGER);
1467 #define      GetDiskFreeSpaceEx WINELIB_NAME_AW(GetDiskFreeSpaceEx)
1468 UINT        WINAPI GetDriveTypeA (LPCSTR);
1469 UINT        WINAPI GetDriveTypeW (LPCWSTR);
1470 #define      GetDriveType WINELIB_NAME_AW(GetDriveType)
1471 BOOL        WINAPI GetExitCodeProcess (HANDLE, LPDWORD);
1472 DWORD      WINAPI GetFileAttributesA (LPCSTR);
1473 DWORD      WINAPI GetFileAttributesW (LPCWSTR);
1474 #define      GetFileAttributes WINELIB_NAME_AW(GetFileAttributes)
1475 #define      GetFreeSpace(w) (0x100000L)
1476 UINT        WINAPI GetLogicalDriveStringsA (UINT, LPSTR);
1477 UINT        WINAPI GetLogicalDriveStringsW (UINT, LPWSTR);
1478 #define      GetLogicalDriveStrings WINELIB_NAME_AW(GetLogicalDriveStrings)
1479 DWORD      WINAPI GetModuleFileNameA (HMODULE, LPSTR, DWORD);
1480 DWORD      WINAPI GetModuleFileNameW (HMODULE, LPWSTR, DWORD);
1481 #define      GetModuleFileName WINELIB_NAME_AW(GetModuleFileName)
1482 HMODULE     WINAPI GetModuleHandleA (LPCSTR);
1483 HMODULE     WINAPI GetModuleHandleW (LPCWSTR);
1484 #define      GetModuleHandle WINELIB_NAME_AW(GetModuleHandle)
1485 BOOL        WINAPI GetOverlappedResult (HANDLE, LPOVERLAPPED, LPDWORD, BOOL);
1486 UINT        WINAPI GetPrivateProfileIntA (LPCSTR, LPCSTR, INT, LPCSTR);
1487 UINT        WINAPI GetPrivateProfileIntW (LPCWSTR, LPCWSTR, INT, LPCWSTR);
1488 #define      GetPrivateProfileInt WINELIB_NAME_AW(GetPrivateProfileInt)
1489 INT         WINAPI GetPrivateProfileSectionA (LPCSTR, LPSTR, DWORD, LPCSTR);
1490 INT         WINAPI GetPrivateProfileSectionW (LPCWSTR, LPWSTR, DWORD, LPCWSTR);
1491 #define      GetPrivateProfileSection WINELIB_NAME_AW(GetPrivateProfileSection)
1492 DWORD      WINAPI GetPrivateProfileSectionNamesA (LPSTR, DWORD, LPCSTR);
1493 DWORD      WINAPI GetPrivateProfileSectionNamesW (LPWSTR, DWORD, LPCWSTR);
1494 #define      GetPrivateProfileSectionNames WINELIB_NAME_AW(GetPrivateProfileSectionNames)
1495 INT         WINAPI GetPrivateProfileStringA (LPCSTR, LPCSTR, LPCSTR, LPSTR, UINT, LPCSTR);
1496 INT         WINAPI GetPrivateProfileStringW (LPCWSTR, LPCWSTR, LPCWSTR, LPWSTR, UINT, LPCWSTR);

```

```

1497 #define      GetPrivateProfileString WINELIB_NAME_AW(GetPrivateProfileString)
1498 BOOL        WINAPI GetPrivateProfileStructA(LPCSTR, LPCSTR, LPVOID, UINT, LPCSTR);
1499 BOOL        WINAPI GetPrivateProfileStructW(LPCWSTR, LPCWSTR, LPVOID, UINT, LPCWSTR);
1500 #define      GetPrivateProfileStruct WINELIB_NAME_AW(GetPrivateProfileStruct)
1501 FARPROC     WINAPI GetProcAddress(HMODULE, LPCSTR);
1502 UINT        WINAPI GetProfileIntA(LPCSTR, LPCSTR, INT);
1503 UINT        WINAPI GetProfileIntW(LPCWSTR, LPCWSTR, INT);
1504 #define      GetProfileInt WINELIB_NAME_AW(GetProfileInt)
1505 INT         WINAPI GetProfileSectionA(LPCSTR, LPSTR, DWORD);
1506 INT         WINAPI GetProfileSectionW(LPCWSTR, LPWSTR, DWORD);
1507 #define      GetProfileSection WINELIB_NAME_AW(GetProfileSection)
1508 INT         WINAPI GetProfileStringA(LPCSTR, LPCSTR, LPCSTR, LPSTR, UINT);
1509 INT         WINAPI GetProfileStringW(LPCWSTR, LPCWSTR, LPCWSTR, LPWSTR, UINT);
1510 #define      GetProfileString WINELIB_NAME_AW(GetProfileString)
1511 VOID        WINAPI GetStartupInfoA(LPSTARTUPINFOA);
1512 VOID        WINAPI GetStartupInfoW(LPSTARTUPINFOW);
1513 #define      GetStartupInfo WINELIB_NAME_AW(GetStartupInfo)
1514 UINT        WINAPI GetSystemDirectoryA(LPSTR, UINT);
1515 UINT        WINAPI GetSystemDirectoryW(LPWSTR, UINT);
1516 #define      GetSystemDirectory WINELIB_NAME_AW(GetSystemDirectory)
1517 DWORD       WINAPI GetTickCount(void);
1518 UINT        WINAPI GetTempFileNameA(LPCSTR, LPCSTR, UINT, LPSTR);
1519 UINT        WINAPI GetTempFileNameW(LPCWSTR, LPCWSTR, UINT, LPWSTR);
1520 #define      GetTempFileName WINELIB_NAME_AW(GetTempFileName)
1521 UINT        WINAPI GetTempPathA(UINT, LPSTR);
1522 UINT        WINAPI GetTempPathW(UINT, LPWSTR);
1523 #define      GetTempPath WINELIB_NAME_AW(GetTempPath)
1524 LONG        WINAPI GetVersion(void);
1525 BOOL        WINAPI GetVolumeInformationA(LPCSTR, LPSTR, DWORD, LPDWORD, LPDWORD, LPDWORD, LPSTR, DWORD);
1526 BOOL        WINAPI GetVolumeInformationW(LPCWSTR, LPWSTR, DWORD, LPDWORD, LPDWORD, LPDWORD, LPWSTR, DWORD);
1527 #define      GetVolumeInformation WINELIB_NAME_AW(GetVolumeInformation)
1528 UINT        WINAPI GetWindowsDirectoryA(LPSTR, UINT);
1529 UINT        WINAPI GetWindowsDirectoryW(LPWSTR, UINT);
1530 #define      GetWindowsDirectory WINELIB_NAME_AW(GetWindowsDirectory)
1531 ATOM        WINAPI GlobalAddAtomA(LPCSTR);
1532 ATOM        WINAPI GlobalAddAtomW(LPCWSTR);
1533 #define      GlobalAddAtom WINELIB_NAME_AW(GlobalAddAtom)
1534 HGLOBAL     WINAPI GlobalAlloc(UINT, DWORD);
1535 DWORD       WINAPI GlobalCompact(DWORD);
1536 ATOM        WINAPI GlobalDeleteAtom(ATOM);
1537 ATOM        WINAPI GlobalFindAtomA(LPCSTR);
1538 ATOM        WINAPI GlobalFindAtomW(LPCWSTR);
1539 #define      GlobalFindAtom WINELIB_NAME_AW(GlobalFindAtom)
1540 UINT        WINAPI GlobalFlags(HGLOBAL);
1541 HGLOBAL     WINAPI GlobalFree(HGLOBAL);
1542 UINT        WINAPI GlobalGetAtomNameA(ATOM, LPSTR, INT);
1543 UINT        WINAPI GlobalGetAtomNameW(ATOM, LPWSTR, INT);
1544 #define      GlobalGetAtomName WINELIB_NAME_AW(GlobalGetAtomName)
1545 HGLOBAL     WINAPI GlobalHandle(LPCVOID);
1546 VOID        WINAPI GlobalFix(HGLOBAL);
1547 LPVOID      WINAPI GlobalLock(HGLOBAL);
1548 HGLOBAL     WINAPI GlobalReAlloc(HGLOBAL, DWORD, UINT);
1549 DWORD       WINAPI GlobalSize(HGLOBAL);
1550 VOID        WINAPI GlobalUnfix(HGLOBAL);
1551 BOOL        WINAPI GlobalUnlock(HGLOBAL);
1552 BOOL        WINAPI GlobalUnWire(HGLOBAL);
1553 LPVOID      WINAPI GlobalWire(HGLOBAL);
1554 #define      HasOverlappedCompleted(lpOverlapped) ((lpOverlapped)->Internal != STATUS_PENDING)
1555 BOOL        WINAPI InitAtomTable(DWORD);
1556 BOOL        WINAPI IsBadCodePtr(FARPROC);
1557 BOOL        WINAPI IsBadHugeReadPtr(LPCVOID, UINT);
1558 BOOL        WINAPI IsBadHugeWritePtr(LPVOID, UINT);
1559 BOOL        WINAPI IsBadReadPtr(LPCVOID, UINT);
1560 BOOL        WINAPI IsBadStringPtrA(LPCSTR, UINT);
1561 BOOL        WINAPI IsBadStringPtrW(LPCWSTR, UINT);
1562 #define      IsBadStringPtr WINELIB_NAME_AW(IsBadStringPtr)
1563 BOOL        WINAPI IsBadWritePtr(LPVOID, UINT);
1564 BOOL        WINAPI IsDebuggerPresent(void);
1565 HMODULE     WINAPI LoadLibraryA(LPCSTR);
1566 HMODULE     WINAPI LoadLibraryW(LPCWSTR);
1567 #define      LoadLibrary WINELIB_NAME_AW(LoadLibrary)
1568 HMODULE     WINAPI LoadLibraryExA(LPCSTR, HANDLE, DWORD);
1569 HMODULE     WINAPI LoadLibraryExW(LPCWSTR, HANDLE, DWORD);
1570 #define      LoadLibraryEx WINELIB_NAME_AW(LoadLibraryEx)
1571 HINSTANCE   WINAPI LoadModule(LPCSTR, LPVOID);
1572 HGLOBAL     WINAPI LoadResource(HMODULE, HRSRC);
1573 HLOCAL      WINAPI LocalAlloc(UINT, DWORD);
1574 UINT        WINAPI LocalCompact(UINT);
1575 UINT        WINAPI LocalFlags(HLOCAL);
1576 HLOCAL      WINAPI LocalFree(HLOCAL);
1577 HLOCAL      WINAPI LocalHandle(LPCVOID);
1578 LPVOID      WINAPI LocalLock(HLOCAL);
1579 HLOCAL      WINAPI LocalReAlloc(HLOCAL, DWORD, UINT);
1580 UINT        WINAPI LocalShrink(HGLOBAL, UINT);
1581 UINT        WINAPI LocalSize(HLOCAL);
1582 BOOL        WINAPI LocalUnlock(HLOCAL);
1583 LPVOID      WINAPI LockResource(HGLOBAL);

```

```

1584 #define LockSegment(handle) GlobalFix((HANDLE)(handle))
1585 #define MakeProcInstance(proc,inst) (proc)
1586 HFILE WINAPI OpenFile(LPCSTR,OFSTRUCT*,UINT);
1587 VOID WINAPI OutputDebugStringA(LPCSTR);
1588 VOID WINAPI OutputDebugStringW(LPCWSTR);
1589 #define OutputDebugString WINELIB_NAME_AW(OutputDebugString)
1590 BOOL WINAPI ReadProcessMemory(HANDLE, LPCVOID, LPVOID, DWORD, LPDWORD);
1591 BOOL WINAPI RemoveDirectoryA(LPCSTR);
1592 BOOL WINAPI RemoveDirectoryW(LPCWSTR);
1593 #define RemoveDirectory WINELIB_NAME_AW(RemoveDirectory)
1594 BOOL WINAPI SetCurrentDirectoryA(LPCSTR);
1595 BOOL WINAPI SetCurrentDirectoryW(LPCWSTR);
1596 #define SetCurrentDirectory WINELIB_NAME_AW(SetCurrentDirectory)
1597 UINT WINAPI SetErrorMode(UINT);
1598 BOOL WINAPI SetFileAttributesA(LPCSTR,DWORD);
1599 BOOL WINAPI SetFileAttributesW(LPCWSTR,DWORD);
1600 #define SetFileAttributes WINELIB_NAME_AW(SetFileAttributes)
1601 UINT WINAPI SetHandleCount(UINT);
1602 #define SetSwapAreaSize(w) (w)
1603 BOOL WINAPI SetVolumeLabelA(LPCSTR,LPCSTR);
1604 BOOL WINAPI SetVolumeLabelW(LPCWSTR,LPCWSTR);
1605 #define SetVolumeLabel WINELIB_NAME_AW(SetVolumeLabel)
1606 DWORD WINAPI SizeofResource(HMODULE,HRSRC);
1607 BOOL WINAPI UnlockFileEx(HFILE,DWORD,DWORD,DWORD,LPOVERLAPPED);
1608 #define UnlockSegment(handle) GlobalUnfix((HANDLE)(handle))
1609 BOOL WINAPI WritePrivateProfileSectionA(LPCSTR,LPCSTR,LPCSTR);
1610 BOOL WINAPI WritePrivateProfileSectionW(LPCWSTR,LPCWSTR,LPCWSTR);
1611 #define WritePrivateProfileSection WINELIB_NAME_AW(WritePrivateProfileSection)
1612 BOOL WINAPI WritePrivateProfileStringA(LPCSTR,LPCSTR,LPCSTR,LPCSTR);
1613 BOOL WINAPI WritePrivateProfileStringW(LPCWSTR,LPCWSTR,LPCWSTR,LPCWSTR);
1614 #define WritePrivateProfileString WINELIB_NAME_AW(WritePrivateProfileString)
1615 BOOL WINAPI WriteProfileSectionA(LPCSTR,LPCSTR);
1616 BOOL WINAPI WriteProfileSectionW(LPCWSTR,LPCWSTR);
1617 #define WritePrivateProfileSection WINELIB_NAME_AW(WritePrivateProfileSection)
1618 BOOL WINAPI WritePrivateProfileStructA(LPCSTR,LPCSTR,LPVOID,UINT,LPCSTR);
1619 BOOL WINAPI WritePrivateProfileStructW(LPCWSTR,LPCWSTR,LPVOID,UINT,LPCWSTR);
1620 #define WritePrivateProfileStruct WINELIB_NAME_AW(WritePrivateProfileStruct)
1621 BOOL WINAPI WriteProcessMemory(HANDLE,LPVOID,LPCVOID,DWORD,LPDWORD);
1622 BOOL WINAPI WriteProfileStringA(LPCSTR,LPCSTR,LPCSTR);
1623 BOOL WINAPI WriteProfileStringW(LPCWSTR,LPCWSTR,LPCWSTR);
1624 #define WriteProfileString WINELIB_NAME_AW(WriteProfileString)
1625 #define Yield()
1626 LPSTR WINAPI lstrcatA(LPSTR,LPCSTR);
1627 LPWSTR WINAPI lstrcatW(LPWSTR,LPCWSTR);
1628 #define lstrcat WINELIB_NAME_AW(lstrcat)
1629 LPSTR WINAPI lstrcpyA(LPSTR,LPCSTR);
1630 LPWSTR WINAPI lstrcpyW(LPWSTR,LPCWSTR);
1631 #define lstrcpy WINELIB_NAME_AW(lstrcpy)
1632 LPSTR WINAPI lstrcpynA(LPSTR,LPCSTR,INT);
1633 LPWSTR WINAPI lstrcpynW(LPWSTR,LPCWSTR,INT);
1634 #define lstrcpyn WINELIB_NAME_AW(lstrcpyn)
1635 INT WINAPI lstrlenA(LPCSTR);
1636 INT WINAPI lstrlenW(LPCWSTR);
1637 #define lstrlen WINELIB_NAME_AW(lstrlen)
1638 HINSTANCE WINAPI WinExec(LPCSTR,UINT);
1639 LONG WINAPI _hread(HFILE,LPVOID,ULONG);
1640 LONG WINAPI _hwrite(HFILE,LPCSTR,ULONG);
1641 HFILE WINAPI _lcreat(LPCSTR,INT);
1642 HFILE WINAPI _lclose(HFILE);
1643 LONG WINAPI _llseek(HFILE,ULONG,INT);
1644 HFILE WINAPI _lopen(LPCSTR,INT);
1645 UINT WINAPI _lread(HFILE,LPVOID,UINT);
1646 UINT WINAPI _lwrite(HFILE,LPCSTR,UINT);
1647 INT WINAPI lstrcmpA(LPCSTR,LPCSTR);
1648 INT WINAPI lstrcmpW(LPCWSTR,LPCWSTR);
1649 #define lstrcmp WINELIB_NAME_AW(lstrcmp)
1650 INT WINAPI lstrcmpiA(LPCSTR,LPCSTR);
1651 INT WINAPI lstrcmpiW(LPCWSTR,LPCWSTR);
1652 #define lstrcmpi WINELIB_NAME_AW(lstrcmpi)
1653
1654 /* compatibility macros */
1655 #define FillMemory RtlFillMemory
1656 #define MoveMemory RtlMoveMemory
1657 #define ZeroMemory RtlZeroMemory
1658 #define CopyMemory RtlCopyMemory
1659
1660 /* undocumented functions */
1661
1662 typedef struct tagSYSLEVEL
1663 {
1664     CRITICAL_SECTION crst;
1665     INT level;
1666 } SYSLEVEL;
1667
1668 /* [GS]etProcessDword offsets */
1669 #define GPD_APP_COMPAT_FLAGS (-56)
1670 #define GPD_LOAD_DONE_EVENT (-52)

```

```

1671 #define GPD_HINSTANCE16 (-48)
1672 #define GPD_WINDOWS_VERSION (-44)
1673 #define GPD_THDB (-40)
1674 #define GPD_PDB (-36)
1675 #define GPD_STARTF_SHELLDATA (-32)
1676 #define GPD_STARTF_HOTKEY (-28)
1677 #define GPD_STARTF_SHOWWINDOW (-24)
1678 #define GPD_STARTF_SIZE (-20)
1679 #define GPD_STARTF_POSITION (-16)
1680 #define GPD_STARTF_FLAGS (-12)
1681 #define GPD_PARENT (- 8)
1682 #define GPD_FLAGS (- 4)
1683 #define GPD_USERDATA ( 0)
1684
1685 void WINAPI DisposeLZ32Handle(HANDLE);
1686 HANDLE WINAPI DosFileHandleToWin32Handle(HFILE);
1687 DWORD WINAPI GetProcessDword(DWORD, INT);
1688 VOID WINAPI GetpWin16Lock(SYSLEVEL*);
1689 DWORD WINAPI MapLS(LPCVOID);
1690 DWORD WINAPI MapProcessHandle(HANDLE);
1691 LPVOID WINAPI MapSL(DWORD);
1692 VOID WINAPI ReleaseThunkLock(DWORD*);
1693 VOID WINAPI RestoreThunkLock(DWORD);
1694 void WINAPI SetProcessDword(DWORD, INT, DWORD);
1695 VOID WINAPI UnMapLS(DWORD);
1696 HFILE WINAPI Win32HandleToDosFileHandle(HANDLE);
1697 VOID WINAPI _CheckNotSysLevel(SYSLEVEL *lock);
1698 DWORD WINAPI _ConfirmWin16Lock(void);
1699 DWORD WINAPI _ConfirmSysLevel(SYSLEVEL*);
1700 VOID WINAPI _EnterSysLevel(SYSLEVEL*);
1701 VOID WINAPI _LeaveSysLevel(SYSLEVEL*);
1702
1703
1704 /* Wine internal functions */
1705
1706 BOOL WINAPI wine_get_unix_file_name( LPCSTR dos, LPSTR buffer, DWORD len );
1707
1708
1709 /* a few optimizations for i386/gcc */
1710
1711 /*#if defined(__i386__) && defined(__GNUC__)*/
1712 /* Deleted these since the real WINE environment is not available */
1713 #if 0
1714
1715 extern inline LONG WINAPI InterlockedCompareExchange( PLONG dest, LONG xchg, LONG compare );
1716 extern inline LONG WINAPI InterlockedCompareExchange( PLONG dest, LONG xchg, LONG compare )
1717 {
1718     LONG ret;
1719     __asm__ __volatile__( "lock; cmpxchgl %2,%1"
1720 : "=a" (ret) : "r" (dest), "r" (xchg), "0" (compare) : "memory" );
1721     return ret;
1722 }
1723
1724 extern inline LONG WINAPI InterlockedExchange( PLONG dest, LONG val );
1725 extern inline LONG WINAPI InterlockedExchange( PLONG dest, LONG val )
1726 {
1727     LONG ret;
1728     __asm__ __volatile__( "lock; xchgl %0,%1"
1729 : "=r" (ret) : "r" (dest), "0" (val) : "memory" );
1730     return ret;
1731 }
1732
1733 extern inline LONG WINAPI InterlockedExchangeAdd( PLONG dest, LONG incr );
1734 extern inline LONG WINAPI InterlockedExchangeAdd( PLONG dest, LONG incr )
1735 {
1736     LONG ret;
1737     __asm__ __volatile__( "lock; xaddl %0,%1"
1738 : "=r" (ret) : "r" (dest), "0" (incr) : "memory" );
1739     return ret;
1740 }
1741
1742 extern inline LONG WINAPI InterlockedIncrement( PLONG dest );
1743 extern inline LONG WINAPI InterlockedIncrement( PLONG dest )
1744 {
1745     return InterlockedExchangeAdd( dest, 1 ) + 1;
1746 }
1747
1748 extern inline LONG WINAPI InterlockedDecrement( PLONG dest );
1749 extern inline LONG WINAPI InterlockedDecrement( PLONG dest )
1750 {
1751     return InterlockedExchangeAdd( dest, -1 ) - 1;
1752 }
1753
1754 extern inline DWORD WINAPI GetLastError(void);
1755 extern inline DWORD WINAPI GetLastError(void)
1756 {
1757     DWORD ret;

```



```

1758     __asm__ __volatile__( ".byte 0x64\n\tmovl 0x60,%0" : "=r" (ret) );
1759     return ret;
1760 }
1761
1762 extern inline DWORD WINAPI GetCurrentProcessId(void);
1763 extern inline DWORD WINAPI GetCurrentProcessId(void)
1764 {
1765     DWORD ret;
1766     __asm__ __volatile__( ".byte 0x64\n\tmovl 0x20,%0" : "=r" (ret) );
1767     return ret;
1768 }
1769
1770 extern inline DWORD WINAPI GetCurrentThreadId(void);
1771 extern inline DWORD WINAPI GetCurrentThreadId(void)
1772 {
1773     DWORD ret;
1774     __asm__ __volatile__( ".byte 0x64\n\tmovl 0x24,%0" : "=r" (ret) );
1775     return ret;
1776 }
1777
1778 extern inline void WINAPI SetLastError( DWORD err );
1779 extern inline void WINAPI SetLastError( DWORD err )
1780 {
1781     __asm__ __volatile__( ".byte 0x64\n\tmovl %0,0x60" : : "r" (err) : "memory" );
1782 }
1783
1784 extern inline HANDLE WINAPI GetProcessHeap(void);
1785 extern inline HANDLE WINAPI GetProcessHeap(void)
1786 {
1787     HANDLE *pdb;
1788     __asm__ __volatile__( ".byte 0x64\n\tmovl 0x30,%0" : "=r" (pdb) );
1789     return pdb[0x18 / sizeof(HANDLE)]; /* get dword at offset 0x18 in pdb */
1790 }
1791
1792 #else /* __i386__ && __GNUC__ */
1793 DWORD WINAPI GetCurrentProcessId(void);
1794 DWORD WINAPI GetCurrentThreadId(void);
1795 DWORD WINAPI GetLastError(void);
1796 HANDLE WINAPI GetProcessHeap(void);
1797 LONG WINAPI InterlockedCompareExchange(LONG*, LONG, LONG);
1798 LONG WINAPI InterlockedDecrement(PLONG);
1799 LONG WINAPI InterlockedExchange(PLONG, LONG);
1800 LONG WINAPI InterlockedExchangeAdd(PLONG, LONG);
1801 LONG WINAPI InterlockedIncrement(PLONG);
1802 VOID WINAPI SetLastError(DWORD);
1803 #endif /* __i386__ && __GNUC__ */
1804
1805 /* FIXME: should handle platforms where sizeof(void*) != sizeof(long) */
1806 #if 0
1807 /* Unused in libEMF */
1808 static inline PVOID WINAPI InterlockedCompareExchangePointer( PVOID *dest, PVOID xchg, PVOID compare )
1809 {
1810     return (PVOID)InterlockedCompareExchange( (PLONG)dest, (LONG)xchg, (LONG)compare );
1811 }
1812
1813 static inline PVOID WINAPI InterlockedExchangePointer( PVOID *dest, PVOID val )
1814 {
1815     return (PVOID)InterlockedExchange( (PLONG)dest, (LONG)val );
1816 }
1817 #endif
1818 #ifdef __WINE__
1819 #define GetCurrentProcess() ((HANDLE)0xffffffff)
1820 #define GetCurrentThread() ((HANDLE)0xfffffffffe)
1821 #endif
1822
1823 /* WinMain(entry point) must be declared in winbase.h. */
1824 /* If this is not declared, we cannot compile many sources written with C++. */
1825 int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int);
1826
1827 #ifdef __cplusplus
1828 }
1829 #endif
1830
1831 #endif /* __WINE_WINBASE_H */

```

5.9 windef.h

```

1 /*
2  * Basic types definitions
3  *
4  * Copyright 1996 Alexandre Julliard
5  */
6
7 #ifndef __WINE_WINDEF_H

```

```

8 #define __WINE_WINDEF_H
9
10 #ifdef __WINE__
11 # undef UNICODE
12 #endif /* __WINE__ */
13
14 #define WINVER 0x0500
15
16 #include "winnt.h"
17
18
19 #ifdef __cplusplus
20 extern "C" {
21 #endif
22
23
24 /* Macros to map Winelib names to the correct implementation name */
25 /* depending on __WINE__ and UNICODE macros. */
26 /* Note that Winelib is purely Win32. */
27
28 #ifdef __WINE__
29 # define WINELIB_NAME_AW(func) \
30 func##_must_be_suffixed_with_W_or_A_in_this_context \
31 func##_must_be_suffixed_with_W_or_A_in_this_context
32 #else /* __WINE__ */
33 # ifdef UNICODE
34 #  define WINELIB_NAME_AW(func) func##W
35 # else
36 #  define WINELIB_NAME_AW(func) func##A
37 # endif /* UNICODE */
38 #endif /* __WINE__ */
39
40 #ifdef __WINE__
41 # define DECL_WINELIB_TYPE_AW(type) /* nothing */
42 #else /* __WINE__ */
43 # define DECL_WINELIB_TYPE_AW(type) typedef WINELIB_NAME_AW(type) type;
44 #endif /* __WINE__ */
45
46
47 /* Integer types */
48 typedef uint      WPARAM;
49 typedef long      LPARAM;
50 typedef long      LRESULT;
51 typedef word      ATOM;
52 typedef word      CATCHBUF[9];
53 typedef word      *LPCATCHBUF;
54 typedef dword     COLORREF, *LPCOLORREF;
55
56
57 /* Handle types that exist both in Win16 and Win32. */
58
59 typedef int HFILE;
60 DECLARE_OLD_HANDLE(HACCEL);
61 DECLARE_OLD_HANDLE(HBITMAP);
62 DECLARE_OLD_HANDLE(HBRUSH);
63 DECLARE_HANDLE(HCOLORSPACE);
64 DECLARE_OLD_HANDLE(HDC);
65 DECLARE_HANDLE(HDESK);
66 DECLARE_OLD_HANDLE(HENHMETAFILE);
67 DECLARE_OLD_HANDLE(HFONT);
68 DECLARE_OLD_HANDLE(HHOOK);
69 DECLARE_OLD_HANDLE(HICON);
70 DECLARE_OLD_HANDLE(HINSTANCE);
71 DECLARE_OLD_HANDLE(HKEY);
72 DECLARE_OLD_HANDLE(HKL);
73 DECLARE_OLD_HANDLE(HMENU);
74 DECLARE_OLD_HANDLE(HMETAFILE);
75 DECLARE_OLD_HANDLE(HMONITOR);
76 DECLARE_OLD_HANDLE(HPALETTE);
77 DECLARE_OLD_HANDLE(HPEN);
78 DECLARE_OLD_HANDLE(HRGN);
79 DECLARE_OLD_HANDLE(HRSRC);
80 DECLARE_OLD_HANDLE(HTASK);
81 DECLARE_HANDLE(HWINSTA);
82 DECLARE_OLD_HANDLE(HWND);
83
84 /* Handle types that must remain interchangeable even with strict on */
85
86 typedef HINSTANCE HMODULE;
87 typedef HANDLE HGDIOBJ;
88 typedef HANDLE HGLOBAL;
89 typedef HANDLE HLOCAL;
90 typedef HANDLE GLOBALHANDLE;
91 typedef HANDLE LOCALHANDLE;
92 typedef HICON HCURSOR;
93
94 /* Callback function pointers types */

```



```

95
96 typedef INT      CALLBACK (*FARPROC)();
97 typedef INT      CALLBACK (*PROC)();
98
99
100 /* Macros to split words and longs. */
101
102 #define LOBYTE(w)      ((BYTE)(WORD)(w))
103 #define HIBYTE(w)      ((BYTE)((WORD)(w) >> 8))
104
105 #define LOWORD(l)      ((WORD)(DWORD)(l))
106 #define HIWORD(l)      ((WORD)((DWORD)(l) >> 16))
107
108 #define SLOWORD(l)      ((SHORT)(LONG)(l))
109 #define SHIWORD(l)      ((SHORT)((LONG)(l) >> 16))
110
111 #define MAKEWORD(low,high) ((WORD)((BYTE)(low) | ((WORD)((BYTE)(high))) << 8))
112 #define MAKELONG(low,high) ((LONG)((WORD)(low) | ((DWORD)((WORD)(high))) << 16))
113 #define MAKELPARAM(low,high) ((LPARAM)MAKELONG(low,high))
114 #define MAKEWPARAM(low,high) ((WPARAM)MAKELONG(low,high))
115 #define MAKELRESULT(low,high) ((LRESULT)MAKELONG(low,high))
116
117 #define SELECTOROF(ptr) (HIWORD(ptr))
118 #define OFFSETOF(ptr) (LOWORD(ptr))
119
120 #ifdef __WINE__
121 /* macros to set parts of a DWORD (not in the Windows API) */
122 #define SET_LOWORD(dw,val) ((dw) = ((dw) & 0xffff0000) | LOWORD(val))
123 #define SET_LOBYTE(dw,val) ((dw) = ((dw) & 0xffffffff00) | LOBYTE(val))
124 #define SET_HIBYTE(dw,val) ((dw) = ((dw) & 0xffff00ff) | (LOBYTE(val) << 8))
125 #define ADD_LOWORD(dw,val) ((dw) = ((dw) & 0xffff0000) | LOWORD((DWORD)(dw)+(val)))
126 #endif
127
128 /* min and max macros */
129 #ifndef NOMINMAX
130 #ifndef max
131 #define max(a,b)      (((a) > (b)) ? (a) : (b))
132 #endif
133 #ifndef min
134 #define min(a,b)      (((a) < (b)) ? (a) : (b))
135 #endif
136 #endif /* NOMINMAX */
137
138 #ifndef _MAX_PATH
139 /* FIXME: These are supposed to be in stdlib.h only */
140 #define _MAX_DRIVE      3
141 #define _MAX_FNAME      256
142 #define _MAX_DIR        _MAX_FNAME
143 #define _MAX_EXT        _MAX_FNAME
144 #define _MAX_PATH      260
145 #endif
146 #define MAX_PATH        _MAX_PATH
147
148
149 #define HFILE_ERROR      ((HFILE)-1)
150
151 /* The SIZE structure */
152 typedef struct tagSIZE
153 {
154     LONG cx;
155     LONG cy;
156 } SIZE, *PSIZE, *LPSIZE;
157
158 typedef SIZE SIZEL, *PSIZEL, *LPSIZEL;
159
160 /* The POINT structure */
161 typedef struct tagPOINT
162 {
163     LONG x;
164     LONG y;
165 } POINT, *PPOINT, *LPPOINT;
166
167 typedef struct _POINTL
168 {
169     LONG x;
170     LONG y;
171 } POINTL;
172
173 /* The POINTS structure */
174
175 typedef struct tagPOINTS
176 {
177     SHORT x;
178     SHORT y;
179 } POINTS, *PPOINTS, *LPPOINTS;
180
181 /* The RECT structure */

```

```

182 typedef struct tagRECT
183 {
184     INT    left;
185     INT    top;
186     INT    right;
187     INT    bottom;
188 } RECT, *PRECT, *LPRECT;
189 typedef const RECT *LPCRECT;
190
191
192 typedef struct tagRECTL
193 {
194     LONG left;
195     LONG top;
196     LONG right;
197     LONG bottom;
198 } RECTL, *PRECTL, *LPRECTL;
199
200 typedef const RECTL *LPCRECTL;
201
202 #ifdef __cplusplus
203 }
204 #endif
205
206 #endif /* __WINE_WINDEF_H */

```

5.10 winerror.h

```

1 #ifndef __WINE_WINERROR_H
2 #define __WINE_WINERROR_H
3
4
5 extern int WIN32_LastError;
6
7 #define FACILITY_NULL      0
8 #define FACILITY_RPC      1
9 #define FACILITY_DISPATCH 2
10 #define FACILITY_STORAGE  3
11 #define FACILITY_ITF      4
12 #define FACILITY_WIN32    7
13 #define FACILITY_WINDOWS  8
14 #define FACILITY_SSPI     9
15 #define FACILITY_CONTROL  10
16 #define FACILITY_CERT     11
17 #define FACILITY_INTERNET 12
18
19 #define SEVERITY_SUCCESS   0
20 #define SEVERITY_ERROR    1
21
22
23 #define MAKE_HRESULT(sev,fac,code) \
24 ((HRESULT) (((unsigned long)(sev)<<31) | ((unsigned long)(fac)<<16) | ((unsigned long)(code)))) )
25 #define MAKE_SCODE(sev,fac,code) \
26 ((SCODE) (((unsigned long)(sev)<<31) | ((unsigned long)(fac)<<16) | ((unsigned long)(code)))) )
27 #define SUCCEEDED(stat) ((HRESULT)(stat)>=0)
28 #define FAILED(stat) ((HRESULT)(stat)<0)
29 #define IS_ERROR(stat) (((unsigned long)(stat)>>31) == SEVERITY_ERROR)
30
31 #define HRESULT_CODE(hr) ((hr) & 0xFFFF)
32 #define SCODE_CODE(sc) ((sc) & 0xFFFF)
33
34 #define HRESULT_FACILITY(hr) (((hr) >> 16) & 0x1FFF)
35 #define SCODE_FACILITY(sc) (((sc) >> 16) & 0x1FFF)
36
37 #define HRESULT_SEVERITY(hr) (((hr) >> 31) & 0x1)
38 #define SCODE_SEVERITY(sc) (((sc) >> 31) & 0x1)
39
40 #define FACILITY_NT_BIT      0x10000000
41 #define HRESULT_FROM_WIN32(x) ((x) ? ((HRESULT) (((x) & 0x0000FFFF) | (FACILITY_WIN32 << 16) | 0x80000000)) : 0)
42 #define HRESULT_FROM_NT(x) ((HRESULT) ((x) | FACILITY_NT_BIT))
43
44 /* SCODE <-> HRESULT functions */
45 /* This macros is obsolete and should not be used in new apps. */
46 #define GetSCode(hr) ((SCODE)(hr))
47 /* This macros is obsolete and should not be used in new apps. */
48 #define ResultFromSCode(sc) ((HRESULT)(sc))
49
50 /* ERROR_UNKNOWN is a placeholder for error conditions which haven't
51 * been tested yet so we're not exactly sure what will be returned.
52 * All instances of ERROR_UNKNOWN should be tested under Win95/NT
53 * and replaced.
54 */
55 #define ERROR_UNKNOWN          99999

```

```
56
57 #define NO_ERROR 0
58 #define ERROR_SUCCESS 0
59 #define ERROR_INVALID_FUNCTION 1
60 #define ERROR_FILE_NOT_FOUND 2
61 #define ERROR_PATH_NOT_FOUND 3
62 #define ERROR_TOO_MANY_OPEN_FILES 4
63 #define ERROR_ACCESS_DENIED 5
64 #define ERROR_INVALID_HANDLE 6
65 #define ERROR_ARENA_TRASHED 7
66 #define ERROR_NOT_ENOUGH_MEMORY 8
67 #define ERROR_INVALID_BLOCK 9
68 #define ERROR_BAD_ENVIRONMENT 10
69 #define ERROR_BAD_FORMAT 11
70 #define ERROR_INVALID_ACCESS 12
71 #define ERROR_INVALID_DATA 13
72 #define ERROR_OUTOFMEMORY 14
73 #define ERROR_INVALID_DRIVE 15
74 #define ERROR_CURRENT_DIRECTORY 16
75 #define ERROR_NOT_SAME_DEVICE 17
76 #define ERROR_NO_MORE_FILES 18
77 #define ERROR_WRITE_PROTECT 19
78 #define ERROR_BAD_UNIT 20
79 #define ERROR_NOT_READY 21
80 #define ERROR_BAD_COMMAND 22
81 #define ERROR_CRC 23
82 #define ERROR_BAD_LENGTH 24
83 #define ERROR_SEEK 25
84 #define ERROR_NOT_DOS_DISK 26
85 #define ERROR_SECTOR_NOT_FOUND 27
86 #define ERROR_OUT_OF_PAPER 28
87 #define ERROR_WRITE_FAULT 29
88 #define ERROR_READ_FAULT 30
89 #define ERROR_GEN_FAILURE 31
90 #define ERROR_SHARING_VIOLATION 32
91 #define ERROR_LOCK_VIOLATION 33
92 #define ERROR_WRONG_DISK 34
93 /* FIXME: 35 gets returned for some unsuccessful DeviceIoControl calls */
94 #define ERROR_UNKNOWN_NAME_01 35
95 #define ERROR_SHARING_BUFFER_EXCEEDED 36
96 #define ERROR_HANDLE_EOF 38
97 #define ERROR_HANDLE_DISK_FULL 39
98 #define ERROR_NOT_SUPPORTED 50
99 #define ERROR_REM_NOT_LIST 51
100 #define ERROR_DUP_NAME 52
101 #define ERROR_BAD_NETPATH 53
102 #define ERROR_NETWORK_BUSY 54
103 #define ERROR_DEV_NOT_EXIST 55
104 #define ERROR_TOO_MANY_CMDS 56
105 #define ERROR_ADAP_HDW_ERR 57
106 #define ERROR_BAD_NET_RESP 58
107 #define ERROR_UNEXP_NET_ERR 59
108 #define ERROR_BAD_REM_ADAP 60
109 #define ERROR_PRINTQ_FULL 61
110 #define ERROR_NO_SPOOL_SPACE 62
111 #define ERROR_PRINT_CANCELLED 63
112 #define ERROR_NETNAME_DELETED 64
113 #define ERROR_NETWORK_ACCESS_DENIED 65
114 #define ERROR_BAD_DEV_TYPE 66
115 #define ERROR_BAD_NET_NAME 67
116 #define ERROR_TOO_MANY_NAMES 68
117 #define ERROR_TOO_MANY_SESS 69
118 #define ERROR_SHARING_PAUSED 70
119 #define ERROR_REQ_NOT_ACCEP 71
120 #define ERROR_REDIR_PAUSED 72
121 #define ERROR_FILE_EXISTS 80
122 #define ERROR_CANNOT_MAKE 82
123 #define ERROR_FAIL_I24 83
124 #define ERROR_OUT_OF_STRUCTURES 84
125 #define ERROR_ALREADY_ASSIGNED 85
126 #define ERROR_INVALID_PASSWORD 86
127 #define ERROR_INVALID_PARAMETER 87
128 #define ERROR_NET_WRITE_FAULT 88
129 #define ERROR_NO_PROC_SLOTS 89
130 #define ERROR_TOO_MANY_SEMAPHORES 100
131 #define ERROR_EXCL_SEM_ALREADY_OWNED 101
132 #define ERROR_SEM_IS_SET 102
133 #define ERROR_TOO_MANY_SEM_REQUESTS 103
134 #define ERROR_INVALID_AT_INTERRUPT_TIME 104
135 #define ERROR_SEM_OWNER_DIED 105
136 #define ERROR_SEM_USER_LIMIT 106
137 #define ERROR_DISK_CHANGE 107
138 #define ERROR_DRIVE_LOCKED 108
139 #define ERROR_BROKEN_PIPE 109
140 #define ERROR_OPEN_FAILED 110
141 #define ERROR_BUFFER_OVERFLOW 111
142 #define ERROR_DISK_FULL 112
```

143	#define	ERROR_NO_MORE_SEARCH_HANDLES	113
144	#define	ERROR_INVALID_TARGET_HANDLE	114
145	#define	ERROR_INVALID_CATEGORY	117
146	#define	ERROR_INVALID_VERIFY_SWITCH	118
147	#define	ERROR_BAD_DRIVER_LEVEL	119
148	#define	ERROR_CALL_NOT_IMPLEMENTED	120
149	#define	ERROR_SEM_TIMEOUT	121
150	#define	ERROR_INSUFFICIENT_BUFFER	122
151	#define	ERROR_INVALID_NAME	123
152	#define	ERROR_INVALID_LEVEL	124
153	#define	ERROR_NO_VOLUME_LABEL	125
154	#define	ERROR_MOD_NOT_FOUND	126
155	#define	ERROR_PROC_NOT_FOUND	127
156	#define	ERROR_WAIT_NO_CHILDREN	128
157	#define	ERROR_CHILD_NOT_COMPLETE	129
158	#define	ERROR_DIRECT_ACCESS_HANDLE	130
159	#define	ERROR_NEGATIVE_SEEK	131
160	#define	ERROR_SEEK_ON_DEVICE	132
161	#define	ERROR_IS_JOIN_TARGET	133
162	#define	ERROR_IS_JOINED	134
163	#define	ERROR_IS_SUBSTED	135
164	#define	ERROR_NOT_JOINED	136
165	#define	ERROR_NOT_SUBSTED	137
166	#define	ERROR_JOIN_TO_JOIN	138
167	#define	ERROR_SUBST_TO_SUBST	139
168	#define	ERROR_JOIN_TO_SUBST	140
169	#define	ERROR_SUBST_TO_JOIN	141
170	#define	ERROR_BUSY_DRIVE	142
171	#define	ERROR_SAME_DRIVE	143
172	#define	ERROR_DIR_NOT_ROOT	144
173	#define	ERROR_DIR_NOT_EMPTY	145
174	#define	ERROR_IS_SUBST_PATH	146
175	#define	ERROR_IS_JOIN_PATH	147
176	#define	ERROR_PATH_BUSY	148
177	#define	ERROR_IS_SUBST_TARGET	149
178	#define	ERROR_SYSTEM_TRACE	150
179	#define	ERROR_INVALID_EVENT_COUNT	151
180	#define	ERROR_TOO_MANY_MUXWAITERS	152
181	#define	ERROR_INVALID_LIST_FORMAT	153
182	#define	ERROR_LABEL_TOO_LONG	154
183	#define	ERROR_TOO_MANY_TCBS	155
184	#define	ERROR_SIGNAL_REFUSED	156
185	#define	ERROR_DISCARDED	157
186	#define	ERROR_NOT_LOCKED	158
187	#define	ERROR_BAD_THREADID_ADDR	159
188	#define	ERROR_BAD_ARGUMENTS	160
189	#define	ERROR_BAD_PATHNAME	161
190	#define	ERROR_SIGNAL_PENDING	162
191	#define	ERROR_MAX_THRDS_REACHED	164
192	#define	ERROR_LOCK_FAILED	167
193	#define	ERROR_BUSY	170
194	#define	ERROR_CANCEL_VIOLATION	173
195	#define	ERROR_ATOMIC_LOCKS_NOT_SUPPORTED	174
196	#define	ERROR_INVALID_SEGMENT_NUMBER	180
197	#define	ERROR_INVALID_ORDINAL	182
198	#define	ERROR_ALREADY_EXISTS	183
199	#define	ERROR_INVALID_FLAG_NUMBER	186
200	#define	ERROR_SEM_NOT_FOUND	187
201	#define	ERROR_INVALID_STARTING_CODESEG	188
202	#define	ERROR_INVALID_STACKSEG	189
203	#define	ERROR_INVALID_MODULETYPE	190
204	#define	ERROR_INVALID_EXE_SIGNATURE	191
205	#define	ERROR_EXE_MARKED_INVALID	192
206	#define	ERROR_BAD_EXE_FORMAT	193
207	#define	ERROR_ITERATED_DATA_EXCEEDS_64k	194
208	#define	ERROR_INVALID_MINALLOCSIZE	195
209	#define	ERROR_DYNLINK_FROM_INVALID_RING	196
210	#define	ERROR_IOPL_NOT_ENABLED	197
211	#define	ERROR_INVALID_SEGDPL	198
212	#define	ERROR_AUTODATASEG_EXCEEDS_64k	199
213	#define	ERROR_RING2SEG_MUST_BE_MOVABLE	200
214	#define	ERROR_RELOC_CHAIN_XEEDS_SEGLIM	201
215	#define	ERROR_INFLOOP_IN_RELOC_CHAIN	202
216	#define	ERROR_ENVVAR_NOT_FOUND	203
217	#define	ERROR_NO_SIGNAL_SENT	205
218	#define	ERROR_FILENAME_EXCED_RANGE	206
219	#define	ERROR_RING2_STACK_IN_USE	207
220	#define	ERROR_META_EXPANSION_TOO_LONG	208
221	#define	ERROR_INVALID_SIGNAL_NUMBER	209
222	#define	ERROR_THREAD_1_INACTIVE	210
223	#define	ERROR_LOCKED	212
224	#define	ERROR_TOO_MANY_MODULES	214
225	#define	ERROR_NESTING_NOT_ALLOWED	215
226	#define	ERROR_EXE_MACHINE_TYPE_MISMATCH	216
227	#define	ERROR_BAD_PIPE	230
228	#define	ERROR_PIPE_BUSY	231
229	#define	ERROR_NO_DATA	232

230	#define	ERROR_PIPE_NOT_CONNECTED	233
231	#define	ERROR_MORE_DATA	234
232	#define	ERROR_VC_DISCONNECTED	240
233	#define	ERROR_INVALID_EA_NAME	254
234	#define	ERROR_EA_LIST_INCONSISTENT	255
235	#define	ERROR_NO_MORE_ITEMS	259
236	#define	ERROR_CANNOT_COPY	266
237	#define	ERROR_DIRECTORY	267
238	#define	ERROR_EAS_DIDNT_FIT	275
239	#define	ERROR_EA_FILE_CORRUPT	276
240	#define	ERROR_EA_TABLE_FULL	277
241	#define	ERROR_INVALID_EA_HANDLE	278
242	#define	ERROR_EAS_NOT_SUPPORTED	282
243	#define	ERROR_NOT_OWNER	288
244	#define	ERROR_TOO_MANY_POSTS	298
245	#define	ERROR_PARTIAL_COPY	299
246	#define	ERROR_OPLOCK_NOT_GRANTED	300
247	#define	ERROR_INVALID_OPLOCK_PROTOCOL	301
248	#define	ERROR_MR_MID_NOT_FOUND	317
249	#define	ERROR_INVALID_ADDRESS	487
250	#define	ERROR_ARITHMETIC_OVERFLOW	534
251	#define	ERROR_PIPE_CONNECTED	535
252	#define	ERROR_PIPE_LISTENING	536
253	#define	ERROR_EA_ACCESS_DENIED	994
254	#define	ERROR_OPERATION_ABORTED	995
255	#define	ERROR_IO_INCOMPLETE	996
256	#define	ERROR_IO_PENDING	997
257	#define	ERROR_NOACCESS	998
258	#define	ERROR_SWAPERROR	999
259	#define	ERROR_STACK_OVERFLOW	1001
260	#define	ERROR_INVALID_MESSAGE	1002
261	#define	ERROR_CAN_NOT_COMPLETE	1003
262	#define	ERROR_INVALID_FLAGS	1004
263	#define	ERROR_UNRECOGNIZED_VOLUME	1005
264	#define	ERROR_FILE_INVALID	1006
265	#define	ERROR_FULLSCREEN_MODE	1007
266	#define	ERROR_NO_TOKEN	1008
267	#define	ERROR_BADDB	1009
268	#define	ERROR_BADKEY	1010
269	#define	ERROR_CANTOPEN	1011
270	#define	ERROR_CANTREAD	1012
271	#define	ERROR_CANTWRITE	1013
272	#define	ERROR_REGISTRY_RECOVERED	1014
273	#define	ERROR_REGISTRY_CORRUPT	1015
274	#define	ERROR_REGISTRY_IO_FAILED	1016
275	#define	ERROR_NOT_REGISTRY_FILE	1017
276	#define	ERROR_KEY_DELETED	1018
277	#define	ERROR_NO_LOG_SPACE	1019
278	#define	ERROR_KEY_HAS_CHILDREN	1020
279	#define	ERROR_CHILD_MUST_BE_VOLATILE	1021
280	#define	ERROR_NOTIFY_ENUM_DIR	1022
281	#define	ERROR_DEPENDENT_SERVICES_RUNNING	1051
282	#define	ERROR_INVALID_SERVICE_CONTROL	1052
283	#define	ERROR_SERVICE_REQUEST_TIMEOUT	1053
284	#define	ERROR_SERVICE_NO_THREAD	1054
285	#define	ERROR_SERVICE_DATABASE_LOCKED	1055
286	#define	ERROR_SERVICE_ALREADY_RUNNING	1056
287	#define	ERROR_INVALID_SERVICE_ACCOUNT	1057
288	#define	ERROR_SERVICE_DISABLED	1058
289	#define	ERROR_CIRCULAR_DEPENDENCY	1059
290	#define	ERROR_SERVICE_DOES_NOT_EXIST	1060
291	#define	ERROR_SERVICE_CANNOT_ACCEPT_CTRL	1061
292	#define	ERROR_SERVICE_NOT_ACTIVE	1062
293	#define	ERROR_FAILED_SERVICE_CONTROLLER_CONNECT	1063
294	#define	ERROR_EXCEPTION_IN_SERVICE	1064
295	#define	ERROR_DATABASE_DOES_NOT_EXIST	1065
296	#define	ERROR_SERVICE_SPECIFIC_ERROR	1066
297	#define	ERROR_PROCESS_ABORTED	1067
298	#define	ERROR_SERVICE_DEPENDENCY_FAIL	1068
299	#define	ERROR_SERVICE_LOGON_FAILED	1069
300	#define	ERROR_SERVICE_START_HANG	1070
301	#define	ERROR_INVALID_SERVICE_LOCK	1071
302	#define	ERROR_SERVICE_MARKED_FOR_DELETE	1072
303	#define	ERROR_SERVICE_EXISTS	1073
304	#define	ERROR_ALREADY_RUNNING_LKG	1074
305	#define	ERROR_SERVICE_DEPENDENCY_DELETED	1075
306	#define	ERROR_BOOT_ALREADY_ACCEPTED	1076
307	#define	ERROR_SERVICE_NEVER_STARTED	1077
308	#define	ERROR_DUPLICATE_SERVICE_NAME	1078
309	#define	ERROR_DIFFERENT_SERVICE_ACCOUNT	1079
310	#define	ERROR_CANNOT_DETECT_DRIVER_FAILURE	1080
311	#define	ERROR_CANNOT_DETECT_PROCESS_ABORT	1081
312	#define	ERROR_NO_RECOVERY_PROGRAM	1082
313	#define	ERROR_SERVICE_NOT_IN_EXE	1083
314	#define	ERROR_END_OF_MEDIA	1100
315	#define	ERROR_FILEMARK_DETECTED	1101
316	#define	ERROR_BEGINNING_OF_MEDIA	1102

```
317 #define ERROR_SETMARK_DETECTED 1103
318 #define ERROR_NO_DATA_DETECTED 1104
319 #define ERROR_PARTITION_FAILURE 1105
320 #define ERROR_INVALID_BLOCK_LENGTH 1106
321 #define ERROR_DEVICE_NOT_PARTITIONED 1107
322 #define ERROR_UNABLE_TO_LOCK_MEDIA 1108
323 #define ERROR_UNABLE_TO_UNLOAD_MEDIA 1109
324 #define ERROR_MEDIA_CHANGED 1110
325 #define ERROR_BUS_RESET 1111
326 #define ERROR_NO_MEDIA_IN_DRIVE 1112
327 #define ERROR_NO_UNICODE_TRANSLATION 1113
328 #define ERROR_DLL_INIT_FAILED 1114
329 #define ERROR_SHUTDOWN_IN_PROGRESS 1115
330 #define ERROR_NO_SHUTDOWN_IN_PROGRESS 1116
331 #define ERROR_IO_DEVICE 1117
332 #define ERROR_SERIAL_NO_DEVICE 1118
333 #define ERROR_IRQ_BUSY 1119
334 #define ERROR_MORE_WRITES 1120
335 #define ERROR_COUNTER_TIMEOUT 1121
336 #define ERROR_FLOPPY_ID_MARK_NOT_FOUND 1122
337 #define ERROR_FLOPPY_WRONG_CYLINDER 1123
338 #define ERROR_FLOPPY_UNKNOWN_ERROR 1124
339 #define ERROR_FLOPPY_BAD_REGISTERS 1125
340 #define ERROR_DISK_RECALIBRATE_FAILED 1126
341 #define ERROR_DISK_OPERATION_FAILED 1127
342 #define ERROR_DISK_RESET_FAILED 1128
343 #define ERROR_EOM_OVERFLOW 1129
344 #define ERROR_NOT_ENOUGH_SERVER_MEMORY 1130
345 #define ERROR_POSSIBLE_DEADLOCK 1131
346 #define ERROR_MAPPED_ALIGNMENT 1132
347 #define ERROR_SET_POWER_STATE_VETOED 1140
348 #define ERROR_SET_POWER_STATE_FAILED 1141
349 #define ERROR_TOO_MANY_LINKS 1142
350 #define ERROR_OLD_WIN_VERSION 1150
351 #define ERROR_APP_WRONG_OS 1151
352 #define ERROR_SINGLE_INSTANCE_APP 1152
353 #define ERROR_RMODE_APP 1153
354 #define ERROR_INVALID_DLL 1154
355 #define ERROR_NO_ASSOCIATION 1155
356 #define ERROR_DDE_FAIL 1156
357 #define ERROR_DLL_NOT_FOUND 1157
358 #define ERROR_NO_MORE_USER_HANDLES 1158
359 #define ERROR_MESSAGE_SYNC_ONLY 1159
360 #define ERROR_SOURCE_ELEMENT_EMPTY 1160
361 #define ERROR_DESTINATION_ELEMENT_FULL 1161
362 #define ERROR_ILLEGAL_ELEMENT_ADDRESS 1162
363 #define ERROR_MAGAZINE_NOT_PRESENT 1163
364 #define ERROR_DEVICE_REINITIALIZATION_NEEDED 1164
365 #define ERROR_DEVICE_REQUIRES_CLEANING 1165
366 #define ERROR_DEVICE_DOOR_OPEN 1166
367 #define ERROR_DEVICE_NOT_CONNECTED 1167
368 #define ERROR_NOT_FOUND 1168
369 #define ERROR_NO_MATCH 1169
370 #define ERROR_SET_NOT_FOUND 1170
371 #define ERROR_POINT_NOT_FOUND 1171
372 #define ERROR_NO_TRACKING_SERVICE 1172
373 #define ERROR_NO_VOLUME_ID 1173
374 #define ERROR_UNABLE_TO_REMOVE_REPLACED 1175
375 #define ERROR_UNABLE_TO_MOVE_REPLACEMENT 1176
376 #define ERROR_UNABLE_TO_MOVE_REPLACEMENT_2 1177
377 #define ERROR_JOURNAL_DELETE_IN_PROGRESS 1178
378 #define ERROR_JOURNAL_NOT_ACTIVE 1179
379 #define ERROR_POTENTIAL_FILE_FOUND 1180
380 #define ERROR_JOURNAL_ENTRY_DELETED 1181
381 #define ERROR_BAD_DEVICE 1200
382 #define ERROR_CONNECTION_UNAVAIL 1201
383 #define ERROR_DEVICE_ALREADY_REMEMBERED 1202
384 #define ERROR_NO_NET_OR_BAD_PATH 1203
385 #define ERROR_BAD_PROVIDER 1204
386 #define ERROR_CANNOT_OPEN_PROFILE 1205
387 #define ERROR_BAD_PROFILE 1206
388 #define ERROR_NOT_CONTAINER 1207
389 #define ERROR_EXTENDED_ERROR 1208
390 #define ERROR_INVALID_GROUPNAME 1209
391 #define ERROR_INVALID_COMPUTERNAME 1210
392 #define ERROR_INVALID_EVENTNAME 1211
393 #define ERROR_INVALID_DOMAINNAME 1212
394 #define ERROR_INVALID_SERVICENAME 1213
395 #define ERROR_INVALID_NETNAME 1214
396 #define ERROR_INVALID_SHARENAME 1215
397 #define ERROR_INVALID_PASSWORDNAME 1216
398 #define ERROR_INVALID_MESSAGE 1217
399 #define ERROR_INVALID_MESSAGEDEST 1218
400 #define ERROR_SESSION_CREDENTIAL_CONFLICT 1219
401 #define ERROR_REMOTE_SESSION_LIMIT_EXCEEDED 1220
402 #define ERROR_DUP_DOMAINNAME 1221
403 #define ERROR_NO_NETWORK 1222
```

404	#define	ERROR_CANCELLED	1223
405	#define	ERROR_USER_MAPPED_FILE	1224
406	#define	ERROR_CONNECTION_REFUSED	1225
407	#define	ERROR_GRACEFUL_DISCONNECT	1226
408	#define	ERROR_ADDRESS_ALREADY_ASSOCIATED	1227
409	#define	ERROR_ADDRESS_NOT_ASSOCIATED	1228
410	#define	ERROR_CONNECTION_INVALID	1229
411	#define	ERROR_CONNECTION_ACTIVE	1230
412	#define	ERROR_NETWORK_UNREACHABLE	1231
413	#define	ERROR_HOST_UNREACHABLE	1232
414	#define	ERROR_PROTOCOL_UNREACHABLE	1233
415	#define	ERROR_PORT_UNREACHABLE	1234
416	#define	ERROR_REQUEST_ABORTED	1235
417	#define	ERROR_CONNECTION_ABORTED	1236
418	#define	ERROR_RETRY	1237
419	#define	ERROR_CONNECTION_COUNT_LIMIT	1238
420	#define	ERROR_LOGIN_TIME_RESTRICTION	1239
421	#define	ERROR_LOGIN_WKSTA_RESTRICTION	1240
422	#define	ERROR_INCORRECT_ADDRESS	1241
423	#define	ERROR_ALREADY_REGISTERED	1242
424	#define	ERROR_SERVICE_NOT_FOUND	1243
425	#define	ERROR_NOT_AUTHENTICATED	1244
426	#define	ERROR_NOT_LOGGED_ON	1245
427	#define	ERROR_CONTINUE	1246
428	#define	ERROR_ALREADY_INITIALIZED	1247
429	#define	ERROR_NO_MORE_DEVICES	1248
430	#define	ERROR_NO_SUCH_SITE	1249
431	#define	ERROR_DOMAIN_CONTROLLER_EXISTS	1250
432	#define	ERROR_ONLY_IF_CONNECTED	1251
433	#define	ERROR_OVERRIDE_NOCHANGES	1252
434	#define	ERROR_BAD_USER_PROFILE	1253
435	#define	ERROR_NOT_SUPPORTED_ON_SBS	1254
436	#define	ERROR_NOT_ALL_ASSIGNED	1300
437	#define	ERROR_SOME_NOT_MAPPED	1301
438	#define	ERROR_NO_QUOTAS_FOR_ACCOUNT	1302
439	#define	ERROR_LOCAL_USER_SESSION_KEY	1303
440	#define	ERROR_NULL_LM_PASSWORD	1304
441	#define	ERROR_UNKNOWN_REVISION	1305
442	#define	ERROR_REVISION_MISMATCH	1306
443	#define	ERROR_INVALID_OWNER	1307
444	#define	ERROR_INVALID_PRIMARY_GROUP	1308
445	#define	ERROR_NO_IMPERSONATION_TOKEN	1309
446	#define	ERROR_CANT_DISABLE_MANDATORY	1310
447	#define	ERROR_NO_LOGON_SERVERS	1311
448	#define	ERROR_NO_SUCH_LOGON_SESSION	1312
449	#define	ERROR_NO_SUCH_PRIVILEGE	1313
450	#define	ERROR_PRIVILEGE_NOT_HELD	1314
451	#define	ERROR_INVALID_ACCOUNT_NAME	1315
452	#define	ERROR_USER_EXISTS	1316
453	#define	ERROR_NO_SUCH_USER	1317
454	#define	ERROR_GROUP_EXISTS	1318
455	#define	ERROR_NO_SUCH_GROUP	1319
456	#define	ERROR_MEMBER_IN_GROUP	1320
457	#define	ERROR_MEMBER_NOT_IN_GROUP	1321
458	#define	ERROR_LAST_ADMIN	1322
459	#define	ERROR_WRONG_PASSWORD	1323
460	#define	ERROR_ILL_FORMED_PASSWORD	1324
461	#define	ERROR_PASSWORD_RESTRICTION	1325
462	#define	ERROR_LOGON_FAILURE	1326
463	#define	ERROR_ACCOUNT_RESTRICTION	1327
464	#define	ERROR_INVALID_LOGON_HOURS	1328
465	#define	ERROR_INVALID_WORKSTATION	1329
466	#define	ERROR_PASSWORD_EXPIRED	1330
467	#define	ERROR_ACCOUNT_DISABLED	1331
468	#define	ERROR_NONE_MAPPED	1332
469	#define	ERROR_TOO_MANY_LUIDS_REQUESTED	1333
470	#define	ERROR_LUIDS_EXHAUSTED	1334
471	#define	ERROR_INVALID_SUB_AUTHORITY	1335
472	#define	ERROR_INVALID_ACL	1336
473	#define	ERROR_INVALID_SID	1337
474	#define	ERROR_INVALID_SECURITY_DESCR	1338
475	#define	ERROR_BAD_INHERITANCE_ACL	1340
476	#define	ERROR_SERVER_DISABLED	1341
477	#define	ERROR_SERVER_NOT_DISABLED	1342
478	#define	ERROR_INVALID_ID_AUTHORITY	1343
479	#define	ERROR_ALLOTTED_SPACE_EXCEEDED	1344
480	#define	ERROR_INVALID_GROUP_ATTRIBUTES	1345
481	#define	ERROR_BAD_IMPERSONATION_LEVEL	1346
482	#define	ERROR_CANT_OPEN_ANONYMOUS	1347
483	#define	ERROR_BAD_VALIDATION_CLASS	1348
484	#define	ERROR_BAD_TOKEN_TYPE	1349
485	#define	ERROR_NO_SECURITY_ON_OBJECT	1350
486	#define	ERROR_CANT_ACCESS_DOMAIN_INFO	1351
487	#define	ERROR_INVALID_SERVER_STATE	1352
488	#define	ERROR_INVALID_DOMAIN_STATE	1353
489	#define	ERROR_INVALID_DOMAIN_ROLE	1354
490	#define	ERROR_NO_SUCH_DOMAIN	1355


```
491 #define ERROR_DOMAIN_EXISTS 1356
492 #define ERROR_DOMAIN_LIMIT_EXCEEDED 1357
493 #define ERROR_INTERNAL_DB_CORRUPTION 1358
494 #define ERROR_INTERNAL_ERROR 1359
495 #define ERROR_GENERIC_NOT_MAPPED 1360
496 #define ERROR_BAD_DESCRIPTOR_FORMAT 1361
497 #define ERROR_NOT_LOGON_PROCESS 1362
498 #define ERROR_LOGON_SESSION_EXISTS 1363
499 #define ERROR_NO_SUCH_PACKAGE 1364
500 #define ERROR_BAD_LOGON_SESSION_STATE 1365
501 #define ERROR_LOGON_SESSION_COLLISION 1366
502 #define ERROR_INVALID_LOGON_TYPE 1367
503 #define ERROR_CANNOT_IMPERSONATE 1368
504 #define ERROR_RXACT_INVALID_STATE 1369
505 #define ERROR_RXACT_COMMIT_FAILURE 1370
506 #define ERROR_SPECIAL_ACCOUNT 1371
507 #define ERROR_SPECIAL_GROUP 1372
508 #define ERROR_SPECIAL_USER 1373
509 #define ERROR_MEMBERS_PRIMARY_GROUP 1374
510 #define ERROR_TOKEN_ALREADY_IN_USE 1375
511 #define ERROR_NO_SUCH_ALIAS 1376
512 #define ERROR_MEMBER_NOT_IN_ALIAS 1377
513 #define ERROR_MEMBER_IN_ALIAS 1378
514 #define ERROR_ALIAS_EXISTS 1379
515 #define ERROR_LOGON_NOT_GRANTED 1380
516 #define ERROR_TOO_MANY_SECRETS 1381
517 #define ERROR_SECRET_TOO_LONG 1382
518 #define ERROR_INTERNAL_DB_ERROR 1383
519 #define ERROR_TOO_MANY_CONTEXT_IDS 1384
520 #define ERROR_LOGON_TYPE_NOT_GRANTED 1385
521 #define ERROR_NT_CROSS_ENCRYPTION_REQUIRED 1386
522 #define ERROR_NO_SUCH_MEMBER 1387
523 #define ERROR_INVALID_MEMBER 1388
524 #define ERROR_TOO_MANY_SIDS 1389
525 #define ERROR_LM_CROSS_ENCRYPTION_REQUIRED 1390
526 #define ERROR_NO_INHERITANCE 1391
527 #define ERROR_FILE_CORRUPT 1392
528 #define ERROR_DISK_CORRUPT 1393
529 #define ERROR_NO_USER_SESSION_KEY 1394
530 #define ERROR_LICENSE_QUOTA_EXCEEDED 1395
531 #define ERROR_WRONG_TARGET_NAME 1396
532 #define ERROR_MUTUAL_AUTH_FAILED 1397
533 #define ERROR_TIME_SKEW 1398
534 #define ERROR_INVALID_WINDOW_HANDLE 1400
535 #define ERROR_INVALID_MENU_HANDLE 1401
536 #define ERROR_INVALID_CURSOR_HANDLE 1402
537 #define ERROR_INVALID_ACCEL_HANDLE 1403
538 #define ERROR_INVALID_HOOK_HANDLE 1404
539 #define ERROR_INVALID_DWP_HANDLE 1405
540 #define ERROR_TLW_WITH_WSCHILD 1406
541 #define ERROR_CANNOT_FIND_WND_CLASS 1407
542 #define ERROR_WINDOW_OF_OTHER_THREAD 1408
543 #define ERROR_HOTKEY_ALREADY_REGISTERED 1409
544 #define ERROR_CLASS_ALREADY_EXISTS 1410
545 #define ERROR_CLASS_DOES_NOT_EXIST 1411
546 #define ERROR_CLASS_HAS_WINDOWS 1412
547 #define ERROR_INVALID_INDEX 1413
548 #define ERROR_INVALID_ICON_HANDLE 1414
549 #define ERROR_PRIVATE_DIALOG_INDEX 1415
550 #define ERROR_LISTBOX_ID_NOT_FOUND 1416
551 #define ERROR_NO_WILDCARD_CHARACTERS 1417
552 #define ERROR_CLIPBOARD_NOT_OPEN 1418
553 #define ERROR_HOTKEY_NOT_REGISTERED 1419
554 #define ERROR_WINDOW_NOT_DIALOG 1420
555 #define ERROR_CONTROL_ID_NOT_FOUND 1421
556 #define ERROR_INVALID_COMBOBOX_MESSAGE 1422
557 #define ERROR_WINDOW_NOT_COMBOBOX 1423
558 #define ERROR_INVALID_EDIT_HEIGHT 1424
559 #define ERROR_DC_NOT_FOUND 1425
560 #define ERROR_INVALID_HOOK_FILTER 1426
561 #define ERROR_INVALID_FILTER_PROC 1427
562 #define ERROR_HOOK_NEEDS_HMOD 1428
563 #define ERROR_GLOBAL_ONLY_HOOK 1429
564 #define ERROR_JOURNAL_HOOK_SET 1430
565 #define ERROR_HOOK_NOT_INSTALLED 1431
566 #define ERROR_INVALID_LB_MESSAGE 1432
567 #define ERROR_SETCOUNT_ON_BAD_LB 1433
568 #define ERROR_LB_WITHOUT_TABSTOPS 1434
569 #define ERROR_DESTROY_OBJECT_OF_OTHER_THREAD 1435
570 #define ERROR_CHILD_WINDOW_MENU 1436
571 #define ERROR_NO_SYSTEM_MENU 1437
572 #define ERROR_INVALID_MSGBOX_STYLE 1438
573 #define ERROR_INVALID_SPI_VALUE 1439
574 #define ERROR_SCREEN_ALREADY_LOCKED 1440
575 #define ERROR_HWNDS_HAVE_DIFF_PARENT 1441
576 #define ERROR_NOT_CHILD_WINDOW 1442
577 #define ERROR_INVALID_GW_COMMAND 1443
```



```
578 #define ERROR_INVALID_THREAD_ID 1444
579 #define ERROR_NON_MDICHILD_WINDOW 1445
580 #define ERROR_POPUP_ALREADY_ACTIVE 1446
581 #define ERROR_NO_SCROLLBARS 1447
582 #define ERROR_INVALID_SCROLLBAR_RANGE 1448
583 #define ERROR_INVALID_SHOWWIN_COMMAND 1449
584 #define ERROR_NO_SYSTEM_RESOURCES 1450
585 #define ERROR_NONPAGED_SYSTEM_RESOURCES 1451
586 #define ERROR_PAGED_SYSTEM_RESOURCES 1452
587 #define ERROR_WORKING_SET_QUOTA 1453
588 #define ERROR_PAGEFILE_QUOTA 1454
589 #define ERROR_COMMITMENT_LIMIT 1455
590 #define ERROR_MENU_ITEM_NOT_FOUND 1456
591 #define ERROR_INVALID_KEYBOARD_HANDLE 1457
592 #define ERROR_HOOK_TYPE_NOT_ALLOWED 1458
593 #define ERROR_REQUIRES_INTERACTIVE_WINDOWSTATION 1459
594 #define ERROR_TIMEOUT 1460
595 #define ERROR_INVALID_MONITOR_HANDLE 1461
596 #define ERROR_EVENTLOG_FILE_CORRUPT 1500
597 #define ERROR_EVENTLOG_CANT_START 1501
598 #define ERROR_LOG_FILE_FULL 1502
599 #define ERROR_EVENTLOG_FILE_CHANGED 1503
600 #define ERROR_INSTALL_SERVICE_FAILURE 1601
601 #define ERROR_INSTALL_USEREXIT 1602
602 #define ERROR_INSTALL_FAILURE 1603
603 #define ERROR_INSTALL_SUSPEND 1604
604 #define ERROR_UNKNOWN_PRODUCT 1605
605 #define ERROR_UNKNOWN_FEATURE 1606
606 #define ERROR_UNKNOWN_COMPONENT 1607
607 #define ERROR_UNKNOWN_PROPERTY 1608
608 #define ERROR_INVALID_HANDLE_STATE 1609
609 #define ERROR_BAD_CONFIGURATION 1610
610 #define ERROR_INDEX_ABSENT 1611
611 #define ERROR_INSTALL_SOURCE_ABSENT 1612
612 #define ERROR_INSTALL_PACKAGE_VERSION 1613
613 #define ERROR_PRODUCT_UNINSTALLED 1614
614 #define ERROR_BAD_QUERY_SYNTAX 1615
615 #define ERROR_INVALID_FIELD 1616
616 #define ERROR_DEVICE_REMOVED 1617
617 #define ERROR_INSTALL_ALREADY_RUNNING 1618
618 #define ERROR_INSTALL_PACKAGE_OPEN_FAILED 1619
619 #define ERROR_INSTALL_PACKAGE_INVALID 1620
620 #define ERROR_INSTALL_UI_FAILURE 1621
621 #define ERROR_INSTALL_LOG_FAILURE 1622
622 #define ERROR_INSTALL_LANGUAGE_UNSUPPORTED 1623
623 #define ERROR_INSTALL_TRANSFORM_FAILURE 1624
624 #define ERROR_INSTALL_PACKAGE_REJECTED 1625
625 #define ERROR_FUNCTION_NOT_CALLED 1626
626 #define ERROR_FUNCTION_FAILED 1627
627 #define ERROR_INVALID_TABLE 1628
628 #define ERROR_DATATYPE_MISMATCH 1629
629 #define ERROR_UNSUPPORTED_TYPE 1630
630 #define ERROR_CREATE_FAILED 1631
631 #define ERROR_INSTALL_TEMP_UNWRITABLE 1632
632 #define ERROR_INSTALL_PLATFORM_UNSUPPORTED 1633
633 #define ERROR_INSTALL_NOTUSED 1634
634 #define ERROR_PATCH_PACKAGE_OPEN_FAILED 1635
635 #define ERROR_PATCH_PACKAGE_INVALID 1636
636 #define ERROR_PATCH_PACKAGE_UNSUPPORTED 1637
637 #define ERROR_PRODUCT_VERSION 1638
638 #define ERROR_INVALID_COMMAND_LINE 1639
639 #define ERROR_INSTALL_REMOTE_DISALLOWED 1640
640 #define ERROR_SUCCESS_REBOOT_INITIATED 1641
641 #define RPC_S_INVALID_STRING_BINDING 1700
642 #define RPC_S_WRONG_KIND_OF_BINDING 1701
643 #define RPC_S_INVALID_BINDING 1702
644 #define RPC_S_PROTSEQ_NOT_SUPPORTED 1703
645 #define RPC_S_INVALID_RPC_PROTSEQ 1704
646 #define RPC_S_INVALID_STRING_UUID 1705
647 #define RPC_S_INVALID_ENDPOINT_FORMAT 1706
648 #define RPC_S_INVALID_NET_ADDR 1707
649 #define RPC_S_NO_ENDPOINT_FOUND 1708
650 #define RPC_S_INVALID_TIMEOUT 1709
651 #define RPC_S_OBJECT_NOT_FOUND 1710
652 #define RPC_S_ALREADY_REGISTERED 1711
653 #define RPC_S_TYPE_ALREADY_REGISTERED 1712
654 #define RPC_S_ALREADY_LISTENING 1713
655 #define RPC_S_NO_PROTSEQS_REGISTERED 1714
656 #define RPC_S_NOT_LISTENING 1715
657 #define RPC_S_UNKNOWN_MGR_TYPE 1716
658 #define RPC_S_UNKNOWN_IF 1717
659 #define RPC_S_NO_BINDINGS 1718
660 #define RPC_S_NO_PROTSEQS 1719
661 #define RPC_S_CANT_CREATE_ENDPOINT 1720
662 #define RPC_S_OUT_OF_RESOURCES 1721
663 #define RPC_S_SERVER_UNAVAILABLE 1722
664 #define RPC_S_SERVER_TOO_BUSY 1723
```

```
665 #define RPC_S_INVALID_NETWORK_OPTIONS 1724
666 #define RPC_S_NO_CALL_ACTIVE 1725
667 #define RPC_S_CALL_FAILED 1726
668 #define RPC_S_CALL_FAILED_DNE 1727
669 #define RPC_S_PROTOCOL_ERROR 1728
670 #define RPC_S_UNSUPPORTED_TRANS_SYN 1730
671 #define RPC_S_UNSUPPORTED_TYPE 1732
672 #define RPC_S_INVALID_TAG 1733
673 #define RPC_S_INVALID_BOUND 1734
674 #define RPC_S_NO_ENTRY_NAME 1735
675 #define RPC_S_INVALID_NAME_SYNTAX 1736
676 #define RPC_S_UNSUPPORTED_NAME_SYNTAX 1737
677 #define RPC_S_UUID_NO_ADDRESS 1739
678 #define RPC_S_DUPLICATE_ENDPOINT 1740
679 #define RPC_S_UNKNOWN_AUTHN_TYPE 1741
680 #define RPC_S_MAX_CALLS_TOO_SMALL 1742
681 #define RPC_S_STRING_TOO_LONG 1743
682 #define RPC_S_PROTSEQ_NOT_FOUND 1744
683 #define RPC_S_PROCNUM_OUT_OF_RANGE 1745
684 #define RPC_S_BINDING_HAS_NO_AUTH 1746
685 #define RPC_S_UNKNOWN_AUTHN_SERVICE 1747
686 #define RPC_S_UNKNOWN_AUTHN_LEVEL 1748
687 #define RPC_S_INVALID_AUTH_IDENTITY 1749
688 #define RPC_S_UNKNOWN_AUTHZ_SERVICE 1750
689 #define EPT_S_INVALID_ENTRY 1751
690 #define EPT_S_CANT_PERFORM_OP 1752
691 #define EPT_S_NOT_REGISTERED 1753
692 #define RPC_S_NOTHING_TO_EXPORT 1754
693 #define RPC_S_INCOMPLETE_NAME 1755
694 #define RPC_S_INVALID_VERS_OPTION 1756
695 #define RPC_S_NO_MORE_MEMBERS 1757
696 #define RPC_S_NOT_ALL_OBJS_UNEXPORTED 1758
697 #define RPC_S_INTERFACE_NOT_FOUND 1759
698 #define RPC_S_ENTRY_ALREADY_EXISTS 1760
699 #define RPC_S_ENTRY_NOT_FOUND 1761
700 #define RPC_S_NAME_SERVICE_UNAVAILABLE 1762
701 #define RPC_S_INVALID_NAF_ID 1763
702 #define RPC_S_CANNOT_SUPPORT 1764
703 #define RPC_S_NO_CONTEXT_AVAILABLE 1765
704 #define RPC_S_INTERNAL_ERROR 1766
705 #define RPC_S_ZERO_DIVIDE 1767
706 #define RPC_S_ADDRESS_ERROR 1768
707 #define RPC_S_FP_DIV_ZERO 1769
708 #define RPC_S_FP_UNDERFLOW 1770
709 #define RPC_S_FP_OVERFLOW 1771
710 #define RPC_X_NO_MORE_ENTRIES 1772
711 #define RPC_X_SS_CHAR_TRANS_OPEN_FAIL 1773
712 #define RPC_X_SS_CHAR_TRANS_SHORT_FILE 1774
713 #define RPC_X_SS_IN_NULL_CONTEXT 1775
714 #define RPC_X_SS_CONTEXT_DAMAGED 1777
715 #define RPC_X_SS_HANDLES_MISMATCH 1778
716 #define RPC_X_SS_CANNOT_GET_CALL_HANDLE 1779
717 #define RPC_X_NULL_REF_POINTER 1780
718 #define RPC_X_ENUM_VALUE_OUT_OF_RANGE 1781
719 #define RPC_X_BYTE_COUNT_TOO_SMALL 1782
720 #define RPC_X_BAD_STUB_DATA 1783
721 #define ERROR_INVALID_USER_BUFFER 1784
722 #define ERROR_UNRECOGNIZED_MEDIA 1785
723 #define ERROR_NO_TRUST_LSA_SECRET 1786
724 #define ERROR_NO_TRUST_SAM_ACCOUNT 1787
725 #define ERROR_TRUSTED_DOMAIN_FAILURE 1788
726 #define ERROR_TRUSTED_RELATIONSHIP_FAILURE 1789
727 #define ERROR_TRUST_FAILURE 1790
728 #define RPC_S_CALL_IN_PROGRESS 1791
729 #define ERROR_NETLOGON_NOT_STARTED 1792
730 #define ERROR_ACCOUNT_EXPIRED 1793
731 #define ERROR_REDIRECTOR_HAS_OPEN_HANDLES 1794
732 #define ERROR_PRINTER_DRIVER_ALREADY_INSTALLED 1795
733 #define ERROR_UNKNOWN_PORT 1796
734 #define ERROR_UNKNOWN_PRINTER_DRIVER 1797
735 #define ERROR_UNKNOWN_PRINTPROCESSOR 1798
736 #define ERROR_INVALID_SEPARATOR_FILE 1799
737 #define ERROR_INVALID_PRIORITY 1800
738 #define ERROR_INVALID_PRINTER_NAME 1801
739 #define ERROR_PRINTER_ALREADY_EXISTS 1802
740 #define ERROR_INVALID_PRINTER_COMMAND 1803
741 #define ERROR_INVALID_DATATYPE 1804
742 #define ERROR_INVALID_ENVIRONMENT 1805
743 #define RPC_S_NO_MORE_BINDINGS 1806
744 #define ERROR_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT 1807
745 #define ERROR_NOLOGON_WORKSTATION_TRUST_ACCOUNT 1808
746 #define ERROR_NOLOGON_SERVER_TRUST_ACCOUNT 1809
747 #define ERROR_DOMAIN_TRUST_INCONSISTENT 1810
748 #define ERROR_SERVER_HAS_OPEN_HANDLES 1811
749 #define ERROR_RESOURCE_DATA_NOT_FOUND 1812
750 #define ERROR_RESOURCE_TYPE_NOT_FOUND 1813
751 #define ERROR_RESOURCE_NAME_NOT_FOUND 1814
```

```
752 #define ERROR_RESOURCE_LANG_NOT_FOUND 1815
753 #define ERROR_NOT_ENOUGH_QUOTA 1816
754 #define RPC_S_NO_INTERFACES 1817
755 #define RPC_S_CALL_CANCELLED 1818
756 #define RPC_S_BINDING_INCOMPLETE 1819
757 #define RPC_S_COMM_FAILURE 1820
758 #define RPC_S_UNSUPPORTED_AUTHN_LEVEL 1821
759 #define RPC_S_NO_PRINC_NAME 1822
760 #define RPC_S_NOT_RPC_ERROR 1823
761 #define RPC_S_UUID_LOCAL_ONLY 1824
762 #define RPC_S_SEC_PKG_ERROR 1825
763 #define RPC_S_NOT_CANCELLED 1826
764 #define RPC_X_INVALID_ES_ACTION 1827
765 #define RPC_X_WRONG_ES_VERSION 1828
766 #define RPC_X_WRONG_STUB_VERSION 1829
767 #define RPC_X_INVALID_PIPE_OBJECT 1830
768 #define RPC_X_WRONG_PIPE_ORDER 1831
769 #define RPC_X_WRONG_PIPE_VERSION 1832
770 #define RPC_S_GROUP_MEMBER_NOT_FOUND 1898
771 #define EPT_S_CANT_CREATE 1899
772 #define RPC_S_INVALID_OBJECT 1900
773 #define ERROR_INVALID_TIME 1901
774 #define ERROR_INVALID_FORM_NAME 1902
775 #define ERROR_INVALID_FORM_SIZE 1903
776 #define ERROR_ALREADY_WAITING 1904
777 #define ERROR_PRINTER_DELETED 1905
778 #define ERROR_INVALID_PRINTER_STATE 1906
779 #define ERROR_PASSWORD_MUST_CHANGE 1907
780 #define ERROR_DOMAIN_CONTROLLER_NOT_FOUND 1908
781 #define ERROR_ACCOUNT_LOCKED_OUT 1909
782 #define OR_INVALID_OXID 1910
783 #define OR_INVALID_OID 1911
784 #define OR_INVALID_SET 1912
785 #define RPC_S_SEND_INCOMPLETE 1913
786 #define RPC_S_INVALID_ASYNC_HANDLE 1914
787 #define RPC_S_INVALID_ASYNC_CALL 1915
788 #define RPC_X_PIPE_CLOSED 1916
789 #define RPC_X_PIPE_DISCIPLINE_ERROR 1917
790 #define RPC_X_PIPE_EMPTY 1918
791 #define ERROR_NO_SITENAME 1919
792 #define ERROR_CANT_ACCESS_FILE 1920
793 #define ERROR_CANT_RESOLVE_FILENAME 1921
794 #define RPC_S_ENTRY_TYPE_MISMATCH 1922
795 #define RPC_S_NOT_ALL_OBJS_EXPORTED 1923
796 #define RPC_S_INTERFACE_NOT_EXPORTED 1924
797 #define RPC_S_PROFILE_NOT_ADDED 1925
798 #define RPC_S_PRF_ELT_NOT_ADDED 1926
799 #define RPC_S_PRF_ELT_NOT_REMOVED 1927
800 #define RPC_S_GRP_ELT_NOT_ADDED 1928
801 #define RPC_S_GRP_ELT_NOT_REMOVED 1929
802 #define ERROR_INVALID_PIXEL_FORMAT 2000
803 #define ERROR_BAD_DRIVER 2001
804 #define ERROR_INVALID_WINDOW_STYLE 2002
805 #define ERROR_METAFILE_NOT_SUPPORTED 2003
806 #define ERROR_TRANSFORM_NOT_SUPPORTED 2004
807 #define ERROR_CLIPPING_NOT_SUPPORTED 2005
808 #define ERROR_INVALID_CMM 2010
809 #define ERROR_INVALID_PROFILE 2011
810 #define ERROR_TAG_NOT_FOUND 2012
811 #define ERROR_TAG_NOT_PRESENT 2013
812 #define ERROR_DUPLICATE_TAG 2014
813 #define ERROR_PROFILE_NOT_ASSOCIATED_WITH_DEVICE 2015
814 #define ERROR_PROFILE_NOT_FOUND 2016
815 #define ERROR_INVALID_COLORSPACE 2017
816 #define ERROR_ICM_NOT_ENABLED 2018
817 #define ERROR_DELETING_ICM_XFORM 2019
818 #define ERROR_INVALID_TRANSFORM 2020
819 #define ERROR_COLORSPACE_MISMATCH 2021
820 #define ERROR_INVALID_COLORINDEX 2022
821 #define ERROR_CONNECTED_OTHER_PASSWORD 2108
822 #define ERROR_BAD_USERNAME 2202
823 #define ERROR_NOT_CONNECTED 2250
824 #define ERROR_OPEN_FILES 2401
825 #define ERROR_ACTIVE_CONNECTIONS 2402
826 #define ERROR_DEVICE_IN_USE 2404
827 #define ERROR_UNKNOWN_PRINT_MONITOR 3000
828 #define ERROR_PRINTER_DRIVER_IN_USE 3001
829 #define ERROR_SPOOL_FILE_NOT_FOUND 3002
830 #define ERROR_SPL_NO_STARTDOC 3003
831 #define ERROR_SPL_NO_ADDJOB 3004
832 #define ERROR_PRINT_PROCESSOR_ALREADY_INSTALLED 3005
833 #define ERROR_PRINT_MONITOR_ALREADY_INSTALLED 3006
834 #define ERROR_INVALID_PRINT_MONITOR 3007
835 #define ERROR_PRINT_MONITOR_IN_USE 3008
836 #define ERROR_PRINTER_HAS_JOBS_QUEUED 3009
837 #define ERROR_SUCCESS_REBOOT_REQUIRED 3010
838 #define ERROR_SUCCESS_RESTART_REQUIRED 3011
```

```
839 #define ERROR_PRINTER_NOT_FOUND 3012
840 #define ERROR_WINS_INTERNAL 4000
841 #define ERROR_CAN_NOT_DEL_LOCAL_WINS 4001
842 #define ERROR_STATIC_INIT 4002
843 #define ERROR_INC_BACKUP 4003
844 #define ERROR_FULL_BACKUP 4004
845 #define ERROR_REC_NON_EXISTENT 4005
846 #define ERROR_RPL_NOT_ALLOWED 4006
847 #define ERROR_DHCP_ADDRESS_CONFLICT 4100
848 #define ERROR_WMI_GUID_NOT_FOUND 4200
849 #define ERROR_WMI_INSTANCE_NOT_FOUND 4201
850 #define ERROR_WMI_ITEMID_NOT_FOUND 4202
851 #define ERROR_WMI_TRY_AGAIN 4203
852 #define ERROR_WMI_DP_NOT_FOUND 4204
853 #define ERROR_WMI_UNRESOLVED_INSTANCE_REF 4205
854 #define ERROR_WMI_ALREADY_ENABLED 4206
855 #define ERROR_WMI_GUID_DISCONNECTED 4207
856 #define ERROR_WMI_SERVER_UNAVAILABLE 4208
857 #define ERROR_WMI_DP_FAILED 4209
858 #define ERROR_WMI_INVALID_MOF 4210
859 #define ERROR_WMI_INVALID_REGINFO 4211
860 #define ERROR_WMI_ALREADY_DISABLED 4212
861 #define ERROR_WMI_READ_ONLY 4213
862 #define ERROR_WMI_SET_FAILURE 4214
863 #define ERROR_INVALID_MEDIA 4300
864 #define ERROR_INVALID_LIBRARY 4301
865 #define ERROR_INVALID_MEDIA_POOL 4302
866 #define ERROR_DRIVE_MEDIA_MISMATCH 4303
867 #define ERROR_MEDIA_OFFLINE 4304
868 #define ERROR_LIBRARY_OFFLINE 4305
869 #define ERROR_EMPTY 4306
870 #define ERROR_NOT_EMPTY 4307
871 #define ERROR_MEDIA_UNAVAILABLE 4308
872 #define ERROR_RESOURCE_DISABLED 4309
873 #define ERROR_INVALID_CLEANER 4310
874 #define ERROR_UNABLE_TO_CLEAN 4311
875 #define ERROR_OBJECT_NOT_FOUND 4312
876 #define ERROR_DATABASE_FAILURE 4313
877 #define ERROR_DATABASE_FULL 4314
878 #define ERROR_MEDIA_INCOMPATIBLE 4315
879 #define ERROR_RESOURCE_NOT_PRESENT 4316
880 #define ERROR_INVALID_OPERATION 4317
881 #define ERROR_MEDIA_NOT_AVAILABLE 4318
882 #define ERROR_DEVICE_NOT_AVAILABLE 4319
883 #define ERROR_REQUEST_REFUSED 4320
884 #define ERROR_INVALID_DRIVE_OBJECT 4321
885 #define ERROR_LIBRARY_FULL 4322
886 #define ERROR_MEDIUM_NOT_ACCESSIBLE 4323
887 #define ERROR_UNABLE_TO_LOAD_MEDIUM 4324
888 #define ERROR_UNABLE_TO_INVENTORY_DRIVE 4325
889 #define ERROR_UNABLE_TO_INVENTORY_SLOT 4326
890 #define ERROR_UNABLE_TO_INVENTORY_TRANSPORT 4327
891 #define ERROR_TRANSPORT_FULL 4328
892 #define ERROR_CONTROLLING_IEPORT 4329
893 #define ERROR_UNABLE_TO_EJECT_MOUNTED_MEDIA 4330
894 #define ERROR_CLEANER_SLOT_SET 4331
895 #define ERROR_CLEANER_SLOT_NOT_SET 4332
896 #define ERROR_CLEANER_CARTRIDGE_SPENT 4333
897 #define ERROR_UNEXPECTED_OMID 4334
898 #define ERROR_CANT_DELETE_LAST_ITEM 4335
899 #define ERROR_MESSAGE_EXCEEDS_MAX_SIZE 4336
900 #define ERROR_VOLUME_CONTAINS_SYS_FILES 4337
901 #define ERROR_INDIGENOUS_TYPE 4338
902 #define ERROR_NO_SUPPORTING_DRIVES 4339
903 #define ERROR_FILE_OFFLINE 4350
904 #define ERROR_REMOTE_STORAGE_NOT_ACTIVE 4351
905 #define ERROR_REMOTE_STORAGE_MEDIA_ERROR 4352
906 #define ERROR_NOT_A_REPARSE_POINT 4390
907 #define ERROR_REPARSE_ATTRIBUTE_CONFLICT 4391
908 #define ERROR_INVALID_REPARSE_DATA 4392
909 #define ERROR_REPARSE_TAG_INVALID 4393
910 #define ERROR_REPARSE_TAG_MISMATCH 4394
911 #define ERROR_VOLUME_NOT_SIS_ENABLED 4500
912 #define ERROR_DEPENDENT_RESOURCE_EXISTS 5001
913 #define ERROR_DEPENDENCY_NOT_FOUND 5002
914 #define ERROR_DEPENDENCY_ALREADY_EXISTS 5003
915 #define ERROR_RESOURCE_NOT_ONLINE 5004
916 #define ERROR_HOST_NODE_NOT_AVAILABLE 5005
917 #define ERROR_RESOURCE_NOT_AVAILABLE 5006
918 #define ERROR_RESOURCE_NOT_FOUND 5007
919 #define ERROR_SHUTDOWN_CLUSTER 5008
920 #define ERROR_CANT_EVICT_ACTIVE_NODE 5009
921 #define ERROR_OBJECT_ALREADY_EXISTS 5010
922 #define ERROR_OBJECT_IN_LIST 5011
923 #define ERROR_GROUP_NOT_AVAILABLE 5012
924 #define ERROR_GROUP_NOT_FOUND 5013
925 #define ERROR_GROUP_NOT_ONLINE 5014
```

```
926 #define ERROR_HOST_NODE_NOT_RESOURCE_OWNER 5015
927 #define ERROR_HOST_NODE_NOT_GROUP_OWNER 5016
928 #define ERROR_RESMON_CREATE_FAILED 5017
929 #define ERROR_RESMON_ONLINE_FAILED 5018
930 #define ERROR_RESOURCE_ONLINE 5019
931 #define ERROR_QUORUM_RESOURCE 5020
932 #define ERROR_NOT_QUORUM_CAPABLE 5021
933 #define ERROR_CLUSTER_SHUTTING_DOWN 5022
934 #define ERROR_INVALID_STATE 5023
935 #define ERROR_RESOURCE_PROPERTIES_STORED 5024
936 #define ERROR_NOT_QUORUM_CLASS 5025
937 #define ERROR_CORE_RESOURCE 5026
938 #define ERROR_QUORUM_RESOURCE_ONLINE_FAILED 5027
939 #define ERROR_QUORUMLOG_OPEN_FAILED 5028
940 #define ERROR_CLUSTERLOG_CORRUPT 5029
941 #define ERROR_CLUSTERLOG_RECORD_EXCEEDS_MAXSIZE 5030
942 #define ERROR_CLUSTERLOG_EXCEEDS_MAXSIZE 5031
943 #define ERROR_CLUSTERLOG_CHKPOINT_NOT_FOUND 5032
944 #define ERROR_CLUSTERLOG_NOT_ENOUGH_SPACE 5033
945 #define ERROR_QUORUM_OWNER_ALIVE 5034
946 #define ERROR_NETWORK_NOT_AVAILABLE 5035
947 #define ERROR_NODE_NOT_AVAILABLE 5036
948 #define ERROR_ALL_NODES_NOT_AVAILABLE 5037
949 #define ERROR_RESOURCE_FAILED 5038
950 #define ERROR_CLUSTER_INVALID_NODE 5039
951 #define ERROR_CLUSTER_NODE_EXISTS 5040
952 #define ERROR_CLUSTER_JOIN_IN_PROGRESS 5041
953 #define ERROR_CLUSTER_NODE_NOT_FOUND 5042
954 #define ERROR_CLUSTER_LOCAL_NODE_NOT_FOUND 5043
955 #define ERROR_CLUSTER_NETWORK_EXISTS 5044
956 #define ERROR_CLUSTER_NETWORK_NOT_FOUND 5045
957 #define ERROR_CLUSTER_NETINTERFACE_EXISTS 5046
958 #define ERROR_CLUSTER_NETINTERFACE_NOT_FOUND 5047
959 #define ERROR_CLUSTER_INVALID_REQUEST 5048
960 #define ERROR_CLUSTER_INVALID_NETWORK_PROVIDER 5049
961 #define ERROR_CLUSTER_NODE_DOWN 5050
962 #define ERROR_CLUSTER_NODE_UNREACHABLE 5051
963 #define ERROR_CLUSTER_NODE_NOT_MEMBER 5052
964 #define ERROR_CLUSTER_JOIN_NOT_IN_PROGRESS 5053
965 #define ERROR_CLUSTER_INVALID_NETWORK 5054
966 #define ERROR_CLUSTER_NODE_UP 5056
967 #define ERROR_CLUSTER_IPADDR_IN_USE 5057
968 #define ERROR_CLUSTER_NODE_NOT_PAUSED 5058
969 #define ERROR_CLUSTER_NO_SECURITY_CONTEXT 5059
970 #define ERROR_CLUSTER_NETWORK_NOT_INTERNAL 5060
971 #define ERROR_CLUSTER_NODE_ALREADY_UP 5061
972 #define ERROR_CLUSTER_NODE_ALREADY_DOWN 5062
973 #define ERROR_CLUSTER_NETWORK_ALREADY_ONLINE 5063
974 #define ERROR_CLUSTER_NETWORK_ALREADY_OFFLINE 5064
975 #define ERROR_CLUSTER_NODE_ALREADY_MEMBER 5065
976 #define ERROR_CLUSTER_LAST_INTERNAL_NETWORK 5066
977 #define ERROR_CLUSTER_NETWORK_HAS_DEPENDENTS 5067
978 #define ERROR_INVALID_OPERATION_ON_QUORUM 5068
979 #define ERROR_DEPENDENCY_NOT_ALLOWED 5069
980 #define ERROR_CLUSTER_NODE_PAUSED 5070
981 #define ERROR_NODE_CANT_HOST_RESOURCE 5071
982 #define ERROR_CLUSTER_NODE_NOT_READY 5072
983 #define ERROR_CLUSTER_NODE_SHUTTING_DOWN 5073
984 #define ERROR_CLUSTER_JOIN_ABORTED 5074
985 #define ERROR_CLUSTER_INCOMPATIBLE_VERSIONS 5075
986 #define ERROR_CLUSTER_MAXNUM_OF_RESOURCES_EXCEEDED 5076
987 #define ERROR_CLUSTER_SYSTEM_CONFIG_CHANGED 5077
988 #define ERROR_CLUSTER_RESOURCE_TYPE_NOT_FOUND 5078
989 #define ERROR_CLUSTER_RESTYPE_NOT_SUPPORTED 5079
990 #define ERROR_CLUSTER_RESNAME_NOT_FOUND 5080
991 #define ERROR_CLUSTER_NO_RPC_PACKAGES_REGISTERED 5081
992 #define ERROR_CLUSTER_OWNER_NOT_IN_PREFLIST 5082
993 #define ERROR_CLUSTER_DATABASE_SEQMISMATCH 5083
994 #define ERROR_RESMON_INVALID_STATE 5084
995 #define ERROR_CLUSTER_GUM_NOT_LOCKER 5085
996 #define ERROR_QUORUM_DISK_NOT_FOUND 5086
997 #define ERROR_DATABASE_BACKUP_CORRUPT 5087
998 #define ERROR_CLUSTER_NODE_ALREADY_HAS_DFS_ROOT 5088
999 #define ERROR_RESOURCE_PROPERTY_UNCHANGEABLE 5089
1000 #define ERROR_ENCRYPTION_FAILED 6000
1001 #define ERROR_DECRYPTION_FAILED 6001
1002 #define ERROR_FILE_ENCRYPTED 6002
1003 #define ERROR_NO_RECOVERY_POLICY 6003
1004 #define ERROR_NO_EFS 6004
1005 #define ERROR_WRONG_EFS 6005
1006 #define ERROR_NO_USER_KEYS 6006
1007 #define ERROR_FILE_NOT_ENCRYPTED 6007
1008 #define ERROR_NOT_EXPORT_FORMAT 6008
1009 #define ERROR_FILE_READ_ONLY 6009
1010 #define ERROR_DIR_EFS_DISALLOWED 6010
1011 #define ERROR_EFS_SERVER_NOT_TRUSTED 6011
1012 #define ERROR_NO_BROWSER_SERVERS_FOUND 6118
```

```
1013 #define SCHED_E_SERVICE_NOT_LOCALSYSTEM 6200
1014 #define ERROR_CTX_WINSTATION_NAME_INVALID 7001
1015 #define ERROR_CTX_INVALID_PD 7002
1016 #define ERROR_CTX_PD_NOT_FOUND 7003
1017 #define ERROR_CTX_WD_NOT_FOUND 7004
1018 #define ERROR_CTX_CANNOT_MAKE_EVENTLOG_ENTRY 7005
1019 #define ERROR_CTX_SERVICE_NAME_COLLISION 7006
1020 #define ERROR_CTX_CLOSE_PENDING 7007
1021 #define ERROR_CTX_NO_OUTBUF 7008
1022 #define ERROR_CTX_MODEM_INF_NOT_FOUND 7009
1023 #define ERROR_CTX_INVALID_MODEMNAME 7010
1024 #define ERROR_CTX_MODEM_RESPONSE_ERROR 7011
1025 #define ERROR_CTX_MODEM_RESPONSE_TIMEOUT 7012
1026 #define ERROR_CTX_MODEM_RESPONSE_NO_CARRIER 7013
1027 #define ERROR_CTX_MODEM_RESPONSE_NO_DIALTONE 7014
1028 #define ERROR_CTX_MODEM_RESPONSE_BUSY 7015
1029 #define ERROR_CTX_MODEM_RESPONSE_VOICE 7016
1030 #define ERROR_CTX_TD_ERROR 7017
1031 #define ERROR_CTX_WINSTATION_NOT_FOUND 7022
1032 #define ERROR_CTX_WINSTATION_ALREADY_EXISTS 7023
1033 #define ERROR_CTX_WINSTATION_BUSY 7024
1034 #define ERROR_CTX_BAD_VIDEO_MODE 7025
1035 #define ERROR_CTX_GRAPHICS_INVALID 7035
1036 #define ERROR_CTX_LOGON_DISABLED 7037
1037 #define ERROR_CTX_NOT_CONSOLE 7038
1038 #define ERROR_CTX_CLIENT_QUERY_TIMEOUT 7040
1039 #define ERROR_CTX_CONSOLE_DISCONNECT 7041
1040 #define ERROR_CTX_CONSOLE_CONNECT 7042
1041 #define ERROR_CTX_SHADOW_DENIED 7044
1042 #define ERROR_CTX_WINSTATION_ACCESS_DENIED 7045
1043 #define ERROR_CTX_INVALID_WD 7049
1044 #define ERROR_CTX_SHADOW_INVALID 7050
1045 #define ERROR_CTX_SHADOW_DISABLED 7051
1046 #define ERROR_CTX_CLIENT_LICENSE_IN_USE 7052
1047 #define ERROR_CTX_CLIENT_LICENSE_NOT_SET 7053
1048 #define ERROR_CTX_LICENSE_NOT_AVAILABLE 7054
1049 #define ERROR_CTX_LICENSE_CLIENT_INVALID 7055
1050 #define ERROR_CTX_LICENSE_EXPIRED 7056
1051 #define FRS_ERR_INVALID_API_SEQUENCE 8001
1052 #define FRS_ERR_STARTING_SERVICE 8002
1053 #define FRS_ERR_STOPPING_SERVICE 8003
1054 #define FRS_ERR_INTERNAL_API 8004
1055 #define FRS_ERR_INTERNAL 8005
1056 #define FRS_ERR_SERVICE_COMM 8006
1057 #define FRS_ERR_INSUFFICIENT_PRIV 8007
1058 #define FRS_ERR_AUTHENTICATION 8008
1059 #define FRS_ERR_PARENT_INSUFFICIENT_PRIV 8009
1060 #define FRS_ERR_PARENT_AUTHENTICATION 8010
1061 #define FRS_ERR_CHILD_TO_PARENT_COMM 8011
1062 #define FRS_ERR_PARENT_TO_CHILD_COMM 8012
1063 #define FRS_ERR_SYSVOL_POPULATE 8013
1064 #define FRS_ERR_SYSVOL_POPULATE_TIMEOUT 8014
1065 #define FRS_ERR_SYSVOL_IS_BUSY 8015
1066 #define FRS_ERR_SYSVOL_DEMOTE 8016
1067 #define FRS_ERR_INVALID_SERVICE_PARAMETER 8017
1068 #define ERROR_DS_NOT_INSTALLED 8200
1069 #define ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY 8201
1070 #define ERROR_DS_NO_ATTRIBUTE_OR_VALUE 8202
1071 #define ERROR_DS_INVALID_ATTRIBUTE_SYNTAX 8203
1072 #define ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED 8204
1073 #define ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS 8205
1074 #define ERROR_DS_BUSY 8206
1075 #define ERROR_DS_UNAVAILABLE 8207
1076 #define ERROR_DS_NO_RIDS_ALLOCATED 8208
1077 #define ERROR_DS_NO_MORE_RIDS 8209
1078 #define ERROR_DS_INCORRECT_ROLE_OWNER 8210
1079 #define ERROR_DS_RIDMGR_INIT_ERROR 8211
1080 #define ERROR_DS_OBJ_CLASS_VIOLATION 8212
1081 #define ERROR_DS_CANT_ON_NON_LEAF 8213
1082 #define ERROR_DS_CANT_ON_RDN 8214
1083 #define ERROR_DS_CANT_MOD_OBJ_CLASS 8215
1084 #define ERROR_DS_CROSS_DOM_MOVE_ERROR 8216
1085 #define ERROR_DS_GC_NOT_AVAILABLE 8217
1086 #define ERROR_SHARED_POLICY 8218
1087 #define ERROR_POLICY_OBJECT_NOT_FOUND 8219
1088 #define ERROR_POLICY_ONLY_IN_DS 8220
1089 #define ERROR_PROMOTION_ACTIVE 8221
1090 #define ERROR_NO_PROMOTION_ACTIVE 8222
1091 #define ERROR_DS_OPERATIONS_ERROR 8224
1092 #define ERROR_DS_PROTOCOL_ERROR 8225
1093 #define ERROR_DS_TIMELIMIT_EXCEEDED 8226
1094 #define ERROR_DS_SIZELIMIT_EXCEEDED 8227
1095 #define ERROR_DS_ADMIN_LIMIT_EXCEEDED 8228
1096 #define ERROR_DS_COMPARE_FALSE 8229
1097 #define ERROR_DS_COMPARE_TRUE 8230
1098 #define ERROR_DS_AUTH_METHOD_NOT_SUPPORTED 8231
1099 #define ERROR_DS_STRONG_AUTH_REQUIRED 8232
```


1100	#define	ERROR_DS_INAPPROPRIATE_AUTH	8233
1101	#define	ERROR_DS_AUTH_UNKNOWN	8234
1102	#define	ERROR_DS_REFERRAL	8235
1103	#define	ERROR_DS_UNAVAILABLE_CRIT_EXTENSION	8236
1104	#define	ERROR_DS_CONFIDENTIALITY_REQUIRED	8237
1105	#define	ERROR_DS_INAPPROPRIATE_MATCHING	8238
1106	#define	ERROR_DS_CONSTRAINT_VIOLATION	8239
1107	#define	ERROR_DS_NO_SUCH_OBJECT	8240
1108	#define	ERROR_DS_ALIAS_PROBLEM	8241
1109	#define	ERROR_DS_INVALID_DN_SYNTAX	8242
1110	#define	ERROR_DS_IS_LEAF	8243
1111	#define	ERROR_DS_ALIAS_DEREF_PROBLEM	8244
1112	#define	ERROR_DS_UNWILLING_TO_PERFORM	8245
1113	#define	ERROR_DS_LOOP_DETECT	8246
1114	#define	ERROR_DS_NAMING_VIOLATION	8247
1115	#define	ERROR_DS_OBJECT_RESULTS_TOO_LARGE	8248
1116	#define	ERROR_DS_AFFECTS_MULTIPLE_DSAS	8249
1117	#define	ERROR_DS_SERVER_DOWN	8250
1118	#define	ERROR_DS_LOCAL_ERROR	8251
1119	#define	ERROR_DS_ENCODING_ERROR	8252
1120	#define	ERROR_DS_DECODING_ERROR	8253
1121	#define	ERROR_DS_FILTER_UNKNOWN	8254
1122	#define	ERROR_DS_PARAM_ERROR	8255
1123	#define	ERROR_DS_NOT_SUPPORTED	8256
1124	#define	ERROR_DS_NO_RESULTS_RETURNED	8257
1125	#define	ERROR_DS_CONTROL_NOT_FOUND	8258
1126	#define	ERROR_DS_CLIENT_LOOP	8259
1127	#define	ERROR_DS_REFERRAL_LIMIT_EXCEEDED	8260
1128	#define	ERROR_DS_ROOT_MUST_BE_NC	8301
1129	#define	ERROR_DS_ADD_REPLICA_INHIBITED	8302
1130	#define	ERROR_DS_ATT_NOT_DEF_IN_SCHEMA	8303
1131	#define	ERROR_DS_MAX_OBJ_SIZE_EXCEEDED	8304
1132	#define	ERROR_DS_OBJ_STRING_NAME_EXISTS	8305
1133	#define	ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA	8306
1134	#define	ERROR_DS_RDN_DOESNT_MATCH_SCHEMA	8307
1135	#define	ERROR_DS_NO_REQUESTED_ATTS_FOUND	8308
1136	#define	ERROR_DS_USER_BUFFER_TO_SMALL	8309
1137	#define	ERROR_DS_ATT_IS_NOT_ON_OBJ	8310
1138	#define	ERROR_DS_ILLEGAL_MOD_OPERATION	8311
1139	#define	ERROR_DS_OBJ_TOO_LARGE	8312
1140	#define	ERROR_DS_BAD_INSTANCE_TYPE	8313
1141	#define	ERROR_DS_MASTERDSA_REQUIRED	8314
1142	#define	ERROR_DS_OBJECT_CLASS_REQUIRED	8315
1143	#define	ERROR_DS_MISSING_REQUIRED_ATT	8316
1144	#define	ERROR_DS_ATT_NOT_DEF_FOR_CLASS	8317
1145	#define	ERROR_DS_ATT_ALREADY_EXISTS	8318
1146	#define	ERROR_DS_CANT_ADD_ATT_VALUES	8320
1147	#define	ERROR_DS_SINGLE_VALUE_CONSTRAINT	8321
1148	#define	ERROR_DS_RANGE_CONSTRAINT	8322
1149	#define	ERROR_DS_ATT_VAL_ALREADY_EXISTS	8323
1150	#define	ERROR_DS_CANT_REM_MISSING_ATT	8324
1151	#define	ERROR_DS_CANT_REM_MISSING_ATT_VAL	8325
1152	#define	ERROR_DS_ROOT_CANT_BE_SUBREF	8326
1153	#define	ERROR_DS_NO_CHAINING	8327
1154	#define	ERROR_DS_NO_CHAINED_EVAL	8328
1155	#define	ERROR_DS_NO_PARENT_OBJECT	8329
1156	#define	ERROR_DS_PARENT_IS_AN_ALIAS	8330
1157	#define	ERROR_DS_CANT_MIX_MASTER_AND_REPS	8331
1158	#define	ERROR_DS_CHILDREN_EXIST	8332
1159	#define	ERROR_DS_OBJ_NOT_FOUND	8333
1160	#define	ERROR_DS_ALIASSED_OBJ_MISSING	8334
1161	#define	ERROR_DS_BAD_NAME_SYNTAX	8335
1162	#define	ERROR_DS_ALIAS_POINTS_TO_ALIAS	8336
1163	#define	ERROR_DS_CANT_DEREF_ALIAS	8337
1164	#define	ERROR_DS_OUT_OF_SCOPE	8338
1165	#define	ERROR_DS_CANT_DELETE_DSA_OBJ	8340
1166	#define	ERROR_DS_GENERIC_ERROR	8341
1167	#define	ERROR_DS_DSA_MUST_BE_INT_MASTER	8342
1168	#define	ERROR_DS_CLASS_NOT_DSA	8343
1169	#define	ERROR_DS_INSUFF_ACCESS_RIGHTS	8344
1170	#define	ERROR_DS_ILLEGAL_SUPERIOR	8345
1171	#define	ERROR_DS_ATTRIBUTE_OWNED_BY_SAM	8346
1172	#define	ERROR_DS_NAME_TOO_MANY_PARTS	8347
1173	#define	ERROR_DS_NAME_TOO_LONG	8348
1174	#define	ERROR_DS_NAME_VALUE_TOO_LONG	8349
1175	#define	ERROR_DS_NAME_UNPARSEABLE	8350
1176	#define	ERROR_DS_NAME_TYPE_UNKNOWN	8351
1177	#define	ERROR_DS_NOT_AN_OBJECT	8352
1178	#define	ERROR_DS_SEC_DESC_TOO_SHORT	8353
1179	#define	ERROR_DS_SEC_DESC_INVALID	8354
1180	#define	ERROR_DS_NO_DELETED_NAME	8355
1181	#define	ERROR_DS_SUBREF_MUST_HAVE_PARENT	8356
1182	#define	ERROR_DS_NCNAME_MUST_BE_NC	8357
1183	#define	ERROR_DS_CANT_ADD_SYSTEM_ONLY	8358
1184	#define	ERROR_DS_CLASS_MUST_BE_CONCRETE	8359
1185	#define	ERROR_DS_INVALID_DMD	8360
1186	#define	ERROR_DS_OBJ_GUID_EXISTS	8361

```
1187 #define ERROR_DS_NOT_ON_BACKLINK 8362
1188 #define ERROR_DS_NO_CROSSREF_FOR_NC 8363
1189 #define ERROR_DS_SHUTTING_DOWN 8364
1190 #define ERROR_DS_UNKNOWN_OPERATION 8365
1191 #define ERROR_DS_INVALID_ROLE_OWNER 8366
1192 #define ERROR_DS_COULDNT_CONTACT_FSMO 8367
1193 #define ERROR_DS_CROSS_NC_DN_RENAME 8368
1194 #define ERROR_DS_CANT_MOD_SYSTEM_ONLY 8369
1195 #define ERROR_DS_REPLICATOR_ONLY 8370
1196 #define ERROR_DS_OBJ_CLASS_NOT_DEFINED 8371
1197 #define ERROR_DS_OBJ_CLASS_NOT_SUBCLASS 8372
1198 #define ERROR_DS_NAME_REFERENCE_INVALID 8373
1199 #define ERROR_DS_CROSS_REF_EXISTS 8374
1200 #define ERROR_DS_CANT_DEL_MASTER_CROSSREF 8375
1201 #define ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD 8376
1202 #define ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX 8377
1203 #define ERROR_DS_DUP_RDN 8378
1204 #define ERROR_DS_DUP_OID 8379
1205 #define ERROR_DS_DUP_MAPI_ID 8380
1206 #define ERROR_DS_DUP_SCHEMA_ID_GUID 8381
1207 #define ERROR_DS_DUP_LDAP_DISPLAY_NAME 8382
1208 #define ERROR_DS_SEMANTIC_ATT_TEST 8383
1209 #define ERROR_DS_SYNTAX_MISMATCH 8384
1210 #define ERROR_DS_EXISTS_IN_MUST_HAVE 8385
1211 #define ERROR_DS_EXISTS_IN_MAY_HAVE 8386
1212 #define ERROR_DS_NONEXISTENT_MAY_HAVE 8387
1213 #define ERROR_DS_NONEXISTENT_MUST_HAVE 8388
1214 #define ERROR_DS_AUX_CLS_TEST_FAIL 8389
1215 #define ERROR_DS_NONEXISTENT_POSS_SUP 8390
1216 #define ERROR_DS_SUB_CLS_TEST_FAIL 8391
1217 #define ERROR_DS_BAD_RDN_ATT_ID_SYNTAX 8392
1218 #define ERROR_DS_EXISTS_IN_AUX_CLS 8393
1219 #define ERROR_DS_EXISTS_IN_SUB_CLS 8394
1220 #define ERROR_DS_EXISTS_IN_POSS_SUP 8395
1221 #define ERROR_DS_RECALCSHEMA_FAILED 8396
1222 #define ERROR_DS_TREE_DELETE_NOT_FINISHED 8397
1223 #define ERROR_DS_CANT_DELETE 8398
1224 #define ERROR_DS_ATT_SCHEMA_REQ_ID 8399
1225 #define ERROR_DS_BAD_ATT_SCHEMA_SYNTAX 8400
1226 #define ERROR_DS_CANT_CACHE_ATT 8401
1227 #define ERROR_DS_CANT_CACHE_CLASS 8402
1228 #define ERROR_DS_CANT_REMOVE_ATT_CACHE 8403
1229 #define ERROR_DS_CANT_REMOVE_CLASS_CACHE 8404
1230 #define ERROR_DS_CANT_RETRIEVE_DN 8405
1231 #define ERROR_DS_MISSING_SUPREF 8406
1232 #define ERROR_DS_CANT_RETRIEVE_INSTANCE 8407
1233 #define ERROR_DS_CODE_INCONSISTENCY 8408
1234 #define ERROR_DS_DATABASE_ERROR 8409
1235 #define ERROR_DS_GOVERNSID_MISSING 8410
1236 #define ERROR_DS_MISSING_EXPECTED_ATT 8411
1237 #define ERROR_DS_NCNAME_MISSING_CR_REF 8412
1238 #define ERROR_DS_SECURITY_CHECKING_ERROR 8413
1239 #define ERROR_DS_SCHEMA_NOT_LOADED 8414
1240 #define ERROR_DS_SCHEMA_ALLOC_FAILED 8415
1241 #define ERROR_DS_ATT_SCHEMA_REQ_SYNTAX 8416
1242 #define ERROR_DS_GCVERIFY_ERROR 8417
1243 #define ERROR_DS_DRA_SCHEMA_MISMATCH 8418
1244 #define ERROR_DS_CANT_FIND_DSA_OBJ 8419
1245 #define ERROR_DS_CANT_FIND_EXPECTED_NC 8420
1246 #define ERROR_DS_CANT_FIND_NC_IN_CACHE 8421
1247 #define ERROR_DS_CANT_RETRIEVE_CHILD 8422
1248 #define ERROR_DS_SECURITY_ILLEGAL_MODIFY 8423
1249 #define ERROR_DS_CANT_REPLACE_HIDDEN_REC 8424
1250 #define ERROR_DS_BAD_HIERARCHY_FILE 8425
1251 #define ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED 8426
1252 #define ERROR_DS_CONFIG_PARAM_MISSING 8427
1253 #define ERROR_DS_COUNTING_AB_INDICES_FAILED 8428
1254 #define ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED 8429
1255 #define ERROR_DS_INTERNAL_FAILURE 8430
1256 #define ERROR_DS_UNKNOWN_ERROR 8431
1257 #define ERROR_DS_ROOT_REQUIRES_CLASS_TOP 8432
1258 #define ERROR_DS_REFUSING_FSMO_ROLES 8433
1259 #define ERROR_DS_MISSING_FSMO_SETTINGS 8434
1260 #define ERROR_DS_UNABLE_TO_SURRENDER_ROLES 8435
1261 #define ERROR_DS_DRA_GENERIC 8436
1262 #define ERROR_DS_DRA_INVALID_PARAMETER 8437
1263 #define ERROR_DS_DRA_BUSY 8438
1264 #define ERROR_DS_DRA_BAD_DN 8439
1265 #define ERROR_DS_DRA_BAD_NC 8440
1266 #define ERROR_DS_DRA_DN_EXISTS 8441
1267 #define ERROR_DS_DRA_INTERNAL_ERROR 8442
1268 #define ERROR_DS_DRA_INCONSISTENT_DIT 8443
1269 #define ERROR_DS_DRA_CONNECTION_FAILED 8444
1270 #define ERROR_DS_DRA_BAD_INSTANCE_TYPE 8445
1271 #define ERROR_DS_DRA_OUT_OF_MEM 8446
1272 #define ERROR_DS_DRA_MAIL_PROBLEM 8447
1273 #define ERROR_DS_DRA_REF_ALREADY_EXISTS 8448
```


1274	#define	ERROR_DS_DRA_REF_NOT_FOUND	8449
1275	#define	ERROR_DS_DRA_OBJ_IS_REP_SOURCE	8450
1276	#define	ERROR_DS_DRA_DB_ERROR	8451
1277	#define	ERROR_DS_DRA_NO_REPLICA	8452
1278	#define	ERROR_DS_DRA_ACCESS_DENIED	8453
1279	#define	ERROR_DS_DRA_NOT_SUPPORTED	8454
1280	#define	ERROR_DS_DRA_RPC_CANCELLED	8455
1281	#define	ERROR_DS_DRA_SOURCE_DISABLED	8456
1282	#define	ERROR_DS_DRA_SINK_DISABLED	8457
1283	#define	ERROR_DS_DRA_NAME_COLLISION	8458
1284	#define	ERROR_DS_DRA_SOURCE_REINSTALLED	8459
1285	#define	ERROR_DS_DRA_MISSING_PARENT	8460
1286	#define	ERROR_DS_DRA_PREEMPTED	8461
1287	#define	ERROR_DS_DRA_ABANDON_SYNC	8462
1288	#define	ERROR_DS_DRA_SHUTDOWN	8463
1289	#define	ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET	8464
1290	#define	ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA	8465
1291	#define	ERROR_DS_DRA_EXTN_CONNECTION_FAILED	8466
1292	#define	ERROR_DS_INSTALL_SCHEMA_MISMATCH	8467
1293	#define	ERROR_DS_DUP_LINK_ID	8468
1294	#define	ERROR_DS_NAME_ERROR_RESOLVING	8469
1295	#define	ERROR_DS_NAME_ERROR_NOT_FOUND	8470
1296	#define	ERROR_DS_NAME_ERROR_NOT_UNIQUE	8471
1297	#define	ERROR_DS_NAME_ERROR_NO_MAPPING	8472
1298	#define	ERROR_DS_NAME_ERROR_DOMAIN_ONLY	8473
1299	#define	ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING	8474
1300	#define	ERROR_DS_CONSTRUCTED_ATT_MOD	8475
1301	#define	ERROR_DS_WRONG_OM_OBJ_CLASS	8476
1302	#define	ERROR_DS_DRA_REPL_PENDING	8477
1303	#define	ERROR_DS_DS_REQUIRED	8478
1304	#define	ERROR_DS_INVALID_LDAP_DISPLAY_NAME	8479
1305	#define	ERROR_DS_NON_BASE_SEARCH	8480
1306	#define	ERROR_DS_CANT_RETRIEVE_ATTS	8481
1307	#define	ERROR_DS_BACKLINK_WITHOUT_LINK	8482
1308	#define	ERROR_DS_EPOCH_MISMATCH	8483
1309	#define	ERROR_DS_SRC_NAME_MISMATCH	8484
1310	#define	ERROR_DS_SRC_AND_DST_NC_IDENTICAL	8485
1311	#define	ERROR_DS_DST_NC_MISMATCH	8486
1312	#define	ERROR_DS_NOT_AUTHORITIVE_FOR_DST_NC	8487
1313	#define	ERROR_DS_SRC_GUID_MISMATCH	8488
1314	#define	ERROR_DS_CANT_MOVE_DELETED_OBJECT	8489
1315	#define	ERROR_DS_PDC_OPERATION_IN_PROGRESS	8490
1316	#define	ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD	8491
1317	#define	ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION	8492
1318	#define	ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS	8493
1319	#define	ERROR_DS_NC_MUST_HAVE_NC_PARENT	8494
1320	#define	ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE	8495
1321	#define	ERROR_DS_DST_DOMAIN_NOT_NATIVE	8496
1322	#define	ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER	8497
1323	#define	ERROR_DS_CANT_MOVE_ACCOUNT_GROUP	8498
1324	#define	ERROR_DS_CANT_MOVE_RESOURCE_GROUP	8499
1325	#define	ERROR_DS_INVALID_SEARCH_FLAG	8500
1326	#define	ERROR_DS_NO_TREE_DELETE_ABOVE_NC	8501
1327	#define	ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE	8502
1328	#define	ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE	8503
1329	#define	ERROR_DS_SAM_INIT_FAILURE	8504
1330	#define	ERROR_DS_SENSITIVE_GROUP_VIOLATION	8505
1331	#define	ERROR_DS_CANT_MOD_PRIMARYGROUPID	8506
1332	#define	ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD	8507
1333	#define	ERROR_DS_NONSAFE_SCHEMA_CHANGE	8508
1334	#define	ERROR_DS_SCHEMA_UPDATE_DISALLOWED	8509
1335	#define	ERROR_DS_CANT_CREATE_UNDER_SCHEMA	8510
1336	#define	ERROR_DS_INSTALL_NO_SRC_SCH_VERSION	8511
1337	#define	ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE	8512
1338	#define	ERROR_DS_INVALID_GROUP_TYPE	8513
1339	#define	ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN	8514
1340	#define	ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN	8515
1341	#define	ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER	8516
1342	#define	ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER	8517
1343	#define	ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER	8518
1344	#define	ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER	8519
1345	#define	ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBER	8520
1346	#define	ERROR_DS_HAVE_PRIMARY_MEMBERS	8521
1347	#define	ERROR_DS_STRING_SD_CONVERSION_FAILED	8522
1348	#define	ERROR_DS_NAMING_MASTER_GC	8523
1349	#define	ERROR_DS_LOOKUP_FAILURE	8524
1350	#define	ERROR_DS_COULDNT_UPDATE_SPNS	8525
1351	#define	ERROR_DS_CANT_RETRIEVE_SD	8526
1352	#define	ERROR_DS_KEY_NOT_UNIQUE	8527
1353	#define	ERROR_DS_WRONG_LINKED_ATT_SYNTAX	8528
1354	#define	ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD	8529
1355	#define	ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY	8530
1356	#define	ERROR_DS_CANT_START	8531
1357	#define	ERROR_DS_INIT_FAILURE	8532
1358	#define	ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION	8533
1359	#define	ERROR_DS_SOURCE_DOMAIN_IN_FOREST	8534
1360	#define	ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST	8535

```
1361 #define ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED 8536
1362 #define ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN 8537
1363 #define ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER 8538
1364 #define ERROR_DS_SRC_SID_EXISTS_IN_FOREST 8539
1365 #define ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH 8540
1366 #define ERROR_DS_INIT_FAILURE 8541
1367 #define ERROR_DS_DRA_SCHEMA_INFO_SHIP 8542
1368 #define ERROR_DS_DRA_SCHEMA_CONFLICT 8543
1369 #define ERROR_DS_DRA_EARLIER_SCHEMA_CONFLICT 8544
1370 #define ERROR_DS_DRA_OBJ_NC_MISMATCH 8545
1371 #define ERROR_DS_NC_STILL_HAS_DSAS 8546
1372 #define ERROR_DS_GC_REQUIRED 8547
1373 #define ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY 8548
1374 #define ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS 8549
1375 #define ERROR_DS_CANT_ADD_TO_GC 8550
1376 #define ERROR_DS_NO_CHECKPOINT_WITH_PDC 8551
1377 #define ERROR_DS_SOURCE_AUDITING_NOT_ENABLED 8552
1378 #define ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC 8553
1379 #define ERROR_DS_INVALID_NAME_FOR_SPN 8554
1380 #define ERROR_DS_FILTER_USES_CONSTRUCTED_ATTRS 8555
1381 #define ERROR_DS_UNICODEPWD_NOT_IN_QUOTES 8556
1382 #define ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED 8557
1383 #define ERROR_DS_MUST_BE_RUN_ON_DST_DC 8558
1384 #define ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER 8559
1385 #define ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ 8560
1386 #define DNS_ERROR_RCODE_FORMAT_ERROR 9001
1387 #define DNS_ERROR_RCODE_SERVER_FAILURE 9002
1388 #define DNS_ERROR_RCODE_NAME_ERROR 9003
1389 #define DNS_ERROR_RCODE_NOT_IMPLEMENTED 9004
1390 #define DNS_ERROR_RCODE_REFUSED 9005
1391 #define DNS_ERROR_RCODE_YXDOMAIN 9006
1392 #define DNS_ERROR_RCODE_YXRRSET 9007
1393 #define DNS_ERROR_RCODE_NXRRSET 9008
1394 #define DNS_ERROR_RCODE_NOTAUTH 9009
1395 #define DNS_ERROR_RCODE_NOTZONE 9010
1396 #define DNS_ERROR_RCODE_BADSIG 9016
1397 #define DNS_ERROR_RCODE_BADKEY 9017
1398 #define DNS_ERROR_RCODE_BADTIME 9018
1399 #define DNS_INFO_NO_RECORDS 9501
1400 #define DNS_ERROR_BAD_PACKET 9502
1401 #define DNS_ERROR_NO_PACKET 9503
1402 #define DNS_ERROR_RCODE 9504
1403 #define DNS_ERROR_UNSECURE_PACKET 9505
1404 #define DNS_ERROR_INVALID_TYPE 9551
1405 #define DNS_ERROR_INVALID_IP_ADDRESS 9552
1406 #define DNS_ERROR_INVALID_PROPERTY 9553
1407 #define DNS_ERROR_TRY_AGAIN_LATER 9554
1408 #define DNS_ERROR_NOT_UNIQUE 9555
1409 #define DNS_ERROR_NON_RFC_NAME 9556
1410 #define DNS_STATUS_FQDN 9557
1411 #define DNS_STATUS_DOTTED_NAME 9558
1412 #define DNS_STATUS_SINGLE_PART_NAME 9559
1413 #define DNS_ERROR_INVALID_NAME_CHAR 9560
1414 #define DNS_ERROR_NUMERIC_NAME 9561
1415 #define DNS_ERROR_ZONE_DOES_NOT_EXIST 9601
1416 #define DNS_ERROR_NO_ZONE_INFO 9602
1417 #define DNS_ERROR_INVALID_ZONE_OPERATION 9603
1418 #define DNS_ERROR_ZONE_CONFIGURATION_ERROR 9604
1419 #define DNS_ERROR_ZONE_HAS_NO_SOA_RECORD 9605
1420 #define DNS_ERROR_ZONE_HAS_NO_NS_RECORDS 9606
1421 #define DNS_ERROR_ZONE_LOCKED 9607
1422 #define DNS_ERROR_ZONE_CREATION_FAILED 9608
1423 #define DNS_ERROR_ZONE_ALREADY_EXISTS 9609
1424 #define DNS_ERROR_AUTOZONE_ALREADY_EXISTS 9610
1425 #define DNS_ERROR_INVALID_ZONE_TYPE 9611
1426 #define DNS_ERROR_SECONDARY_REQUIRES_MASTER_IP 9612
1427 #define DNS_ERROR_ZONE_NOT_SECONDARY 9613
1428 #define DNS_ERROR_NEED_SECONDARY_ADDRESSES 9614
1429 #define DNS_ERROR_WINS_INIT_FAILED 9615
1430 #define DNS_ERROR_NEED_WINS_SERVERS 9616
1431 #define DNS_ERROR_NBSTAT_INIT_FAILED 9617
1432 #define DNS_ERROR_SOA_DELETE_INVALID 9618
1433 #define DNS_ERROR_PRIMARY_REQUIRES_DATAFILE 9651
1434 #define DNS_ERROR_INVALID_DATAFILE_NAME 9652
1435 #define DNS_ERROR_DATAFILE_OPEN_FAILURE 9653
1436 #define DNS_ERROR_FILE_WRITEBACK_FAILED 9654
1437 #define DNS_ERROR_DATAFILE_PARSING 9655
1438 #define DNS_ERROR_RECORD_DOES_NOT_EXIST 9701
1439 #define DNS_ERROR_RECORD_FORMAT 9702
1440 #define DNS_ERROR_NODE_CREATION_FAILED 9703
1441 #define DNS_ERROR_UNKNOWN_RECORD_TYPE 9704
1442 #define DNS_ERROR_RECORD_TIMED_OUT 9705
1443 #define DNS_ERROR_NAME_NOT_IN_ZONE 9706
1444 #define DNS_ERROR_CNAME_LOOP 9707
1445 #define DNS_ERROR_NODE_IS_CNAME 9708
1446 #define DNS_ERROR_CNAME_COLLISION 9709
1447 #define DNS_ERROR_RECORD_ONLY_AT_ZONE_ROOT 9710
```

```

1448 #define DNS_ERROR_RECORD_ALREADY_EXISTS 9711
1449 #define DNS_ERROR_SECONDARY_DATA 9712
1450 #define DNS_ERROR_NO_CREATE_CACHE_DATA 9713
1451 #define DNS_ERROR_NAME_DOES_NOT_EXIST 9714
1452 #define DNS_WARNING_PTR_CREATE_FAILED 9715
1453 #define DNS_WARNING_DOMAIN_UNDELETED 9716
1454 #define DNS_ERROR_DS_UNAVAILABLE 9717
1455 #define DNS_ERROR_DS_ZONE_ALREADY_EXISTS 9718
1456 #define DNS_ERROR_NO_BOOTFILE_IF_DS_ZONE 9719
1457 #define DNS_INFO_AXFR_COMPLETE 9751
1458 #define DNS_ERROR_AXFR 9752
1459 #define DNS_INFO_ADDED_LOCAL_WINS 9753
1460 #define DNS_STATUS_CONTINUE_NEEDED 9801
1461 #define DNS_ERROR_NO_TCPIP 9851
1462 #define DNS_ERROR_NO_DNS_SERVERS 9852
1463
1464 /* HRESULT values for OLE, SHELL and other Interface stuff */
1465 /* the codes 4000-40ff are reserved for OLE */
1466 #define NOERROR 0L
1467 #define S_OK ((HRESULT) 0L)
1468 #define S_FALSE ((HRESULT) 1L)
1469
1470 #define E_PENDING 0x8000000AL
1471
1472
1473 #define E_NOTIMPL 0x80004001L
1474 #define E_NOINTERFACE 0x80004002L
1475 #define E_POINTER 0x80004003L
1476 #define E_ABORT 0x80004004L
1477 #define E_FAIL 0x80004005L
1478 /* FIXME: E_UNSPEC is not a standard value but it is used by
1479 * FileMoniker, IoleLink and DoDragDrop as a return value.
1480 */
1481 #define E_UNSPEC E_FAIL
1482
1483
1484 #define CO_E_INIT_TLS 0x80004006L
1485 #define CO_E_INIT_SHARED_ALLOCATOR 0x80004007L
1486 #define CO_E_INIT_MEMORY_ALLOCATOR 0x80004008L
1487 #define CO_E_INIT_CLASS_CACHE 0x80004009L
1488 #define CO_E_INIT_RPC_CHANNEL 0x8000400AL
1489 #define CO_E_INIT_TLS_SET_CHANNEL_CONTROL 0x8000400BL
1490 #define CO_E_INIT_TLS_CHANNEL_CONTROL 0x8000400CL
1491 #define CO_E_INIT_UNACCEPTED_USER_ALLOCATOR 0x8000400DL
1492 #define CO_E_INIT_SCM_MUTEX_EXISTS 0x8000400EL
1493 #define CO_E_INIT_SCM_FILE_MAPPING_EXISTS 0x8000400FL
1494 #define CO_E_INIT_SCM_MAP_VIEW_OF_FILE 0x80004010L
1495 #define CO_E_INIT_SCM_EXEC_FAILURE 0x80004011L
1496 #define CO_E_INIT_ONLY_SINGLE_THREADED 0x80004012L
1497
1498 #define E_UNEXPECTED 0x8000FFFFL
1499
1500 #define RPC_E_CALL_REJECTED 0x80010001L
1501 #define RPC_E_CALL_CANCELED 0x80010002L
1502 #define RPC_E_CANTPOST_INSENDCALL 0x80010003L
1503 #define RPC_E_CANTCALLOUT_INASYNCALL 0x80010004L
1504 #define RPC_E_CANTCALLOUT_INEXTERNALCALL 0x80010005L
1505 #define RPC_E_CONNECTION_TERMINATED 0x80010006L
1506 #define RPC_E_SERVER_DIED 0x80010007L
1507 #define RPC_E_CLIENT_DIED 0x80010008L
1508 #define RPC_E_INVALID_DATAPACKET 0x80010009L
1509 #define RPC_E_CANTTRANSMIT_CALL 0x8001000AL
1510 #define RPC_E_CLIENT_CANTMARSHAL_DATA 0x8001000BL
1511 #define RPC_E_CLIENT_CANTUNMARSHAL_DATA 0x8001000CL
1512 #define RPC_E_SERVER_CANTMARSHAL_DATA 0x8001000DL
1513 #define RPC_E_SERVER_CANTUNMARSHAL_DATA 0x8001000EL
1514 #define RPC_E_INVALID_DATA 0x8001000FL
1515 #define RPC_E_INVALID_PARAMETER 0x80010010L
1516 #define RPC_E_CANTCALLOUT_AGAIN 0x80010011L
1517 #define RPC_E_SERVER_DIED_DNE 0x80010012L
1518 #define RPC_E_SYS_CALL_FAILED 0x80010100L
1519 #define RPC_E_OUT_OF_RESOURCES 0x80010101L
1520 #define RPC_E_ATTEMPTED_MULTITHREAD 0x80010102L
1521 #define RPC_E_NOT_REGISTERED 0x80010103L
1522 #define RPC_E_FAULT 0x80010104L
1523 #define RPC_E_SERVERFAULT 0x80010105L
1524 #define RPC_E_CHANGED_MODE 0x80010106L
1525 #define RPC_E_INVALIDMETHOD 0x80010107L
1526 #define RPC_E_DISCONNECTED 0x80010108L
1527 #define RPC_E_RETRY 0x80010109L
1528 #define RPC_E_SERVERCALL_RETRYLATER 0x8001010AL
1529 #define RPC_E_SERVERCALL_REJECTED 0x8001010BL
1530 #define RPC_E_INVALID_CALLDATA 0x8001010CL
1531 #define RPC_E_CANTCALLOUT_ININPUTSYNCCALL 0x8001010DL
1532 #define RPC_E_WRONG_THREAD 0x8001010EL
1533 #define RPC_E_THREAD_NOT_INIT 0x8001010FL
1534 #define RPC_E_VERSION_MISMATCH 0x80010110L

```

```
1535 #define RPC_E_INVALID_HEADER 0x80010111L
1536 #define RPC_E_INVALID_EXTENSION 0x80010112L
1537 #define RPC_E_INVALID_IPID 0x80010113L
1538 #define RPC_E_INVALID_OBJECT 0x80010114L
1539 #define RPC_S_CALLPENDING 0x80010115L
1540 #define RPC_S_WAITONTIMER 0x80010116L
1541 #define RPC_E_CALL_COMPLETE 0x80010117L
1542 #define RPC_E_UNSECURE_CALL 0x80010118L
1543 #define RPC_E_TOO_LATE 0x80010119L
1544 #define RPC_E_NO_GOOD_SECURITY_PACKAGES 0x8001011AL
1545 #define RPC_E_ACCESS_DENIED 0x8001011BL
1546 #define RPC_E_REMOTE_DISABLED 0x8001011CL
1547 #define RPC_E_INVALID_OBJREF 0x8001011DL
1548 #define RPC_E_NO_CONTEXT 0x8001011EL
1549 #define RPC_E_TIMEOUT 0x8001011FL
1550 #define RPC_E_NO_SYNC 0x80010120L
1551 #define RPC_E_UNEXPECTED 0x8001FFFFL
1552
1553 #define DISP_E_UNKNOWNINTERFACE 0x80020001L
1554 #define DISP_E_MEMBERNOTFOUND 0x80020003L
1555 #define DISP_E_PARAMNOTFOUND 0x80020004L
1556 #define DISP_E_TYPEMISMATCH 0x80020005L
1557 #define DISP_E_UNKNOWNNAME 0x80020006L
1558 #define DISP_E_NONAMEDARGS 0x80020007L
1559 #define DISP_E_BADVARTYPE 0x80020008L
1560 #define DISP_E_EXCEPTION 0x80020009L
1561 #define DISP_E_OVERFLOW 0x8002000AL
1562 #define DISP_E_BADINDEX 0x8002000BL
1563 #define DISP_E_UNKNOWNLCID 0x8002000CL
1564 #define DISP_E_ARRAYISLOCKED 0x8002000DL
1565 #define DISP_E_BADPARAMCOUNT 0x8002000EL
1566 #define DISP_E_PARAMNOTOPTIONAL 0x8002000FL
1567 #define DISP_E_BADCALLEE 0x80020010L
1568 #define DISP_E_NOTACCOLLECTION 0x80020011L
1569 #define DISP_E_DIVBYZERO 0x80020012L
1570
1571 #define TYPE_E_BUFFERTOOSMALL 0x80028016L
1572 #define TYPE_E_FIELDNOTFOUND 0x80028017L
1573 #define TYPE_E_INVDATAREAD 0x80028018L
1574 #define TYPE_E_UNSUPFORMAT 0x80028019L
1575 #define TYPE_E_REGISTRYACCESS 0x8002801CL
1576 #define TYPE_E_LIBNOTREGISTERED 0x8002801DL
1577 #define TYPE_E_UNDEFINEDTYPE 0x80028027L
1578 #define TYPE_E_QUALIFIEDNAMEDISALLOWED 0x80028028L
1579 #define TYPE_E_INVALIDSTATE 0x80028029L
1580 #define TYPE_E_WRONGTYPEKIND 0x8002802AL
1581 #define TYPE_E_ELEMENTNOTFOUND 0x8002802BL
1582 #define TYPE_E_AMBIGUOUSNAME 0x8002802CL
1583 #define TYPE_E_NAMECONFLICT 0x8002802DL
1584 #define TYPE_E_UNKNOWNLCID 0x8002802EL
1585 #define TYPE_E_DLLFUNCTIONNOTFOUND 0x8002802FL
1586 #define TYPE_E_BADMODULEKIND 0x800288BDL
1587 #define TYPE_E_SIZETOOBIG 0x800288C5L
1588 #define TYPE_E_DUPLICATEID 0x800288C6L
1589 #define TYPE_E_INVALIDID 0x800288CFL
1590 #define TYPE_E_TYPEMISMATCH 0x80028CA0L
1591 #define TYPE_E_OUTOFBOUNDS 0x80028CA1L
1592 #define TYPE_E_IOERROR 0x80028CA2L
1593 #define TYPE_E_CANTCREATETMPFILE 0x80028CA3L
1594 #define TYPE_E_CANTLOADLIBRARY 0x80029C4AL
1595 #define TYPE_E_INCONSISTENTPROPFUNCS 0x80029C83L
1596 #define TYPE_E_CIRCULARTYPE 0x80029C84L
1597
1598 #define STG_S_CONVERTED 0x00030200L
1599 #define STG_S_BLOCK 0x00030201L
1600 #define STG_S_RETRYNOW 0x00030202L
1601 #define STG_S_MONITORING 0x00030203L
1602 #define STG_S_MULTIPLEOPENS 0x00030204L
1603 #define STG_S_CONSOLIDATIONFAILED 0x00030205L
1604 #define STG_S_CANNOTCONSOLIDATE 0x00030206L
1605
1606 #define STG_E_INVALIDFUNCTION 0x80030001L
1607 #define STG_E_FILENOTFOUND 0x80030002L
1608 #define STG_E_PATHNOTFOUND 0x80030003L
1609 #define STG_E_TOOMANYOPENFILES 0x80030004L
1610 #define STG_E_ACCESSDENIED 0x80030005L
1611 #define STG_E_INVALIDHANDLE 0x80030006L
1612 #define STG_E_INSUFFICIENTMEMORY 0x80030008L
1613 #define STG_E_INVALIDPOINTER 0x80030009L
1614 #define STG_E_NOMOREFILES 0x80030012L
1615 #define STG_E_DISKISWRITEPROTECTED 0x80030013L
1616 #define STG_E_SEEKERROR 0x80030019L
1617 #define STG_E_WRITEFAULT 0x8003001DL
1618 #define STG_E_READFAULT 0x8003001EL
1619 #define STG_E_SHAREVIOLATION 0x80030020L
1620 #define STG_E_LOCKVIOLATION 0x80030021L
1621 #define STG_E_FILEALREADYEXISTS 0x80030050L
```

```
1622 #define STG_E_INVALIDPARAMETER 0x80030057L
1623 #define STG_E_MEDIUMFULL 0x80030070L
1624 #define STG_E_ABNORMALAPIEXIT 0x800300FAL
1625 #define STG_E_INVALIDHEADER 0x800300FBL
1626 #define STG_E_INVALIDNAME 0x800300FCL
1627 #define STG_E_UNKNOWN 0x800300FDL
1628 #define STG_E_UNIMPLEMENTEDFUNCTION 0x800300FEL
1629 #define STG_E_INVALIDFLAG 0x800300FFL
1630 #define STG_E_INUSE 0x80030100L
1631 #define STG_E_NOTCURRENT 0x80030101L
1632 #define STG_E_REVERTED 0x80030102L
1633 #define STG_E_CANTSAVE 0x80030103L
1634 #define STG_E_OLDFORMAT 0x80030104L
1635 #define STG_E_OLDDLL 0x80030105L
1636 #define STG_E_SHAREREQUIRED 0x80030106L
1637 #define STG_E_NOTFILEBASEDSTORAGE 0x80030107L
1638 #define STG_E_EXTANTMARSHALLINGS 0x80030108L
1639
1640 #define OLE_S_FIRST 0x00040000L
1641 #define OLE_S_USEREG 0x00040000L
1642 #define OLE_S_STATIC 0x00040001L
1643 #define OLE_S_MAC_CLIPFORMAT 0x00040002L
1644 #define OLE_S_LAST 0x000400FFL
1645
1646 #define OLE_E_FIRST 0x80040000L
1647 #define OLE_E_OLEVERB 0x80040000L
1648 #define OLE_E_ADVF 0x80040001L
1649 #define OLE_E_ENUM_NOMORE 0x80040002L
1650 #define OLE_E_ADVISENOTSUPPORTED 0x80040003L
1651 #define OLE_E_NOCONNECTION 0x80040004L
1652 #define OLE_E_NOTRUNNING 0x80040005L
1653 #define OLE_E_NOCACHE 0x80040006L
1654 #define OLE_E_BLANK 0x80040007L
1655 #define OLE_E_CLASSDIFF 0x80040008L
1656 #define OLE_E_CANT_GETMONIKER 0x80040009L
1657 #define OLE_E_CANT_BINDTOSOURCE 0x8004000AL
1658 #define OLE_E_STATIC 0x8004000BL
1659 #define OLE_E_PROMPTSAVECANCELLED 0x8004000CL
1660 #define OLE_E_INVALIDRECT 0x8004000DL
1661 #define OLE_E_WRONGCOMPOBJ 0x8004000EL
1662 #define OLE_E_INVALIDHWND 0x8004000FL
1663 #define OLE_E_NOT_INPLACEACTIVE 0x80040010L
1664 #define OLE_E_CANTCONVERT 0x80040011L
1665 #define OLE_E_NOSTORAGE 0x80040012L
1666 #define DV_E_FORMATETC 0x80040064L
1667 #define DV_E_DVTARGETDEVICE 0x80040065L
1668 #define DV_E_STGMEDIUM 0x80040066L
1669 #define DV_E_STATDATA 0x80040067L
1670 #define DV_E_LINDEX 0x80040068L
1671 #define DV_E_TYMED 0x80040069L
1672 #define DV_E_CLIPFORMAT 0x8004006AL
1673 #define DV_E_DVASPECT 0x8004006BL
1674 #define DV_E_DVTARGETDEVICE_SIZE 0x8004006CL
1675 #define DV_E_NOVIEWOBJECT 0x8004006DL
1676 #define OLE_E_LAST 0x800400FFL
1677
1678 #define DRAGDROP_S_FIRST 0x00040100L
1679 #define DRAGDROP_S_DROP 0x00040100L
1680 #define DRAGDROP_S_CANCEL 0x00040101L
1681 #define DRAGDROP_S_USEDEFAULTCURSORS 0x00040102L
1682 #define DRAGDROP_S_LAST 0x0004010FL
1683
1684 #define DRAGDROP_E_FIRST 0x80040100L
1685 #define DRAGDROP_E_NOTREGISTERED 0x80040100L
1686 #define DRAGDROP_E_ALREADYREGISTERED 0x80040101L
1687 #define DRAGDROP_E_INVALIDHWND 0x80040102L
1688 #define DRAGDROP_E_LAST 0x8004010FL
1689
1690
1691 #define CLASSFACTORY_S_FIRST 0x00040110L
1692 #define CLASSFACTORY_S_LAST 0x0004011FL
1693
1694 #define CLASSFACTORY_E_FIRST 0x80040110L
1695 #define CLASS_E_NOAGGREGATION 0x80040110L
1696 #define CLASS_E_CLASSNOTAVAILABLE 0x80040111L
1697 #define CLASS_E_NOTLICENSED 0x80040112L
1698 #define CLASSFACTORY_E_LAST 0x8004011FL
1699
1700 #define MARSHAL_S_FIRST 0x00040120L
1701 #define MARSHAL_S_LAST 0x0004012FL
1702
1703 #define MARSHAL_E_FIRST 0x80040120L
1704 #define MARSHAL_E_LAST 0x8004012FL
1705
1706 #define DATA_S_FIRST 0x00040130L
1707 #define DATA_S_SAMEFORMATETC 0x00040130L
1708 #define DATA_S_LAST 0x0004013FL
```

```
1709
1710 #define DATA_E_FIRST 0x80040130L
1711 #define DATA_E_LAST 0x8004013FL
1712
1713 #define VIEW_S_FIRST 0x00040140L
1714 #define VIEW_S_ALREADY_FROZEN 0x00040140L
1715 #define VIEW_S_LAST 0x0004014FL
1716
1717 #define VIEW_E_FIRST 0x80040140L
1718 #define VIEW_E_DRAW 0x80040140L
1719 #define VIEW_E_LAST 0x8004014FL
1720
1721 #define REGDB_S_FIRST 0x00040150L
1722 #define REGDB_S_LAST 0x0004015FL
1723
1724 #define REGDB_E_FIRST 0x80040150L
1725 #define REGDB_E_READREGDB 0x80040150L
1726 #define REGDB_E_WRITEREGDB 0x80040151L
1727 #define REGDB_E_KEYMISSING 0x80040152L
1728 #define REGDB_E_INVALIDVALUE 0x80040153L
1729 #define REGDB_E_CLASSNOTREG 0x80040154L
1730 #define REGDB_E_IIDNOTREG 0x80040155L
1731 #define REGDB_E_LAST 0x8004015FL
1732
1733 #define CACHE_S_FIRST 0x00040170L
1734 #define CACHE_S_FORMATETC_NOTSUPPORTED 0x00040170L
1735 #define CACHE_S_SAMECACHE 0x00040171L
1736 #define CACHE_S_SOMECACHES_NOTUPDATED 0x00040172L
1737 #define CACHE_S_LAST 0x0004017FL
1738
1739 #define CACHE_E_FIRST 0x80040170L
1740 #define CACHE_E_NOCACHE_UPDATED 0x80040170L
1741 #define CACHE_E_LAST 0x8004017FL
1742
1743 #define OLEOBJ_S_FIRST 0x00040180L
1744 #define OLEOBJ_S_INVALIDVERB 0x00040180L
1745 #define OLEOBJ_S_CANNOT_DOVERB_NOW 0x00040181L
1746 #define OLEOBJ_S_INVALIDHWND 0x00040182L
1747 #define OLEOBJ_S_LAST 0x0004018FL
1748
1749 #define OLEOBJ_E_FIRST 0x80040180L
1750 #define OLEOBJ_E_NOVERBS 0x80040180L
1751 #define OLEOBJ_E_INVALIDVERB 0x80040181L
1752 #define OLEOBJ_E_LAST 0x8004018FL
1753
1754 #define CLIENTSITE_S_FIRST 0x00040190L
1755 #define CLIENTSITE_S_LAST 0x0004019FL
1756
1757 #define CLIENTSITE_E_FIRST 0x80040190L
1758 #define CLIENTSITE_E_LAST 0x8004019FL
1759
1760 #define INPLACE_S_FIRST 0x000401A0L
1761 #define INPLACE_S_TRUNCATED 0x000401A0L
1762 #define INPLACE_S_LAST 0x000401AFL
1763
1764 #define INPLACE_E_FIRST 0x800401A0L
1765 #define INPLACE_E_NOTUNDOABLE 0x800401A0L
1766 #define INPLACE_E_NOTOOLSPACE 0x800401A1L
1767 #define INPLACE_E_LAST 0x800401AFL
1768
1769 #define ENUM_S_FIRST 0x000401B0L
1770 #define ENUM_S_LAST 0x000401BFL
1771
1772 #define ENUM_E_FIRST 0x800401B0L
1773 #define ENUM_E_LAST 0x800401BFL
1774
1775 #define CONVERT10_S_FIRST 0x000401C0L
1776 #define CONVERT10_S_NO_PRESENTATION 0x000401C0L
1777 #define CONVERT10_S_LAST 0x000401CFL
1778
1779 #define CONVERT10_E_FIRST 0x800401C0L
1780 #define CONVERT10_E_OLESTREAM_GET 0x800401C0L
1781 #define CONVERT10_E_OLESTREAM_PUT 0x800401C1L
1782 #define CONVERT10_E_OLESTREAM_FMT 0x800401C2L
1783 #define CONVERT10_E_OLESTREAM_BITMAP_TO_DIB 0x800401C3L
1784 #define CONVERT10_E_STG_FMT 0x800401C4L
1785 #define CONVERT10_E_STG_NO_STD_STREAM 0x800401C5L
1786 #define CONVERT10_E_STG_DIB_TO_BITMAP 0x800401C6L
1787 #define CONVERT10_E_LAST 0x800401CFL
1788
1789 #define CLIPBRD_S_FIRST 0x000401D0L
1790 #define CLIPBRD_S_LAST 0x000401DFL
1791
1792 #define CLIPBRD_E_FIRST 0x800401D0L
1793 #define CLIPBRD_E_LAST 0x800401DFL
1794 #define CLIPBRD_E_CANT_OPEN 0x800401D0L
1795 #define CLIPBRD_E_CANT_EMPTY 0x800401D1L
```



```
1796 #define CLIPBRD_E_CANT_SET 0x800401D2L
1797 #define CLIPBRD_E_BAD_DATA 0x800401D3L
1798 #define CLIPBRD_E_CANT_CLOSE 0x800401D4L
1799
1800 #define MK_S_FIRST 0x000401E0L
1801 #define MK_S_REDUCED_TO_SELF 0x000401E2L
1802 #define MK_S_ME 0x000401E4L
1803 #define MK_S_HIM 0x000401E5L
1804 #define MK_S_US 0x000401E6L
1805 #define MK_S_MONIKERALREADYREGISTERED 0x000401E7L
1806 #define MK_S_LAST 0x000401EFL
1807
1808 #define MK_E_FIRST 0x800401E0L
1809 #define MK_E_CONNECTMANUALLY 0x800401E0L
1810 #define MK_E_EXCEEDEDDEADLINE 0x800401E1L
1811 #define MK_E_NEEDGENERIC 0x800401E2L
1812 #define MK_E_UNAVAILABLE 0x800401E3L
1813 #define MK_E_SYNTAX 0x800401E4L
1814 #define MK_E_NOOBJECT 0x800401E5L
1815 #define MK_E_INVALIDEXTENSION 0x800401E6L
1816 #define MK_E_INTERMEDIATEINTERFACENOTSUPPORTED 0x800401E7L
1817 #define MK_E_NOTBINDABLE 0x800401E8L
1818 #define MK_E_NOTBOUND 0x800401E9L
1819 #define MK_E_CANTOPENFILE 0x800401EAL
1820 #define MK_E_MUSTBOTHERUSER 0x800401EBL
1821 #define MK_E_NOINVERSE 0x800401ECL
1822 #define MK_E_NOSTORAGE 0x800401EDL
1823 #define MK_E_NOPREFIX 0x800401EEL
1824 #define MK_E_ENUMERATION_FAILED 0x800401EFL
1825 #define MK_E_LAST 0x800401EFL
1826
1827 #define CO_S_FIRST 0x000401F0L
1828 #define CO_S_LAST 0x000401FFL
1829
1830 #define CO_E_FIRST 0x800401F0L
1831 #define CO_E_NOTINITIALIZED 0x800401F0L
1832 #define CO_E_ALREADYINITIALIZED 0x800401F1L
1833 #define CO_E_CANTDETERMINECLASS 0x800401F2L
1834 #define CO_E_CLASSSTRING 0x800401F3L
1835 #define CO_E_IIDSTRING 0x800401F4L
1836 #define CO_E_APPNOTFOUND 0x800401F5L
1837 #define CO_E_APPSINGLEUSE 0x800401F6L
1838 #define CO_E_ERRORINAPP 0x800401F7L
1839 #define CO_E_DLLNOTFOUND 0x800401F8L
1840 #define CO_E_ERRORINDLL 0x800401F9L
1841 #define CO_E_WRONGOSFORAPP 0x800401FAL
1842 #define CO_E_OBJNOTREG 0x800401FBL
1843 #define CO_E_OBJISREG 0x800401FCL
1844 #define CO_E_OBJNOTCONNECTED 0x800401FDL
1845 #define CO_E_APPDIDNTREG 0x800401FEL
1846 #define CO_E_RELEASED 0x800401FFL
1847 #define CO_E_LAST 0x800401FFL
1848 #define CO_E_FAILEDTOIMPERSONATE 0x80040200L
1849 #define CO_E_FAILEDTOGETSECCTX 0x80040201L
1850 #define CO_E_FAILEDTOOPENTHREADTOKEN 0x80040202L
1851 #define CO_E_FAILEDTOGETTOKENINFO 0x80040203L
1852 #define CO_E_TRUSTEEDOESNTMATCHCLIENT 0x80040204L
1853 #define CO_E_FAILEDTOQUERYCLIENTBLANKET 0x80040205L
1854 #define CO_E_FAILEDTOSETDACL 0x80040206L
1855 #define CO_E_ACCESSCHECKFAILED 0x80040207L
1856 #define CO_E_NETACCESSAPIFAILED 0x80040208L
1857 #define CO_E_WRONGTRUSTEENAMESYNTAX 0x80040209L
1858 #define CO_E_INVALIDSID 0x8004020AL
1859 #define CO_E_CONVERSIONFAILED 0x8004020BL
1860 #define CO_E_NOMATCHINGSIDFOUND 0x8004020CL
1861 #define CO_E_LOOKUPACCSIDFAILED 0x8004020DL
1862 #define CO_E_NOMATCHINGNAMEFOUND 0x8004020EL
1863 #define CO_E_LOOKUPACCNAMFAILED 0x8004020FL
1864 #define CO_E_SETSERLHNDLFAILED 0x80040210L
1865 #define CO_E_FAILEDTOGETWINDIR 0x80040211L
1866 #define CO_E_PATHTOOLONG 0x80040212L
1867 #define CO_E_FAILEDTOGENUUID 0x80040213L
1868 #define CO_E_FAILEDTOCREATEFILE 0x80040214L
1869 #define CO_E_FAILEDTOCLOSEHANDLE 0x80040215L
1870 #define CO_E_EXCEEDSYSACLLIMIT 0x80040216L
1871 #define CO_E_ACESINWRONGORDER 0x80040217L
1872 #define CO_E_INCOMPATIBLESTREAMVERSION 0x80040218L
1873 #define CO_E_FAILEDTOOPENPROCESSTOKEN 0x80040219L
1874 #define CO_E_DECODEFAILED 0x8004021AL
1875 #define CO_E_ACNOTINITIALIZED 0x8004021BL
1876
1877 #define E_ACCESSDENIED 0x80070005L
1878 #define E_HANDLE 0x80070006L
1879 #define E_OUTOFMEMORY 0x8007000EL
1880 #define E_INVALIDARG 0x80070057L
1881
1882 /* For IksPropertySets */
```

```

1883 #define E_PROP_ID_UNSUPPORTED 0x80070490L
1884 #define E_PROP_SET_UNSUPPORTED 0x80070492L
1885
1886 #define CO_S_NOTALLINTERFACES 0x00080012L
1887
1888 #define CO_E_CLASS_CREATE_FAILED 0x80080001L
1889 #define CO_E_SCM_ERROR 0x80080002L
1890 #define CO_E_SCM_RPC_FAILURE 0x80080003L
1891 #define CO_E_BAD_PATH 0x80080004L
1892 #define CO_E_SERVER_EXEC_FAILURE 0x80080005L
1893 #define CO_E_OBJSRV_RPC_FAILURE 0x80080006L
1894 #define MK_E_NO_NORMALIZED 0x80080007L
1895 #define CO_E_SERVER_STOPPING 0x80080008L
1896 #define MEM_E_INVALID_ROOT 0x80080009L
1897 #define MEM_E_INVALID_LINK 0x80080010L
1898 #define MEM_E_INVALID_SIZE 0x80080011L
1899
1900
1901 #endif /* __WINE_WINERROR_H */

```

5.11 wingdi.h

```

1 #ifndef _WINGDI_
2 #define _WINGDI_
3 #ifndef NOGDI
4
5 #ifdef __cplusplus
6 extern "C" {
7 #endif
8
9 typedef struct _ABCFLOAT {
10     FLOAT    abcfA;
11     FLOAT    abcfB;
12     FLOAT    abcfC;
13 } ABCFLOAT, *PABCFLOAT, *LPABCFLOAT;
14
15 #define FONTMAPPER_MAX 10
16
17 typedef struct
18 {
19     WORD    wFirst;
20     WORD    wSecond;
21     INT     iKernAmount;
22 } KERNINGPAIR, *LPKERNINGPAIR;
23
24 typedef struct tagPIXELFORMATDESCRIPTOR {
25     WORD    nSize;
26     WORD    nVersion;
27     DWORD   dwFlags;
28     BYTE    iPixelFormat;
29     BYTE    cColorBits;
30     BYTE    cRedBits;
31     BYTE    cRedShift;
32     BYTE    cGreenBits;
33     BYTE    cGreenShift;
34     BYTE    cBlueBits;
35     BYTE    cBlueShift;
36     BYTE    cAlphaBits;
37     BYTE    cAlphaShift;
38     BYTE    cAccumBits;
39     BYTE    cAccumRedBits;
40     BYTE    cAccumGreenBits;
41     BYTE    cAccumBlueBits;
42     BYTE    cAccumAlphaBits;
43     BYTE    cDepthBits;
44     BYTE    cStencilBits;
45     BYTE    cAuxBuffers;
46     BYTE    iLayerType;
47     BYTE    bReserved;
48     DWORD   dwLayerMask;
49     DWORD   dwVisibleMask;
50     DWORD   dwDamageMask;
51 } PIXELFORMATDESCRIPTOR, *PPIXELFORMATDESCRIPTOR, *LPPIXELFORMATDESCRIPTOR;
52
53 #define PFD_TYPE_RGBA 0
54 #define PFD_TYPE_COLORINDEX 1
55
56 #define PFD_MAIN_PLANE 0
57 #define PFD_OVERLAY_PLANE 1
58 #define PFD_UNDERLAY_PLANE (-1)
59
60 #define PFD_DOUBLEBUFFER 0x00000001
61 #define PFD_STEREO 0x00000002
62 #define PFD_DRAW_TO_WINDOW 0x00000004

```



```

63 #define PFD_DRAW_TO_BITMAP          0x00000008
64 #define PFD_SUPPORT_GDI              0x00000010
65 #define PFD_SUPPORT_OPENGL           0x00000020
66 #define PFD_GENERIC_FORMAT           0x00000040
67 #define PFD_NEED_PALETTE             0x00000080
68 #define PFD_NEED_SYSTEM_PALETTE     0x00000100
69 #define PFD_SWAP_EXCHANGE             0x00000200
70 #define PFD_SWAP_COPY                0x00000400
71 #define PFD_SWAP_LAYER_BUFFERS       0x00000800
72 #define PFD_GENERIC_ACCELERATED      0x00001000
73
74 #define PFD_DEPTH_DONTCARE           0x20000000
75 #define PFD_DOUBLEBUFFER_DONTCARE    0x40000000
76 #define PFD_STEREO_DONTCARE          0x80000000
77
78 typedef struct tagCOLORADJUSTMENT
79 {
80     WORD    caSize;
81     WORD    caFlags;
82     WORD    caIlluminantIndex;
83     WORD    caRedGamma;
84     WORD    caGreenGamma;
85     WORD    caBlueGamma;
86     WORD    caReferenceBlack;
87     WORD    caReferenceWhite;
88     SHORT   caContrast;
89     SHORT   caBrightness;
90     SHORT   caColorfulness;
91     SHORT   caRedGreenTint;
92 } COLORADJUSTMENT, *PCOLORADJUSTMENT, *LPCOLORADJUSTMENT;
93
94 #define CA_NEGATIVE                    0x0001
95 #define CA_LOG_FILTER                  0x0002
96
97 #define ILLUMINANT_DEVICE_DEFAULT      0
98 #define ILLUMINANT_A                   1
99 #define ILLUMINANT_B                   2
100 #define ILLUMINANT_C                   3
101 #define ILLUMINANT_D50                 4
102 #define ILLUMINANT_D55                 5
103 #define ILLUMINANT_D65                 6
104 #define ILLUMINANT_D75                 7
105 #define ILLUMINANT_F2                  8
106 #define ILLUMINANT_MAX_INDEX           ILLUMINANT_F2
107
108 #define ILLUMINANT_TUNGSTEN             ILLUMINANT_A
109 #define ILLUMINANT_DAYLIGHT             ILLUMINANT_C
110 #define ILLUMINANT_FLUORESCENT          ILLUMINANT_F2
111 #define ILLUMINANT_NTSC                 ILLUMINANT_C
112
113 #define RGB_GAMMA_MIN                   (WORD) 02500
114 #define RGB_GAMMA_MAX                   (WORD) 65000
115
116 #define REFERENCE_WHITE_MIN             (WORD) 6000
117 #define REFERENCE_WHITE_MAX             (WORD) 10000
118 #define REFERENCE_BLACK_MIN             (WORD) 0
119 #define REFERENCE_BLACK_MAX             (WORD) 4000
120
121 #define COLOR_ADJ_MIN                   ((SHORT) -100)
122 #define COLOR_ADJ_MAX                   (SHORT) 100
123
124 typedef LONG FXPT16DOT16, *LPFXPT16DOT16;
125 typedef LONG FXPT2DOT30, *LPFXPT2DOT30;
126 typedef LONG LCSCSTYPE;
127 typedef LONG LCSGAMUTMATCH;
128
129 #define LCS_CALIBRATED_RGB              0x00000000L
130 #define LCS_DEVICE_RGB                  0x00000001L
131 #define LCS_DEVICE_CMYK                 0x00000002L
132
133 #define LCS_GM_BUSINESS                  0x00000001L
134 #define LCS_GM_GRAPHICS                  0x00000002L
135 #define LCS_GM_IMAGES                    0x00000004L
136
137 #define CM_OUT_OF_GAMUT                  255
138 #define CM_IN_GAMUT                     0
139
140 typedef struct tagCIEXYZ
141 {
142     FXPT2DOT30 ciexyzX;
143     FXPT2DOT30 ciexyzY;
144     FXPT2DOT30 ciexyzZ;
145 } CIEXYZ, *LPCIEXYZ;
146
147 typedef struct tagCIEXYZTRIPLE
148 {
149     CIEXYZ ciexyzRed;

```

```

150  CIEXYZ ciexyzGreen;
151  CIEXYZ ciexyzBlue;
152 } CIEXYZTRIPLE, *LPCIEXYZTRIPLE;
153
154 typedef struct tagLOGCOLORSPACEA
155 {
156     DWORD lcsSignature;
157     DWORD lcsVersion;
158     DWORD lcsSize;
159     LCSCSTYPE lcsCSType;
160     LCSGAMUTMATCH lcsIntent;
161     CIEXYZTRIPLE lcsEndpoints;
162     DWORD lcsGammaRed;
163     DWORD lcsGammaGreen;
164     DWORD lcsGammaBlue;
165     CHAR lcsFilename[MAX_PATH];
166 } LOGCOLORSPACEA, *LPLOGCOLORSPACEA;
167
168 typedef struct tagLOGCOLORSPACEW
169 {
170     DWORD lcsSignature;
171     DWORD lcsVersion;
172     DWORD lcsSize;
173     LCSCSTYPE lcsCSType;
174     LCSGAMUTMATCH lcsIntent;
175     CIEXYZTRIPLE lcsEndpoints;
176     DWORD lcsGammaRed;
177     DWORD lcsGammaGreen;
178     DWORD lcsGammaBlue;
179     WCHAR lcsFilename[MAX_PATH];
180 } LOGCOLORSPACEW, *LPLOGCOLORSPACEW;
181
182 DECL_WINELIB_TYPE_AW(LPLOGCOLORSPACE)
183 DECL_WINELIB_TYPE_AW(LOGCOLORSPACE)
184
185 #define DC_FIELDS          1
186 #define DC_PAPERS         2
187 #define DC_PAPERSIZE      3
188 #define DC_MINEXTENT      4
189 #define DC_MAXEXTENT      5
190 #define DC_BINS           6
191 #define DC_DUPLEX         7
192 #define DC_SIZE           8
193 #define DC_EXTRA          9
194 #define DC_VERSION        10
195 #define DC_DRIVER         11
196 #define DC_BINNAMES       12
197 #define DC_ENUMRESOLUTIONS 13
198 #define DC_FILEDEPENDENCIES 14
199 #define DC_TRUETYPE        15
200 #define DC_PAPERNAME       16
201 #define DC_ORIENTATION     17
202 #define DC_COPIES          18
203 #define DC_BINADJUST       19
204 #define DC_EMF_COMPLIANT   20
205 #define DC_DATATYPE_PRODUCED 21
206 #define DC_COLLATE         22
207 #define DC_MANUFACTURER    23
208 #define DC_MODEL           24
209 #define DC_PERSONALITY     25
210 #define DC_PRINTRATE       26
211 #define DC_PRINTRATEUNIT   27
212 #define DC_PRINTERMEM      28
213 #define DC_MEDIAREADY      29
214 #define DC_STAPLE          30
215 #define DC_PRINTRATEPPM    31
216 #define DC_COLORDEVICE     32
217 #define DC_NUP             33
218
219 #define DCTT_BITMAP        0x00000001L
220 #define DCTT_DOWNLOAD      0x00000002L
221 #define DCTT_SUBDEV        0x00000004L
222 #define DCTT_DOWNLOAD_OUTLINE 0x00000008L
223
224 #define DCBA_FACEUPNONE    0x0000
225 #define DCBA_FACEUPCENTER  0x0001
226 #define DCBA_FACEUPLEFT    0x0002
227 #define DCBA_FACEUPRIGHT   0x0003
228 #define DCBA_FACEDOWNNONE  0x0100
229 #define DCBA_FACEDOWNCENTER 0x0101
230 #define DCBA_FACEDOWNLEFT  0x0102
231 #define DCBA_FACEDOWNRIGHT 0x0103
232
233 #define PRINTRATEUNIT_PPM   1
234 #define PRINTRATEUNIT_CPS   2
235 #define PRINTRATEUNIT_LPM   3
236 #define PRINTRATEUNIT_IPM   4

```

```
237
238 /* Flag returned from Escape QUERYDIBSUPPORT */
239 #define QDI_SETDIBITS 1
240 #define QDI_GETDIBITS 2
241 #define QDI_DIBTOSCREEN 4
242 #define QDI_STRETCHDIB 8
243
244
245 /* GDI Escape commands */
246 #define NEWFRAME 1
247 #define ABORTDOC 2
248 #define NEXTBAND 3
249 #define SETCOLORTABLE 4
250 #define GETCOLORTABLE 5
251 #define FLUSHOUTPUT 6
252 #define DRAFTMODE 7
253 #define QUERYESCSUPPORT 8
254 #define SETABORTPROC 9
255 #define STARTDOC 10
256 #define ENDDOC 11
257 #define GETPHYSPAGE_SIZE 12
258 #define GETPRINTINGOFFSET 13
259 #define GETSCALINGFACTOR 14
260 #define MFCOMMENT 15
261 #define GETPENWIDTH 16
262 #define SETCOPYCOUNT 17
263 #define SELECTPAPERSOURCE 18
264 #define DEVICEDATA 19
265 #define PASSTHROUGH 19
266 #define GETTECHNOLOGY 20
267 #define GETTECHNOLOGY 20 /* yes, both of them */
268 #define SETLINECAP 21
269 #define SETLINEJOIN 22
270 #define SETMITERLIMIT 23
271 #define BANDINFO 24
272 #define DRAWPATTERNRECT 25
273 #define GETVECTORPEN_SIZE 26
274 #define GETVECTORBRUSH_SIZE 27
275 #define ENABLEDUPLEX 28
276 #define GETSETPAPERBINS 29
277 #define GETSETPRINTORIENT 30
278 #define ENUMPAPERBINS 31
279 #define SETDIBSCALING 32
280 #define EPSPRINTING 33
281 #define ENUMPAPERMETRICS 34
282 #define GETSETPAPERMETRICS 35
283 #define POSTSCRIPT_DATA 37
284 #define POSTSCRIPT_IGNORE 38
285 #define MOUSETRAILS 39
286 #define GETDEVICEUNITS 42
287
288 #define DESKTOPVERTRES 117
289 #define DESKTOPHORZRES 118
290
291 #define GETEXTENDEDTEXTMETRICS 256
292 #define GETTEXTENTTABLE 257
293 #define GETPAIRKERNTABLE 258
294 #define GETTRACKKERNTABLE 259
295 #define EXTTEXTOUT 512
296 #define GETFACENAME 513
297 #define DOWNLOADFACE 514
298 #define ENABLERELATIVEWIDTHS 768
299 #define ENABLEPAIRKERNING 769
300 #define SETKERNTACK 770
301 #define SETALLJUSTVALUES 771
302 #define SETCHARSET 772
303
304 #define STRETCHBLT 2048
305 #define GETSETSCREENPARAMS 3072
306 #define QUERYDIBSUPPORT 3073
307 #define BEGIN_PATH 4096
308 #define CLIP_TO_PATH 4097
309 #define END_PATH 4098
310 #define EXT_DEVICE_CAPS 4099
311 #define RESTORE_CTM 4100
312 #define SAVE_CTM 4101
313 #define SET_ARC_DIRECTION 4102
314 #define SET_BACKGROUND_COLOR 4103
315 #define SET_POLY_MODE 4104
316 #define SET_SCREEN_ANGLE 4105
317 #define SET_SPREAD 4106
318 #define TRANSFORM_CTM 4107
319 #define SET_CLIP_BOX 4108
320 #define SET_BOUNDS 4109
321 #define SET_MIRROR_MODE 4110
322 #define OPENCHANNEL 4110
323 #define DOWNLOADHEADER 4111
```

```

324 #define CLOSECHANNEL          4112
325 #define POSTSCRIPT_PASSTHROUGH 4115
326 #define ENCAPSULATED_POSTSCRIPT 4116
327 #define POSTSCRIPT_IDENTIFY 4117
328 #define POSTSCRIPT_INJECTION    4118
329
330 /* for POSTSCRIPT_IDENTIFY */
331 #define PSIDENT_GDICENTRIC 0
332 #define PSIDENT_PSCENTRIC 1
333
334
335 #define QDI_SETDIBITS          1
336 #define QDI_GETDIBITS          2
337 #define QDI_DIBTOSCREEN        4
338 #define QDI_STRETCHDIB        8
339
340 /* Spooler Error Codes */
341 #define SP_NOTREPORTED 0x4000
342 #define SP_ERROR      (-1)
343 #define SP_APPABORT   (-2)
344 #define SP_USERABORT  (-3)
345 #define SP_OUTOFDISK  (-4)
346 #define SP_OUTOFMEMORY (-5)
347
348 #define PR_JOBSTATUS    0
349
350 /* Raster operations */
351
352 #define R2_BLACK          1
353 #define R2_NOTMERGEPEN    2
354 #define R2_MASKNOTPEN     3
355 #define R2_NOTCOPYPEN    4
356 #define R2_MASKPENNNOT   5
357 #define R2_NOT            6
358 #define R2_XORPEN         7
359 #define R2_NOTMASKPEN     8
360 #define R2_MASKPEN        9
361 #define R2_NOTXORPEN      10
362 #define R2_NOP            11
363 #define R2_MERGENOTPEN    12
364 #define R2_COPYPEN        13
365 #define R2_MERGEPENNOT    14
366 #define R2_MERGEPEPEN     15
367 #define R2_WHITE          16
368
369 #define SRCCOPY            0xcc0020
370 #define SRCPAINT           0xee0086
371 #define SRCAND             0x8800c6
372 #define SRCINVERT          0x660046
373 #define SRCERASE           0x440328
374 #define NOTSRCCOPY         0x330008
375 #define NOTSRCERASE        0x1100a6
376 #define MERGECOPY          0xc000ca
377 #define MERGEPAINT         0xbb0226
378 #define PATCOPY            0xf00021
379 #define PATPAINT           0xfb0a09
380 #define PATINVERT          0x5a0049
381 #define DSTINVERT          0x550009
382 #define BLACKNESS          0x000042
383 #define WHITENESS          0xff0062
384
385 /* StretchBlt() modes */
386 #define BLACKONWHITE        1
387 #define WHITEONBLACK        2
388 #define COLORONCOLOR        3
389 #define HALFTONE            4
390 #define MAXSTRETCHBLTMODE   4
391
392 #define STRETCH_ANDSCANS     BLACKONWHITE
393 #define STRETCH_ORSCANS      WHITEONBLACK
394 #define STRETCH_DELETESCANS  COLORONCOLOR
395 #define STRETCH_HALFTONE     HALFTONE
396
397 /* Colors */
398
399 #define RGB(r,g,b)           ((COLORREF)((r) | ((g) << 8) | ((b) << 16)))
400 #define PALETTE_RGB(r,g,b)   (0x02000000 | RGB(r,g,b))
401 #define PALETTE_INDEX(i)     ((COLORREF)(0x01000000 | (WORD)(i)))
402
403 #define GetRValue(rgb)        ((rgb) & 0xff)
404 #define GetGValue(rgb)        (((rgb) >> 8) & 0xff)
405 #define GetBValue(rgb)        (((rgb) >> 16) & 0xff)
406
407 #define GetKValue(cmyk)       ((BYTE) (cmyk) )
408 #define GetYValue(cmyk)       ((BYTE) ((cymk) >> 8))
409 #define GetMValue(cmyk)       ((BYTE) ((cymk) >> 16))
410 #define GetCValue(cmyk)       ((BYTE) ((cymk) >> 24))

```

```

411
412 #define CMYK(c,m,y,k)
413     ((COLORREF) ((( (BYTE) (k) | ((WORD) ((BYTE) (y) << 8)) | (((DWORD) (BYTE) (m) << 16)) | (((DWORD) (BYTE) (c) << 24)) ))
414
415 #define ICM_OFF 1
416 #define ICM_ON 2
417 #define ICM_QUERY 3
418
419 /* Bounds Accumulation APIs */
420 #define DCB_RESET 0x0001
421 #define DCB_ACCUMULATE 0x0002
422 #define DCB_DIRTY DCB_ACCUMULATE
423 #define DCB_SET (DCB_RESET | DCB_ACCUMULATE)
424 #define DCB_ENABLE 0x0004
425 #define DCB_DISABLE 0x0008
426
427 typedef struct
428 {
429     LONG paXCount;
430     LONG paYCount;
431     LONG paXExt;
432     LONG paYExt;
433     BYTE paRGBs;
434 } PELARRAY, *PELARRAY, *LPPELARRAY;
435
436 /* Bitmaps */
437
438 typedef struct
439 {
440     INT bmType;
441     INT bmWidth;
442     INT bmHeight;
443     INT bmWidthBytes;
444     WORD bmPlanes;
445     WORD bmBitsPixel;
446     LPVOID bmBits;
447 } BITMAP, *PBITMAP, *LPBITMAP;
448
449
450 /* Brushes */
451
452 typedef struct
453 {
454     UINT lbStyle;
455     COLORREF lbColor;
456     INT lbHatch;
457 } LOGBRUSH, *PLOGBRUSH, *LPLOGBRUSH;
458
459 typedef LOGBRUSH PATTERN, *PPATTERN, *LPPATTERN;
460
461
462 /* Brush styles */
463 #define BS_SOLID 0
464 #define BS_NULL 1
465 #define BS_HOLLOW 1
466 #define BS_HATCHED 2
467 #define BS_PATTERN 3
468 #define BS_INDEXED 4
469 #define BS_DIBPATTERN 5
470 #define BS_DIBPATTERNPT 6
471 #define BS_PATTERN8X8 7
472 #define BS_DIBPATTERN8X8 8
473 #define BS_MONOPATTERN 9
474
475 /* Hatch styles */
476 #define HS_HORIZONTAL 0
477 #define HS_VERTICAL 1
478 #define HS_FDIAGONAL 2
479 #define HS_BDIAGONAL 3
480 #define HS_CROSS 4
481 #define HS_DIAGCROSS 5
482
483 /* Fonts */
484
485 #define LF_FACESIZE 32
486 #define LF_FULLFACESIZE 64
487
488 #define RASTER_FONTTYPE 0x0001
489 #define DEVICE_FONTTYPE 0x0002
490 #define TRUETYPE_FONTTYPE 0x0004
491
492 typedef struct
493 {
494     LONG lfHeight;
495     LONG lfWidth;
496     LONG lfEscapement;

```

```

497     LONG     lfOrientation;
498     LONG     lfWeight;
499     BYTE     lfItalic;
500     BYTE     lfUnderline;
501     BYTE     lfStrikeOut;
502     BYTE     lfCharSet;
503     BYTE     lfOutPrecision;
504     BYTE     lfClipPrecision;
505     BYTE     lfQuality;
506     BYTE     lfPitchAndFamily;
507     CHAR     lfFaceName[LF_FACESIZE];
508 } LOGFONTA, *PLOGFONTA, *LPLOGFONTA;
509
510 typedef struct
511 {
512     LONG     lfHeight;
513     LONG     lfWidth;
514     LONG     lfEscapement;
515     LONG     lfOrientation;
516     LONG     lfWeight;
517     BYTE     lfItalic;
518     BYTE     lfUnderline;
519     BYTE     lfStrikeOut;
520     BYTE     lfCharSet;
521     BYTE     lfOutPrecision;
522     BYTE     lfClipPrecision;
523     BYTE     lfQuality;
524     BYTE     lfPitchAndFamily;
525     WCHAR    lfFaceName[LF_FACESIZE];
526 } LOGFONTW, *PLOGFONTW, *LPLOGFONTW;
527
528 DECL_WINELIB_TYPE_AW(LOGFONT)
529 DECL_WINELIB_TYPE_AW(PLOGFONT)
530 DECL_WINELIB_TYPE_AW(LPLOGFONT)
531
532 typedef struct
533 {
534     LOGFONTA  elfLogFont;
535     BYTE      elfFullName[LF_FULLFACESIZE];
536     BYTE      elfStyle[LF_FACESIZE];
537 } ENUMLOGFONTA, *LPENUMLOGFONTA;
538
539 typedef struct
540 {
541     LOGFONTW  elfLogFont;
542     WCHAR     elfFullName[LF_FULLFACESIZE];
543     WCHAR     elfStyle[LF_FACESIZE];
544 } ENUMLOGFONTW, *LPENUMLOGFONTW;
545
546 DECL_WINELIB_TYPE_AW(ENUMLOGFONT)
547 DECL_WINELIB_TYPE_AW(LPENUMLOGFONT)
548
549 typedef struct
550 {
551     LOGFONTA  elfLogFont;
552     BYTE      elfFullName[LF_FULLFACESIZE];
553     BYTE      elfStyle[LF_FACESIZE];
554     BYTE      elfScript[LF_FACESIZE];
555 } ENUMLOGFONTEXA, *LPENUMLOGFONTEXA;
556
557 typedef struct
558 {
559     LOGFONTW  elfLogFont;
560     WCHAR     elfFullName[LF_FULLFACESIZE];
561     WCHAR     elfStyle[LF_FACESIZE];
562     WCHAR     elfScript[LF_FACESIZE];
563 } ENUMLOGFONTEXW, *LPENUMLOGFONTEXW;
564
565 DECL_WINELIB_TYPE_AW(ENUMLOGFONTEX)
566 DECL_WINELIB_TYPE_AW(LPENUMLOGFONTEX)
567
568 /*
569 * The FONTSIGNATURE tells which Unicode ranges and which code pages
570 * have glyphs in a font.
571 *
572 * fsUsb  128-bit bitmap. The most significant bits are 10 (magic number).
573 *        The remaining 126 bits map the Unicode ISO 10646 subranges
574 *        for which the font provides glyphs.
575 *
576 * fsCsb  64-bit bitmap. The low 32 bits map the Windows codepages for
577 *        which the font provides glyphs. The high 32 bits are for
578 *        non Windows codepages.
579 */
580 typedef struct
581 {
582     DWORD fsUsb[4];
583     DWORD fsCsb[2];

```

```

584 } FONTSIGNATURE, *PFONTSIGNATURE, *LPFONTSIGNATURE;
585
586 typedef struct
587 {
588     UINT ciCharset; /* character set */
589     UINT ciACP; /* ANSI code page */
590     FONTSIGNATURE fs;
591 } CHARSETINFO, *PCHARSETINFO, *LPCHARSETINFO;
592
593 /* Flags for TranslateCharsetInfo */
594 #define TCI_SRCCHARSET 1
595 #define TCI_SRCCODEPAGE 2
596 #define TCI_SRCFONTSIG 3
597
598 typedef struct
599 {
600     DWORD lsUsb[4];
601     DWORD lsCsbDefault[2];
602     DWORD lsCsbSupported[2];
603 } LOCALESIGNATURE, *PLOCALESIGNATUR, *LPLOCALESIGNATURE;
604
605
606 /* Flags for ModifyWorldTransform */
607 #define MWT_IDENTITY 1
608 #define MWT_LEFTMULTIPLY 2
609 #define MWT_RIGHTMULTIPLY 3
610 #define MWT_MIN MWT_IDENTITY
611 #define MWT_MAX MWT_RIGHTMULTIPLY
612
613 /* Object Definitions for EnumObjects() */
614 #define OBJ_PEN 1
615 #define OBJ_BRUSH 2
616 #define OBJ_DC 3
617 #define OBJ_METADC 4
618 #define OBJ_PAL 5
619 #define OBJ_FONT 6
620 #define OBJ_BITMAP 7
621 #define OBJ_REGION 8
622 #define OBJ_METAFILE 9
623 #define OBJ_MEMDC 10
624 #define OBJ_EXTPEN 11
625 #define OBJ_ENHMETADC 12
626 #define OBJ_ENHMETAFILE 13
627
628 typedef struct
629 {
630     FLOAT eM11;
631     FLOAT eM12;
632     FLOAT eM21;
633     FLOAT eM22;
634     FLOAT eDx;
635     FLOAT eDy;
636 } XFORM, *PXFORM, *LPXFORM;
637
638 /* lfWeight values */
639 #define FW_DONTCARE 0
640 #define FW_THIN 100
641 #define FW_EXTRALIGHT 200
642 #define FW_ULTRALIGHT 200
643 #define FW_LIGHT 300
644 #define FW_NORMAL 400
645 #define FW_REGULAR 400
646 #define FW_MEDIUM 500
647 #define FW_SEMIBOLD 600
648 #define FW_DEMIBOLD 600
649 #define FW_BOLD 700
650 #define FW_EXTRABOLD 800
651 #define FW_ULTRABOLD 800
652 #define FW_HEAVY 900
653 #define FW_BLACK 900
654
655 /* lfCharSet values */
656 #define ANSI_CHARSET (BYTE)0 /* CP1252, ansi-0, iso8859-{1,15} */
657 #define DEFAULT_CHARSET (BYTE)1
658 #define SYMBOL_CHARSET (BYTE)2
659 #define SHIFTJIS_CHARSET (BYTE)128 /* CP932 */
660 #define HANGEUL_CHARSET (BYTE)129 /* CP949, ksc5601.1987-0 */
661 #define HANGUL_CHARSET HANGEUL_CHARSET
662 #define GB2312_CHARSET (BYTE)134 /* CP936, gb2312.1980-0 */
663 #define CHINESEBIG5_CHARSET (BYTE)136 /* CP950, big5.et-0 */
664 #define GREEK_CHARSET (BYTE)161 /* CP1253 */
665 #define TURKISH_CHARSET (BYTE)162 /* CP1254, -iso8859-9 */
666 #define HEBREW_CHARSET (BYTE)177 /* CP1255, -iso8859-8 */
667 #define ARABIC_CHARSET (BYTE)178 /* CP1256, -iso8859-6 */
668 #define BALTIC_CHARSET (BYTE)186 /* CP1257, -iso8859-13 */
669 #define RUSSIAN_CHARSET (BYTE)204 /* CP1251, -iso8859-5 */
670 #define EE_CHARSET (BYTE)238 /* CP1250, -iso8859-2 */

```

```

671 #define EASTEUROPE_CHARSET    EE_CHARSET
672 #define THAI_CHARSET           (BYTE)222 /* CP874, iso8859-11, tis620 */
673 #define JOHAB_CHARSET          (BYTE)130 /* korean (johab) CP1361 */
674 #define MAC_CHARSET            (BYTE)77
675 #define OEM_CHARSET            (BYTE)255
676 /* I don't know if the values of *_CHARSET macros are defined in Windows
677 * or if we can choose them as we want. -- srtxg
678 */
679 #define VISCI1_CHARSET         (BYTE)240 /* viscii1.1-1 */
680 #define TCVN_CHARSET           (BYTE)241 /* tcvn-0 */
681 #define KOI8_CHARSET           (BYTE)242 /* koi8-{r,u,ru} */
682 #define ISO3_CHARSET           (BYTE)243 /* iso8859-3 */
683 #define ISO4_CHARSET           (BYTE)244 /* iso8859-4 */
684 #define ISO10_CHARSET          (BYTE)245 /* iso8859-10 */
685 #define CELTIC_CHARSET         (BYTE)246 /* iso8859-14 */
686
687 #define FS_LATIN1               0x00000001L
688 #define FS_LATIN2               0x00000002L
689 #define FS_CYRILLIC            0x00000004L
690 #define FS_GREEK               0x00000008L
691 #define FS_TURKISH             0x00000010L
692 #define FS_HEBREW              0x00000020L
693 #define FS_ARABIC              0x00000040L
694 #define FS_BALTIC              0x00000080L
695 #define FS_VIETNAMESE          0x00000100L
696 #define FS_THAI                0x00010000L
697 #define FS_JISJAPAN            0x00020000L
698 #define FS_CHINESESIMP         0x00040000L
699 #define FS_WANSUNG             0x00080000L
700 #define FS_CHINESETRAD         0x00100000L
701 #define FS_JOHAB               0x00200000L
702 #define FS_SYMBOL              0x80000000L
703
704 /* lfOutPrecision values */
705 #define OUT_DEFAULT_PRECIS     0
706 #define OUT_STRING_PRECIS      1
707 #define OUT_CHARACTER_PRECIS   2
708 #define OUT_STROKE_PRECIS      3
709 #define OUT_TT_PRECIS          4
710 #define OUT_DEVICE_PRECIS      5
711 #define OUT_RASTER_PRECIS      6
712 #define OUT_TT_ONLY_PRECIS     7
713 #define OUT_OUTLINE_PRECIS     8
714
715 /* lfClipPrecision values */
716 #define CLIP_DEFAULT_PRECIS    0x00
717 #define CLIP_CHARACTER_PRECIS  0x01
718 #define CLIP_STROKE_PRECIS     0x02
719 #define CLIP_MASK              0x0F
720 #define CLIP_LH_ANGLES         0x10
721 #define CLIP_TT_ALWAYS         0x20
722 #define CLIP_EMBEDDED          0x80
723
724 /* lfQuality values */
725 #define DEFAULT_QUALITY        0
726 #define DRAFT_QUALITY          1
727 #define PROOF_QUALITY          2
728 #define NONANTIALIASED_QUALITY 3
729 #define ANTIALIASED_QUALITY    4
730
731 /* lfPitchAndFamily pitch values */
732 #define DEFAULT_PITCH          0x00
733 #define FIXED_PITCH            0x01
734 #define VARIABLE_PITCH        0x02
735 #define MONO_FONT              0x08
736
737 #define FF_DONTCARE             0x00
738 #define FF_ROMAN                0x10
739 #define FF_SWISS                0x20
740 #define FF_MODERN              0x30
741 #define FF_SCRIPT              0x40
742 #define FF_DECORATIVE          0x50
743
744 typedef struct
745 {
746     LONG        tmHeight;
747     LONG        tmAscent;
748     LONG        tmDescent;
749     LONG        tmInternalLeading;
750     LONG        tmExternalLeading;
751     LONG        tmAveCharWidth;
752     LONG        tmMaxCharWidth;
753     LONG        tmWeight;
754     LONG        tmOverhang;
755     LONG        tmDigitizedAspectX;
756     LONG        tmDigitizedAspectY;
757     BYTE        tmFirstChar;

```



```

758     BYTE        tmLastChar;
759     BYTE        tmDefaultChar;
760     BYTE        tmBreakChar;
761     BYTE        tmItalic;
762     BYTE        tmUnderlined;
763     BYTE        tmStruckOut;
764     BYTE        tmPitchAndFamily;
765     BYTE        tmCharSet;
766 } TEXTMETRICA, *LPTEXTMETRICA, *PTEXTMETRICA;
767
768 typedef struct
769 {
770     LONG        tmHeight;
771     LONG        tmAscent;
772     LONG        tmDescent;
773     LONG        tmInternalLeading;
774     LONG        tmExternalLeading;
775     LONG        tmAveCharWidth;
776     LONG        tmMaxCharWidth;
777     LONG        tmWeight;
778     LONG        tmOverhang;
779     LONG        tmDigitizedAspectX;
780     LONG        tmDigitizedAspectY;
781     WCHAR       tmFirstChar;
782     WCHAR       tmLastChar;
783     WCHAR       tmDefaultChar;
784     WCHAR       tmBreakChar;
785     BYTE        tmItalic;
786     BYTE        tmUnderlined;
787     BYTE        tmStruckOut;
788     BYTE        tmPitchAndFamily;
789     BYTE        tmCharSet;
790 } TEXTMETRICW, *LPTEXTMETRICW, *PTEXTMETRICW;
791
792 DECL_WINELIB_TYPE_AW(TEXTMETRIC)
793 DECL_WINELIB_TYPE_AW(PTEXTMETRIC)
794 DECL_WINELIB_TYPE_AW(LPTEXTMETRIC)
795
796
797 typedef struct tagPANOSE
798 {
799     BYTE bFamilyType;
800     BYTE bSerifStyle;
801     BYTE bWeight;
802     BYTE bProportion;
803     BYTE bContrast;
804     BYTE bStrokeVariation;
805     BYTE bArmStyle;
806     BYTE bLetterform;
807     BYTE bMidline;
808     BYTE bXHeight;
809 } PANOSE, *LPPANOSE;
810
811 #define PANOSE_COUNT                10
812
813 #define PANOSE_FAMILYTYPE_INDEX     0
814 #define PAN_SERIFSTYLE_INDEX        1
815 #define PAN_WEIGHT_INDEX            2
816 #define PAN_PROPORTION_INDEX        3
817 #define PAN_CONTRAST_INDEX          4
818 #define PAN_STROKEVARIATION_INDEX   5
819 #define PAN_ARMSTYLE_INDEX          6
820 #define PAN_LETTERFORM_INDEX        7
821 #define PAN_MIDLINE_INDEX           8
822 #define PAN_XHEIGHT_INDEX           9
823
824 #define PAN_CULTURE_LATIN           0
825
826 #define PAN_ANY                      0
827 #define PAN_NO_FIT                   1
828
829 #define PAN_FAMILY_TEXT_DISPLAY     2
830 #define PAN_FAMILY_SCRIPT           3
831 #define PAN_FAMILY_DECORATIVE       4
832 #define PAN_FAMILY_PICTORIAL        5
833
834 #define PAN_SERIF_COVE              2
835 #define PAN_SERIF_OBTUSE_COVE       3
836 #define PAN_SERIF_SQUARE_COVE       4
837 #define PAN_SERIF_OBTUSE_SQUARE_COVE 5
838 #define PAN_SERIF_SQUARE            6
839 #define PAN_SERIF_THIN              7
840 #define PAN_SERIF_BONE              8
841 #define PAN_SERIF_EXAGGERATED       9
842 #define PAN_SERIF_TRIANGLE          10
843 #define PAN_SERIF_NORMAL_SANS       11
844 #define PAN_SERIF_OBTUSE_SANS       12

```

```
845 #define PAN_SERIF_PERP_SANS      13
846 #define PAN_SERIF_FLARED          14
847 #define PAN_SERIF_ROUNDED          15
848
849 #define PAN_WEIGHT_VERY_LIGHT      2
850 #define PAN_WEIGHT_LIGHT           3
851 #define PAN_WEIGHT_THIN            4
852 #define PAN_WEIGHT_BOOK            5
853 #define PAN_WEIGHT_MEDIUM          6
854 #define PAN_WEIGHT_DEMI            7
855 #define PAN_WEIGHT_BOLD            8
856 #define PAN_WEIGHT_HEAVY           9
857 #define PAN_WEIGHT_BLACK           10
858 #define PAN_WEIGHT_NORD            11
859
860 #define PAN_PROP_OLD_STYLE          2
861 #define PAN_PROP_MODERN             3
862 #define PAN_PROP_EVEN_WIDTH         4
863 #define PAN_PROP_EXPANDED           5
864 #define PAN_PROP_CONDENSED          6
865 #define PAN_PROP_VERY_EXPANDED      7
866 #define PAN_PROP_VERY_CONDENSED     8
867 #define PAN_PROP_MONOSPACED         9
868
869 #define PAN_CONTRAST_NONE            2
870 #define PAN_CONTRAST_VERY_LOW        3
871 #define PAN_CONTRAST_LOW             4
872 #define PAN_CONTRAST_MEDIUM_LOW      5
873 #define PAN_CONTRAST_MEDIUM          6
874 #define PAN_CONTRAST_MEDIUM_HIGH     7
875 #define PAN_CONTRAST_HIGH            8
876 #define PAN_CONTRAST_VERY_HIGH       9
877
878 #define PAN_STROKE_GRADUAL_DIAG      2
879 #define PAN_STROKE_GRADUAL_TRAN      3
880 #define PAN_STROKE_GRADUAL_VERT      4
881 #define PAN_STROKE_GRADUAL_HORZ      5
882 #define PAN_STROKE_RAPID_VERT        6
883 #define PAN_STROKE_RAPID_HORZ        7
884 #define PAN_STROKE_INSTANT_VERT      8
885
886 #define PAN_STRAIGHT_ARMS_HORZ       2
887 #define PAN_STRAIGHT_ARMS_WEDGE     3
888 #define PAN_STRAIGHT_ARMS_VERT       4
889 #define PAN_STRAIGHT_ARMS_SINGLE_SERIF 5
890 #define PAN_STRAIGHT_ARMS_DOUBLE_SERIF 6
891 #define PAN_BENT_ARMS_HORZ           7
892 #define PAN_BENT_ARMS_WEDGE          8
893 #define PAN_BENT_ARMS_VERT           9
894 #define PAN_BENT_ARMS_SINGLE_SERIF   10
895 #define PAN_BENT_ARMS_DOUBLE_SERIF   11
896
897 #define PAN_LETT_NORMAL_COMPACT       2
898 #define PAN_LETT_NORMAL_WEIGHTED      3
899 #define PAN_LETT_NORMAL_BOXED         4
900 #define PAN_LETT_NORMAL_FLATTENED     5
901 #define PAN_LETT_NORMAL_ROUNDED       6
902 #define PAN_LETT_NORMAL_OFF_CENTER    7
903 #define PAN_LETT_NORMAL_SQUARE        8
904 #define PAN_LETT_OBLIQUE_COMPACT      9
905 #define PAN_LETT_OBLIQUE_WEIGHTED    10
906 #define PAN_LETT_OBLIQUE_BOXED       11
907 #define PAN_LETT_OBLIQUE_FLATTENED   12
908 #define PAN_LETT_OBLIQUE_ROUNDED     13
909 #define PAN_LETT_OBLIQUE_OFF_CENTER   14
910 #define PAN_LETT_OBLIQUE_SQUARE       15
911
912 #define PAN_MIDLINE_STANDARD_TRIMMED  2
913 #define PAN_MIDLINE_STANDARD_POINTED  3
914 #define PAN_MIDLINE_STANDARD_SERIFED  4
915 #define PAN_MIDLINE_HIGH_TRIMMED      5
916 #define PAN_MIDLINE_HIGH_POINTED      6
917 #define PAN_MIDLINE_HIGH_SERIFED      7
918 #define PAN_MIDLINE_CONSTANT_TRIMMED  8
919 #define PAN_MIDLINE_CONSTANT_POINTED  9
920 #define PAN_MIDLINE_CONSTANT_SERIFED  10
921 #define PAN_MIDLINE_LOW_TRIMMED       11
922 #define PAN_MIDLINE_LOW_POINTED       12
923 #define PAN_MIDLINE_LOW_SERIFED      13
924
925 #define PAN_XHEIGHT_CONSTANT_SMALL    2
926 #define PAN_XHEIGHT_CONSTANT_STANDARD 3
927 #define PAN_XHEIGHT_CONSTANT_LARGE    4
928 #define PAN_XHEIGHT_DUCKING_SMALL     5
929 #define PAN_XHEIGHT_DUCKING_STANDARD  6
930 #define PAN_XHEIGHT_DUCKING_LARGE     7
931
```

```

932 #define ELF_VENDOR_SIZE 4
933 typedef struct
934 {
935     LOGFONTA    elfLogFont;
936     BYTE        elfFullName[LF_FULLFACESIZE];
937     BYTE        elfStyle[LF_FACESIZE];
938     DWORD       elfVersion;
939     DWORD       elfStyleSize;
940     DWORD       elfMatch;
941     DWORD       elfReserved;
942     BYTE        elfVendorId[ELF_VENDOR_SIZE];
943     DWORD       elfCulture;
944     PANOSE      elfPanose;
945 } EXTLOGFONTA, *PEXTLOGFONTA, *LPEXTLOGFONTA;
946
947 typedef struct
948 {
949     LOGFONTW    elfLogFont;
950     WCHAR       elfFullName[LF_FULLFACESIZE];
951     WCHAR       elfStyle[LF_FACESIZE];
952     DWORD       elfVersion;
953     DWORD       elfStyleSize;
954     DWORD       elfMatch;
955     DWORD       elfReserved;
956     BYTE        elfVendorId[ELF_VENDOR_SIZE];
957     DWORD       elfCulture;
958     PANOSE      elfPanose;
959 } EXTLOGFONTW, *PEXTLOGFONTW, *LPEXTLOGFONTW;
960
961 DECL_WINELIB_TYPE_AW(EXTLOGFONT)
962 DECL_WINELIB_TYPE_AW(PEXTLOGFONT)
963 DECL_WINELIB_TYPE_AW(LPEXTLOGFONT)
964
965 #define ELF_VERSION      0
966 #define ELF_CULTURE_LATIN 0
967
968 typedef struct _OUTLINETEXMETRICA
969 {
970     UINT        otmSize;
971     TEXTMETRICA otmTextMetrics;
972     BYTE        otmFiller;
973     PANOSE      otmPanoseNumber;
974     UINT        otmfsSelection;
975     UINT        otmfsType;
976     INT         otmsCharSlopeRise;
977     INT         otmsCharSlopeRun;
978     INT         otmItalicAngle;
979     UINT        otmEMSquare;
980     INT         otmAscent;
981     INT         otmDescent;
982     UINT        otmLineGap;
983     UINT        otmsCapEmHeight;
984     UINT        otmsXHeight;
985     RECT        otmrcFontBox;
986     INT         otmMacAscent;
987     INT         otmMacDescent;
988     UINT        otmMacLineGap;
989     UINT        otmusMinimumPEM;
990     POINT       otmptSubscriptSize;
991     POINT       otmptSubscriptOffset;
992     POINT       otmptSuperscriptSize;
993     POINT       otmptSuperscriptOffset;
994     UINT        otmsStrikeoutSize;
995     INT         otmsStrikeoutPosition;
996     INT         otmsUnderscoreSize;
997     INT         otmsUnderscorePosition;
998     LPSTR       otmpFamilyName;
999     LPSTR       otmpFaceName;
1000    LPSTR       otmpStyleName;
1001    LPSTR       otmpFullName;
1002 } OUTLINETEXMETRICA, *POUTLINETEXMETRICA, *LPOUTLINETEXMETRICA;
1003
1004 typedef struct _OUTLINETEXMETRICW
1005 {
1006     UINT        otmSize;
1007     TEXTMETRICW otmTextMetrics;
1008     BYTE        otmFiller;
1009     PANOSE      otmPanoseNumber;
1010     UINT        otmfsSelection;
1011     UINT        otmfsType;
1012     INT         otmsCharSlopeRise;
1013     INT         otmsCharSlopeRun;
1014     INT         otmItalicAngle;
1015     UINT        otmEMSquare;
1016     INT         otmAscent;
1017     INT         otmDescent;
1018     UINT        otmLineGap;

```

```

1019     UINT            otmsCapEmHeight;
1020     UINT            otmsXHeight;
1021     RECT            otmrcFontBox;
1022     INT             otmMacAscent;
1023     INT             otmMacDescent;
1024     UINT            otmMacLineGap;
1025     UINT            otmusMinimumPPEM;
1026     POINT           otmptSubscriptSize;
1027     POINT           otmptSubscriptOffset;
1028     POINT           otmptSuperscriptSize;
1029     POINT           otmptSuperscriptOffset;
1030     UINT            otmsStrikeoutSize;
1031     INT             otmsStrikeoutPosition;
1032     INT             otmsUnderscoreSize;
1033     INT             otmsUnderscorePosition;
1034     LPSTR           otmpFamilyName;
1035     LPSTR           otmpFaceName;
1036     LPSTR           otmpStyleName;
1037     LPSTR           otmpFullName;
1038 } OUTLINETEXTMETRICW, *POUTLINETEXTMETRICW, *LPOUTLINETEXTMETRICW;
1039
1040 DECL_WINELIB_TYPE_AW(OUTLINETEXTMETRIC)
1041 DECL_WINELIB_TYPE_AW(POUTLINETEXTMETRIC)
1042 DECL_WINELIB_TYPE_AW(LPOUTLINETEXTMETRIC)
1043
1044 typedef struct
1045 {
1046     INT            x;
1047     INT            y;
1048     UINT           n;
1049     LPCSTR         lpstr;
1050     UINT           uiFlags;
1051     RECT           rcl;
1052     INT            *pdx;
1053 } POLYTEXTA, *PPOLYTEXTA, *LPPOLYTEXTA;
1054
1055 typedef struct
1056 {
1057     INT            x;
1058     INT            y;
1059     UINT           n;
1060     LPCWSTR        lpstr;
1061     UINT           uiFlags;
1062     RECT           rcl;
1063     INT            *pdx;
1064 } POLYTEXTW, *PPOLYTEXTW, *LPPOLYTEXTW;
1065
1066 DECL_WINELIB_TYPE_AW(POLYTEXT)
1067 DECL_WINELIB_TYPE_AW(PPOLYTEXT)
1068 DECL_WINELIB_TYPE_AW(LPPOLYTEXT)
1069
1070
1071 /* ntmFlags field flags */
1072 #define NTM_REGULAR    0x00000040L
1073 #define NTM_BOLD       0x00000020L
1074 #define NTM_ITALIC     0x00000001L
1075
1076 typedef struct
1077 {
1078     LONG           tmHeight;
1079     LONG           tmAscent;
1080     LONG           tmDescent;
1081     LONG           tmInternalLeading;
1082     LONG           tmExternalLeading;
1083     LONG           tmAveCharWidth;
1084     LONG           tmMaxCharWidth;
1085     LONG           tmWeight;
1086     LONG           tmOverhang;
1087     LONG           tmDigitizedAspectX;
1088     LONG           tmDigitizedAspectY;
1089     BYTE           tmFirstChar;
1090     BYTE           tmLastChar;
1091     BYTE           tmDefaultChar;
1092     BYTE           tmBreakChar;
1093     BYTE           tmItalic;
1094     BYTE           tmUnderlined;
1095     BYTE           tmStruckOut;
1096     BYTE           tmPitchAndFamily;
1097     BYTE           tmCharSet;
1098     DWORD          ntmFlags;
1099     UINT           ntmSizeEM;
1100     UINT           ntmCellHeight;
1101     UINT           ntmAvgWidth;
1102 } NEWTEXTMETRICA, *PNEWTEXTMETRICA, *LPNEWTEXTMETRICA;
1103
1104 typedef struct
1105 {

```

```

1106     LONG         tmHeight;
1107     LONG         tmAscent;
1108     LONG         tmDescent;
1109     LONG         tmInternalLeading;
1110     LONG         tmExternalLeading;
1111     LONG         tmAveCharWidth;
1112     LONG         tmMaxCharWidth;
1113     LONG         tmWeight;
1114     LONG         tmOverhang;
1115     LONG         tmDigitizedAspectX;
1116     LONG         tmDigitizedAspectY;
1117     WCHAR        tmFirstChar;
1118     WCHAR        tmLastChar;
1119     WCHAR        tmDefaultChar;
1120     WCHAR        tmBreakChar;
1121     BYTE         tmItalic;
1122     BYTE         tmUnderlined;
1123     BYTE         tmStruckOut;
1124     BYTE         tmPitchAndFamily;
1125     BYTE         tmCharSet;
1126     DWORD        ntmFlags;
1127     UINT         ntmSizeEM;
1128     UINT         ntmCellHeight;
1129     UINT         ntmAvgWidth;
1130 } NEWTEXTMETRICW, *PNEWTEXTMETRICW, *LPNEWTEXTMETRICW;
1131
1132 DECL_WINELIB_TYPE_AW (NEWTEXTMETRIC)
1133 DECL_WINELIB_TYPE_AW (PNEWTEXTMETRIC)
1134 DECL_WINELIB_TYPE_AW (LPNEWTEXTMETRIC)
1135
1136 typedef struct
1137 {
1138     NEWTEXTMETRICA ntmIm;
1139     FONTSIGNATURE  ntmFontSig;
1140 } NEWTEXTMETRICEXA, *LPNEWTEXTMETRICEXA;
1141
1142 typedef struct
1143 {
1144     NEWTEXTMETRICW ntmIm;
1145     FONTSIGNATURE  ntmFontSig;
1146 } NEWTEXTMETRICEXW, *LPNEWTEXTMETRICEXW;
1147
1148 DECL_WINELIB_TYPE_AW (NEWTEXTMETRICEX)
1149 DECL_WINELIB_TYPE_AW (LPNEWTEXTMETRICEX)
1150
1151 typedef int CALLBACK (*OLDFONTENUMPROCA) (const LOGFONTA*, const TEXTMETRICA*,
1152                                           DWORD, LPARAM);
1153 typedef int CALLBACK (*OLDFONTENUMPROCW) (const LOGFONTW*, const TEXTMETRICW*,
1154                                           DWORD, LPARAM);
1155 DECL_WINELIB_TYPE_AW (OLDFONTENUMPROC)
1156
1157 typedef OLDFONTENUMPROCA FONTENUMPROCA;
1158 typedef OLDFONTENUMPROCW FONTENUMPROCW;
1159 DECL_WINELIB_TYPE_AW (FONTENUMPROC)
1160
1161 typedef int CALLBACK (*FONTENUMPROCEXA) (LPENUMLOGFONTEXA, LPNEWTEXTMETRICEXA, DWORD, LPARAM);
1162 typedef int CALLBACK (*FONTENUMPROCEXW) (LPENUMLOGFONTEXW, LPNEWTEXTMETRICEXW, DWORD, LPARAM);
1163 DECL_WINELIB_TYPE_AW (FONTENUMPROCEX)
1164
1165 typedef INT        CALLBACK (*GOBJENUMPROC) (LPVOID, LPARAM);
1166 typedef VOID        CALLBACK (*LINEDDAPROC) (INT, INT, LPARAM);
1167
1168 /* tmPitchAndFamily bits */
1169 #define TMPF_FIXED_PITCH    1          /* means variable pitch */
1170 #define TMPF_VECTOR         2
1171 #define TMPF_TRUETYPE       4
1172 #define TMPF_DEVICE         8
1173
1174 /* Text alignment */
1175 #define TA_NOUPDATECP        0x00
1176 #define TA_UPDATECP         0x01
1177 #define TA_LEFT              0x00
1178 #define TA_RIGHT             0x02
1179 #define TA_CENTER            0x06
1180 #define TA_TOP               0x00
1181 #define TA_BOTTOM            0x08
1182 #define TA_BASELINE          0x18
1183 #define TA_RTLREADING        0x100
1184 #define TA_MASK              TA_BASELINE+TA_CENTER+TA_UPDATECP+TA_RTLREADING
1185
1186 #define VTA_BASELINE         TA_BASELINE
1187 #define VTA_LEFT             TA_BOTTOM
1188 #define VTA_RIGHT            TA_TOP
1189 #define VTA_CENTER           TA_CENTER
1190 #define VTA_BOTTOM           TA_RIGHT
1191 #define VTA_TOP              TA_LEFT
1192

```

```

1193
1194  /* ExtTextOut() parameters */
1195 #define ETO_GRAYED          0x0001
1196 #define ETO_OPAQUE          0x0002
1197 #define ETO_CLIPPED         0x0004
1198 #define ETO_GLYPH_INDEX    0x0010
1199 #define ETO_RTLREADING      0x0080
1200 #define ETO_IGNORELANGUAGE 0x1000
1201
1202 #define ASPECT_FILTERING    0x0001
1203
1204 typedef struct
1205 {
1206     UINT      gmBlackBoxX;
1207     UINT      gmBlackBoxY;
1208     POINT      gmptGlyphOrigin;
1209     SHORT      gmCellIncX;
1210     SHORT      gmCellIncY;
1211 } GLYPHMETRICS, *LPGLYPHMETRICS;
1212
1213
1214 #define GGO_METRICS          0
1215 #define GGO_BITMAP          1
1216 #define GGO_NATIVE          2
1217 #define GGO_GRAY2_BITMAP    4
1218 #define GGO_GRAY4_BITMAP    5
1219 #define GGO_GRAY8_BITMAP    6
1220 #define GGO_GLYPH_INDEX     0x80
1221
1222 typedef struct
1223 {
1224     WORD      fract;
1225     SHORT      value;
1226 } FIXED;
1227
1228 typedef struct tagPOINTFX
1229 {
1230     FIXED x;
1231     FIXED y;
1232 } POINTFX, *LPPOINTFX;
1233
1234 typedef struct tagTTPOLYCURVE
1235 {
1236     WORD wType;
1237     WORD cpx;
1238     POINTFX apfx[1];
1239 } TTPOLYCURVE, *LPTTPOLYCURVE;
1240
1241 typedef struct tagTTPOLYGONHEADER
1242 {
1243     DWORD cb;
1244     DWORD dwType;
1245     POINTFX pfxStart;
1246 } TTPOLYGONHEADER, *LPTTPOLYGONHEADER;
1247
1248 typedef struct
1249 {
1250     FIXED eM11;
1251     FIXED eM12;
1252     FIXED eM21;
1253     FIXED eM22;
1254 } MAT2, *LPMAT2;
1255
1256  /* for GetCharABCWidths() */
1257 typedef struct
1258 {
1259     INT      abcA;
1260     UINT      abcB;
1261     INT      abcC;
1262 } ABC, *PABC, *LPABC;
1263
1264
1265  /* for GetCharacterPlacement () */
1266
1267 #define GCP_DBCS          0x0001
1268 #define GCP_REORDER       0x0002
1269 #define GCP_USEKERNING    0x0008
1270 #define GCP_GLYPHSHAPE    0x0010
1271 #define GCP_LIGATE        0x0020
1272 #define GCP_DIACRITIC    0x0100
1273 #define GCP_KASHIDA       0x0200
1274 #define GCP_ERROR         0x8000
1275 #define FLI_MASK          0x103b
1276 #define GCP_JUSTIFY        0x00010000L
1277 #define FLI_GLYPHS        0x00040000L
1278 #define GCP_CLASSIN       0x00080000L
1279 #define GCP_MAXEXTENT      0x00100000L

```

```

1280 #define GCP_JUSTIFYIN      0x00200000L
1281 #define GCP_DISPLAYZWG    0x00400000L
1282 #define GCP_SYMSWAPOFF    0x00800000L
1283 #define GCP_NUMERICOVERRIDE 0x01000000L
1284 #define GCP_NEUTRALOVERRIDE 0x02000000L
1285 #define GCP_NUMERICSLATIN  0x04000000L
1286 #define GCP_NUMERICSLCAL  0x08000000L
1287
1288 #define GCPCLASS_LATIN      1
1289 #define GCPCLASS_HEBREW    2
1290 #define GCPCLASS_ARABIC    3
1291 #define GCPCLASS_NEUTRAL   4
1292 #define GCPCLASS_LOCALNUMBER 5
1293 #define GCPCLASS_LATINNUMBER 6
1294 #define GCPCLASS_LATINNUMERICTERMINATOR 7
1295 #define GCPCLASS_LATINNUMERICSEPARATOR 8
1296 #define GCPCLASS_NUMERICSEPARATOR 9
1297 #define GCPCLASS_PREBOUNDLTR 0x80
1298 #define GCPCLASS_PREBOUNDRLT 0x40
1299 #define GCPCLASS_POSTBOUNDLTR 0x20
1300 #define GCPCLASS_POSTBOUNDRTL 0x10
1301
1302 #define GCPGLYPH_LINKBEFORE 0x8000
1303 #define GCPGLYPH_LINKAFTER 0x4000
1304
1305
1306 typedef struct tagGCP_RESULTS {
1307     DWORD lStructSize;
1308     LPSTR lpOutString;
1309     UINT *lpOrder;
1310     INT *lpDx;
1311     INT *lpCaretPos;
1312     LPSTR lpClass;
1313     LPWSTR lpGlyphs;
1314     UINT nGlyphs;
1315     UINT nMaxFit;
1316 } GCP_RESULTS, *LPGCP_RESULTS;
1317
1318 typedef struct tagGCP_RESULTSW
1319 {
1320     DWORD lStructSize;
1321     LPWSTR lpOutString;
1322     UINT *lpOrder;
1323     INT *lpDx;
1324     INT *lpCaretPos;
1325     LPWSTR lpClass;
1326     LPWSTR lpGlyphs;
1327     UINT nGlyphs;
1328     UINT nMaxFit;
1329 } GCP_RESULTSW, *LPGCP_RESULTSW;
1330
1331 DECL_WINELIB_TYPE_AW(GCP_RESULTS)
1332 DECL_WINELIB_TYPE_AW(LPGCP_RESULTS)
1333
1334 /* Rasterizer status */
1335 typedef struct
1336 {
1337     SHORT nSize;
1338     SHORT wFlags;
1339     SHORT nLanguageID;
1340 } RASTERIZER_STATUS, *LPRASTERIZER_STATUS;
1341
1342 #define TT_AVAILABLE      0x0001
1343 #define TT_ENABLED        0x0002
1344
1345 #define TT_PRIM_LINE      1
1346 #define TT_PRIM_QSPLINE  2
1347 #define TT_POLYGON_TYPE   24
1348
1349 /* Get/SetSystemPaletteUse() values */
1350 #define SYSPAL_ERROR      0
1351 #define SYSPAL_STATIC     1
1352 #define SYSPAL_NOSTATIC   2
1353
1354 typedef struct tagPALETTEENTRY
1355 {
1356     BYTE peRed, peGreen, peBlue, peFlags;
1357 } PALETTEENTRY, *PPALETTEENTRY, *LPPALETTEENTRY;
1358
1359 /* Logical palette entry flags */
1360 #define PC_RESERVED      0x01
1361 #define PC_EXPLICIT      0x02
1362 #define PC_NOCOLLAPSE    0x04
1363
1364 typedef struct tagLOGPALETTE
1365 {
1366     WORD palVersion;

```

```

1367     WORD          palNumEntries;
1368     PALETTEENTRY   palPalEntry[1];
1369 } LOGPALETTE, *PLOGPALETTE, *LPLOGPALETTE, *NPLOGPALETTE;
1370
1371 /* Pens */
1372
1373 typedef struct
1374 {
1375     UINT      lopnStyle;
1376     POINT      lopnWidth;
1377     COLORREF   lopnColor;
1378 } LOGPEN, *LPLOGPEN;
1379
1380
1381 typedef struct tagEXTLOGPEN
1382 {
1383     DWORD      elpPenStyle;
1384     DWORD      elpWidth;
1385     UINT      elpBrushStyle;
1386     COLORREF   elpColor;
1387     LONG       elpHatch;
1388     DWORD      elpNumEntries;
1389     DWORD      elpStyleEntry[1];
1390 } EXTLOGPEN, *PEXTLOGPEN, *NPEXTLOGPEN, *LPEXTLOGPEN;
1391
1392 #define PS_SOLID          0x00000000
1393 #define PS_DASH           0x00000001
1394 #define PS_DOT            0x00000002
1395 #define PS_DASHDOT        0x00000003
1396 #define PS_DASHDOTDOT     0x00000004
1397 #define PS_NULL           0x00000005
1398 #define PS_INSIDEFRAME    0x00000006
1399 #define PS_USERSTYLE      0x00000007
1400 #define PS_ALTERNATE      0x00000008
1401 #define PS_STYLE_MASK     0x0000000f
1402
1403 #define PS_ENDCAP_ROUND    0x00000000
1404 #define PS_ENDCAP_SQUARE  0x00000100
1405 #define PS_ENDCAP_FLAT    0x00000200
1406 #define PS_ENDCAP_MASK    0x00000f00
1407
1408 #define PS_JOIN_ROUND      0x00000000
1409 #define PS_JOIN_BEVEL      0x00001000
1410 #define PS_JOIN_MITER      0x00002000
1411 #define PS_JOIN_MASK       0x0000f000
1412
1413 #define PS_COSMETIC        0x00000000
1414 #define PS_GEOMETRIC       0x00010000
1415 #define PS_TYPE_MASK       0x000f0000
1416
1417 /* Regions */
1418
1419 #define ERROR              0
1420 #define NULLREGION         1
1421 #define SIMPLEREGION       2
1422 #define COMPLEXREGION      3
1423 #define RGN_ERROR          ERROR
1424
1425 #define RGN_AND             1
1426 #define RGN_OR              2
1427 #define RGN_XOR             3
1428 #define RGN_DIFF            4
1429 #define RGN_COPY            5
1430 #define RGN_MIN             RGN_AND
1431 #define RGN_MAX             RGN_COPY
1432 /* Device contexts */
1433
1434 /* Polygon modes */
1435 #define ALTERNATE           1
1436 #define WINDING             2
1437 #define POLYFILL_LAST      2
1438
1439 /* Background modes */
1440 /* Apparently some broken svr4 includes define TRANSPARENT */
1441 #undef TRANSPARENT
1442 #define TRANSPARENT         1
1443 #define OPAQUE              2
1444 #define BKMODE_LAST        2
1445
1446 /* Graphics Modes */
1447 #define GM_COMPATIBLE       1
1448 #define GM_ADVANCED         2
1449 #define GM_LAST            2
1450
1451 /* Arc direction modes */
1452 #define AD_COUNTERCLOCKWISE 1
1453 #define AD_CLOCKWISE        2

```



```
1454
1455  /* Map modes */
1456 #define MM_TEXT 1
1457 #define MM_LOMETRIC 2
1458 #define MM_HIMETRIC 3
1459 #define MM_LOENGLISH 4
1460 #define MM_HIENGLISH 5
1461 #define MM_TWIPS 6
1462 #define MM_ISOTROPIC 7
1463 #define MM_ANISOTROPIC 8
1464
1465 #define MM_MIN MM_TEXT
1466 #define MM_MAX MM_ANISOTROPIC
1467 #define MM_MAX_FIXEDSCALE MM_TWIPS
1468
1469  /* Coordinate modes */
1470 #define ABSOLUTE 1
1471 #define RELATIVE 2
1472
1473  /* Flood fill modes */
1474 #define FLOODFILLBORDER 0
1475 #define FLOODFILLSURFACE 1
1476
1477  /* Device parameters for GetDeviceCaps() */
1478 #define DRIVERVERSION 0
1479 #define TECHNOLOGY 2
1480 #define HORZSIZE 4
1481 #define VERTSIZE 6
1482 #define HORZRES 8
1483 #define VERTRES 10
1484 #define BITSPIXEL 12
1485 #define PLANES 14
1486 #define NUMBRUSHES 16
1487 #define NUMPENS 18
1488 #define NUMMARKERS 20
1489 #define NUMFONTS 22
1490 #define NUMCOLORS 24
1491 #define PDEVICESIZE 26
1492 #define CURVECAPS 28
1493 #define LINECAPS 30
1494 #define POLYGONALCAPS 32
1495 #define TEXTCAPS 34
1496 #define CLIPCAPS 36
1497 #define RASTERCAPS 38
1498 #define ASPECTX 40
1499 #define ASPECTY 42
1500 #define ASPECTXY 44
1501 #define LOGPIXELSX 88
1502 #define LOGPIXELSY 90
1503 #define CAPS1 94
1504 #define SIZEPALETTE 104
1505 #define NUMRESERVED 106
1506 #define COLORRES 108
1507
1508 #define PHYSICALWIDTH 110
1509 #define PHYSICALHEIGHT 111
1510 #define PHYSICALOFFSETX 112
1511 #define PHYSICALOFFSETY 113
1512 #define SCALINGFACTORX 114
1513 #define SCALINGFACTORY 115
1514 #define VREFRESH 116
1515 #define DESKTOPVERTRES 117
1516 #define DESKTOPHORZRES 118
1517 #define BTLALIGNMENT 119
1518
1519  /* TECHNOLOGY */
1520 #define DT_PLOTTER 0
1521 #define DT_RASDISPLAY 1
1522 #define DT_RASPRINTER 2
1523 #define DT_RASCAMERA 3
1524 #define DT_CHARSTREAM 4
1525 #define DT_METAFILE 5
1526 #define DT_DISPPFILE 6
1527
1528  /* CURVECAPS */
1529 #define CC_NONE 0x0000
1530 #define CC_CIRCLES 0x0001
1531 #define CC_PIE 0x0002
1532 #define CC_CHORD 0x0004
1533 #define CC_ELLIPSES 0x0008
1534 #define CC_WIDE 0x0010
1535 #define CC_STYLED 0x0020
1536 #define CC_WIDESTYLED 0x0040
1537 #define CC_INTERIORS 0x0080
1538 #define CC_ROUNDRECT 0x0100
1539
1540  /* LINECAPS */
```

```
1541 #define LC_NONE 0x0000
1542 #define LC_POLYLINE 0x0002
1543 #define LC_MARKER 0x0004
1544 #define LC_POLYMARKER 0x0008
1545 #define LC_WIDE 0x0010
1546 #define LC_STYLED 0x0020
1547 #define LC_WIDESTYLED 0x0040
1548 #define LC_INTERIORS 0x0080
1549
1550 /* POLYGONALCAPS */
1551 #define PC_NONE 0x0000
1552 #define PC_POLYGON 0x0001
1553 #define PC_RECTANGLE 0x0002
1554 #define PC_WINDPOLYGON 0x0004
1555 #define PC_TRAPEZOID 0x0004
1556 #define PC_SCANLINE 0x0008
1557 #define PC_WIDE 0x0010
1558 #define PC_STYLED 0x0020
1559 #define PC_WIDESTYLED 0x0040
1560 #define PC_INTERIORS 0x0080
1561 #define PC_POLYPOLYGON 0x0100
1562 #define PC_PATHS 0x0200
1563
1564 /* TEXTCAPS */
1565 #define TC_OP_CHARACTER 0x0001
1566 #define TC_OP_STROKE 0x0002
1567 #define TC_CP_STROKE 0x0004
1568 #define TC_CR_90 0x0008
1569 #define TC_CR_ANY 0x0010
1570 #define TC_SF_X_YINDEP 0x0020
1571 #define TC_SA_DOUBLE 0x0040
1572 #define TC_SA_INTEGER 0x0080
1573 #define TC_SA_CONTIN 0x0100
1574 #define TC_EA_DOUBLE 0x0200
1575 #define TC_IA_ABLE 0x0400
1576 #define TC_UA_ABLE 0x0800
1577 #define TC_SO_ABLE 0x1000
1578 #define TC_RA_ABLE 0x2000
1579 #define TC_VA_ABLE 0x4000
1580 #define TC_RESERVED 0x8000
1581 #define TC_SCROLLBLT 0x00010000
1582
1583 /* CLIPCAPS */
1584 #define CP_NONE 0x0000
1585 #define CP_RECTANGLE 0x0001
1586 #define CP_REGION 0x0002
1587
1588 /* RASTERCAPS */
1589 #define RC_NONE 0x0000
1590 #define RC_BITBLT 0x0001
1591 #define RC_BANDING 0x0002
1592 #define RC_SCALING 0x0004
1593 #define RC_BITMAP64 0x0008
1594 #define RC_GDI20_OUTPUT 0x0010
1595 #define RC_GDI20_STATE 0x0020
1596 #define RC_SAVEBITMAP 0x0040
1597 #define RC_DI_BITMAP 0x0080
1598 #define RC_PALETTE 0x0100
1599 #define RC_DIBTODEV 0x0200
1600 #define RC_BIGFONT 0x0400
1601 #define RC_STRETCHBLT 0x0800
1602 #define RC_FLOODFILL 0x1000
1603 #define RC_STRETCHDIB 0x2000
1604 #define RC_OP_DX_OUTPUT 0x4000
1605 #define RC_DEVBITS 0x8000
1606
1607 /* CAPS1 */
1608
1609 #define Cl_TRANSPARENT 0x0001
1610 #define TC_TT_ABLE 0x0002
1611 #define Cl_TT_CR_ANY 0x0004
1612 #define Cl_EMF_COMPLIANT 0x0008
1613 #define Cl_DIBENGINE 0x0010
1614 #define Cl_GAMMA_RAMP 0x0040
1615 #define Cl_REINIT_ABLE 0x0080
1616 #define Cl_GLYPH_INDEX 0x0100
1617 #define Cl_BIT_PACKED 0x0200
1618 #define Cl_BYTE_PACKED 0x0400
1619 #define Cl_COLORCURSOR 0x0800
1620 #define Cl_CMYK_ABLE 0x1000
1621 #define Cl_SLOW_CARD 0x2000
1622
1623 /* Device-independent bitmaps */
1624
1625 typedef struct {
1626     BYTE rgbBlue;
1627     BYTE rgbGreen;
```

```
1628     BYTE rgbRed;
1629     BYTE rgbReserved;
1630 } RGBQUAD, *LPRGBQUAD;
1631
1632 typedef struct {
1633     BYTE rgbtBlue;
1634     BYTE rgbtGreen;
1635     BYTE rgbtRed;
1636 } RGBTRIPLE;
1637
1638 #include "pshpack2.h"
1639 typedef struct
1640 {
1641     WORD    bfType;
1642     DWORD   bfSize;
1643     WORD    bfReserved1;
1644     WORD    bfReserved2;
1645     DWORD   bfOffBits;
1646 } BITMAPFILEHEADER, *PBITMAPFILEHEADER, *LPBITMAPFILEHEADER;
1647 #include "poppack.h"
1648
1649 #define MAKEPOINTS(l)    ((POINTS *)&(l))
1650
1651 typedef struct
1652 {
1653     DWORD   biSize;
1654     LONG    biWidth;
1655     LONG    biHeight;
1656     WORD    biPlanes;
1657     WORD    biBitCount;
1658     DWORD   biCompression;
1659     DWORD   biSizeImage;
1660     LONG    biXPelsPerMeter;
1661     LONG    biYPelsPerMeter;
1662     DWORD   biClrUsed;
1663     DWORD   biClrImportant;
1664 } BITMAPINFOHEADER, *PBITMAPINFOHEADER, *LPBITMAPINFOHEADER;
1665
1666 typedef struct
1667 {
1668     DWORD   bv4Size;
1669     LONG    bv4Width;
1670     LONG    bv4Height;
1671     WORD    bv4Planes;
1672     WORD    bv4BitCount;
1673     DWORD   bv4Compression;
1674     DWORD   bv4SizeImage;
1675     LONG    bv4XPelsPerMeter;
1676     LONG    bv4YPelsPerMeter;
1677     DWORD   bv4ClrUsed;
1678     DWORD   bv4ClrImportant;
1679     DWORD   bv4RedMask;
1680     DWORD   bv4GreenMask;
1681     DWORD   bv4BlueMask;
1682     DWORD   bv4AlphaMask;
1683     DWORD   bv4CSType;
1684     CIEXYZTRIPLE bv4EndPoints;
1685     DWORD   bv4GammaRed;
1686     DWORD   bv4GammaGreen;
1687     DWORD   bv4GammaBlue;
1688 } BITMAPV4HEADER, *PBITMAPV4HEADER;
1689
1690 typedef struct {
1691     DWORD   bv5Size;
1692     LONG    bv5Width;
1693     LONG    bv5Height;
1694     WORD    bv5Planes;
1695     WORD    bv5BitCount;
1696     DWORD   bv5Compression;
1697     DWORD   bv5SizeImage;
1698     LONG    bv5XPelsPerMeter;
1699     LONG    bv5YPelsPerMeter;
1700     DWORD   bv5ClrUsed;
1701     DWORD   bv5ClrImportant;
1702     DWORD   bv5RedMask;
1703     DWORD   bv5GreenMask;
1704     DWORD   bv5BlueMask;
1705     DWORD   bv5AlphaMask;
1706     DWORD   bv5CSType;
1707     CIEXYZTRIPLE bv5Endpoints;
1708     DWORD   bv5GammaRed;
1709     DWORD   bv5GammaGreen;
1710     DWORD   bv5GammaBlue;
1711     DWORD   bv5Intent;
1712     DWORD   bv5ProfileData;
1713     DWORD   bv5ProfileSize;
1714     DWORD   bv5Reserved;
```

```

1715 } BITMAPV5HEADER, *LPBITMAPV5HEADER, *PBITMAPV5HEADER;
1716
1717 #define PROFILE_LINKED 'LINK'
1718 #define PROFILE_EMBEDDED 'MBED'
1719
1720
1721 /* biCompression */
1722 #define BI_RGB 0
1723 #define BI_RLE8 1
1724 #define BI_RLE4 2
1725 #define BI_BITFIELDS 3
1726
1727 typedef struct {
1728     BITMAPINFOHEADER bmiHeader;
1729     RGBQUAD bmiColors[1];
1730 } BITMAPINFO, *PBITMAPINFO, *LPBITMAPINFO;
1731
1732 typedef struct
1733 {
1734     DWORD    bcSize;
1735     WORD     bcWidth;
1736     WORD     bcHeight;
1737     WORD     bcPlanes;
1738     WORD     bcBitCount;
1739 } BITMAPCOREHEADER, *PBITMAPCOREHEADER, *LPBITMAPCOREHEADER;
1740
1741 typedef struct
1742 {
1743     BITMAPCOREHEADER bmciHeader;
1744     RGBTRIPLE    bmciColors[1];
1745 } BITMAPCOREINFO, *PBITMAPCOREINFO, *LPBITMAPCOREINFO;
1746
1747 #define DIB_RGB_COLORS 0
1748 #define DIB_PAL_COLORS 1
1749 #define CBM_INIT 4
1750
1751 typedef struct
1752 {
1753     BITMAP    dsBm;
1754     BITMAPINFOHEADER    dsBmih;
1755     DWORD    dsBitFields[3];
1756     HANDLE    dshSection;
1757     DWORD    dsOffset;
1758 } DIBSECTION, *PDIBSECTION, *LPDIBSECTION;
1759
1760 /* Stock GDI objects for GetStockObject() */
1761
1762 #define WHITE_BRUSH 0
1763 #define LTGRAY_BRUSH 1
1764 #define GRAY_BRUSH 2
1765 #define DKGRAY_BRUSH 3
1766 #define BLACK_BRUSH 4
1767 #define NULL_BRUSH 5
1768 #define HOLLOW_BRUSH 5
1769 #define WHITE_PEN 6
1770 #define BLACK_PEN 7
1771 #define NULL_PEN 8
1772 #define OEM_FIXED_FONT 10
1773 #define ANSI_FIXED_FONT 11
1774 #define ANSI_VAR_FONT 12
1775 #define SYSTEM_FONT 13
1776 #define DEVICE_DEFAULT_FONT 14
1777 #define DEFAULT_PALETTE 15
1778 #define SYSTEM_FIXED_FONT 16
1779 #define DEFAULT_GUI_FONT 17
1780
1781 #define STOCK_LAST 17
1782
1783 #define CLR_INVALID 0xffffffff
1784 /* Metafile header structure */
1785 #include "pshpack2.h"
1786 typedef struct
1787 {
1788     WORD    mtType;
1789     WORD    mtHeaderSize;
1790     WORD    mtVersion;
1791     DWORD    mtSize;
1792     WORD    mtNoObjects;
1793     DWORD    mtMaxRecord;
1794     WORD    mtNoParameters;
1795 } METAHEADER, *PMETAHEADER, *LPMETAHEADER;
1796 #include "poppack.h"
1797
1798 /* Metafile typical record structure */
1799 typedef struct
1800 {
1801     DWORD    rdSize;

```

```
1802     WORD        rdFunction;
1803     WORD        rdParm[1];
1804 } METARECORD, *PMETARECORD, *LPMETARECORD;
1805
1806 /* Handle table structure */
1807
1808 typedef struct
1809 {
1810     HGDIOBJ objectHandle[1];
1811 } HANDLETABLE, *PHANDLETABLE, *LPHANDLETABLE;
1812
1813
1814 /* Clipboard metafile picture structure */
1815 typedef struct
1816 {
1817     LONG        mm;
1818     LONG        xExt;
1819     LONG        yExt;
1820     HMETAFILE    hMF;
1821 } METAFILEPICT, *LPMETAFILEPICT;
1822
1823
1824 /* Metafile functions */
1825 #define META_SETBKCOLOR            0x0201
1826 #define META_SETBKMODE            0x0102
1827 #define META_SETMAPMODE            0x0103
1828 #define META_SETROP2              0x0104
1829 #define META_SETRELABS            0x0105
1830 #define META_SETPOLYFILLMODE      0x0106
1831 #define META_SETSTRETCHBLTMODE    0x0107
1832 #define META_SETTEXTCHAREXTRA     0x0108
1833 #define META_SETTEXTCOLOR         0x0209
1834 #define META_SETTEXTJUSTIFICATION 0x020A
1835 #define META_SETWINDOWORG         0x020B
1836 #define META_SETWINDOWEXT         0x020C
1837 #define META_SETVIEWPORTORG       0x020D
1838 #define META_SETVIEWPORTEXT       0x020E
1839 #define META_OFFSETWINDOWORG      0x020F
1840 #define META_SCALEWINDOWEXT       0x0410
1841 #define META_OFFSETVIEWPORTORG    0x0211
1842 #define META_SCALEVIEWPORTEXT     0x0412
1843 #define META_LINETO               0x0213
1844 #define META_MOVETO               0x0214
1845 #define META_EXCLUDECLIPRECT      0x0415
1846 #define META_INTERSECTCLIPRECT    0x0416
1847 #define META_ARC                  0x0817
1848 #define META_ELLIPSE              0x0418
1849 #define META_FLOODFILL            0x0419
1850 #define META_PIE                  0x081A
1851 #define META_RECTANGLE            0x041B
1852 #define META_ROUNDRECT            0x061C
1853 #define META_PATBLT               0x061D
1854 #define META_SAVEDC               0x001E
1855 #define META_SETPIXEL             0x041F
1856 #define META_OFFSETCLIPRGN        0x0220
1857 #define META_TEXTOUT              0x0521
1858 #define META_BITBLT               0x0922
1859 #define META_STRETCHBLT           0x0B23
1860 #define META_POLYGON              0x0324
1861 #define META_POLYLINE             0x0325
1862 #define META_ESCAPE               0x0626
1863 #define META_RESTOREDC            0x0127
1864 #define META_FILLREGION           0x0228
1865 #define META_FRAMEREGION          0x0429
1866 #define META_INVERTREGION         0x012A
1867 #define META_PAINTREGION          0x012B
1868 #define META_SELECTCLIPREGION     0x012C
1869 #define META_SELECTOBJECT         0x012D
1870 #define META_SETTEXTALIGN         0x012E
1871 #define META_DRAWTEXT             0x062F
1872 #define META_CHORD                0x0830
1873 #define META_SETMAPPERFLAGS        0x0231
1874 #define META_EXTTEXTOUT           0x0A32
1875 #define META_SETDIBTODEV          0x0D33
1876 #define META_SELECTPALETTE        0x0234
1877 #define META_REALIZEPALETTE       0x0035
1878 #define META_ANIMATEPALETTE       0x0436
1879 #define META_SETPALENTRIES        0x0037
1880 #define META_POLYPOLYGON          0x0538
1881 #define META_RESIZEPALETTE        0x0139
1882 #define META_DIBBITBLT           0x0940
1883 #define META_DIBSTRETCHBLT        0x0B41
1884 #define META_DIBCREATEPATTERNBRUSH 0x0142
1885 #define META_STRETCHDIB           0x0F43
1886 #define META_EXTFLOODFILL         0x0548
1887 #define META_RESETDC              0x014C
1888 #define META_STARTDOC             0x014D
```

```

1889 #define META_STARTPAGE          0x004F
1890 #define META_ENDPAGE            0x0050
1891 #define META_ABORTDOC           0x0052
1892 #define META_ENDDOC             0x005E
1893 #define META_DELETEOBJECT       0x01F0
1894 #define META_CREATEPALETTE      0x00F7
1895 #define META_CREATEBRUSH        0x00F8
1896 #define META_CREATEPATTERNBRUSH 0x01F9
1897 #define META_CREATEPENINDIRECT  0x02FA
1898 #define META_CREATEFONTINDIRECT 0x02FB
1899 #define META_CREATEBRUSHINDIRECT 0x02FC
1900 #define META_CREATEBITMAPINDIRECT 0x02FD
1901 #define META_CREATEBITMAP       0x06FE
1902 #define META_CREATEREGION       0x06FF
1903 #define META_UNKNOWN            0x0529 /* FIXME: unknown meta magic */
1904
1905 typedef INT CALLBACK (*MFENUMPROC) (HDC, HANDLETABLE*, METARECORD*,
1906                                     INT, LPARAM);
1907
1908 /* enhanced metafile structures and functions */
1909
1910 /* note that ENHMETAHEADER is just a particular kind of ENHMETARECORD,
1911 ie. the header is just the first record in the metafile */
1912 typedef struct {
1913     DWORD iType;
1914     DWORD nSize;
1915     RECTL rclBounds;
1916     RECTL rclFrame;
1917     DWORD dSignature;
1918     DWORD nVersion;
1919     DWORD nBytes;
1920     DWORD nRecords;
1921     WORD nHandles;
1922     WORD sReserved;
1923     DWORD nDescription;
1924     DWORD offDescription;
1925     DWORD nPalEntries;
1926     SIZEL szlDevice;
1927     SIZEL szlMillimeters;
1928
1929     /* Fields for winver >= win95 */
1930     DWORD cbPixelFormat;
1931     DWORD offPixelFormat;
1932     DWORD bOpenGL;
1933 #if 1
1934     /* Fields for winver >= win98 */
1935     SIZEL szlMicrometers;
1936 #endif
1937 } ENHMETAHEADER, *PENHMETAHEADER, *LPENHMETAHEADER;
1938
1939 typedef struct {
1940     DWORD iType;
1941     DWORD nSize;
1942     DWORD dParm[1];
1943 } ENHMETARECORD, *LPENHMETARECORD;
1944
1945 typedef struct {
1946     DWORD iType;
1947     DWORD nSize;
1948 } EMR, *PEMR;
1949
1950 typedef struct {
1951     POINTL ptlReference;
1952     DWORD nChars;
1953     DWORD offString;
1954     DWORD fOptions;
1955     RECTL rcl;
1956     DWORD offDx;
1957 } EMRTEXT, *PEMRTEXT;
1958
1959 typedef struct {
1960     EMR emr;
1961 } EMRABORTPATH, *PEMRABORTPATH,
1962 EMRBEGINPATH, *PEMRBEGINPATH,
1963 EMRENDPATH, *PEMRENDPATH,
1964 EMRCLOSEFIGURE, *PEMRCLOSEFIGURE,
1965 EMRFLATTENPATH, *PEMRFLATTENPATH,
1966 EMRWIDENPATH, *PEMRWIDENPATH,
1967 EMRSETMETARGN, *PEMRSETMETARGN,
1968 EMRSAVEDC, *PEMRSAVEDC,
1969 EMRREALIZEPALETTE, *PEMRREALIZEPALETTE;
1970
1971 typedef struct {
1972     EMR emr;
1973     POINTL ptlCenter;
1974     DWORD nRadius;
1975     FLOAT eStartAngle;

```

```
1976     FLOAT   eSweepAngle;
1977 } EMRANGLEARC, *PEMRANGLEARC;
1978
1979 typedef struct {
1980     EMR      emr;
1981     RECTL    rclBox;
1982     POINTL   ptlStart;
1983     POINTL   ptlEnd;
1984 } EMRARC, *PEMRARC,
1985 EMRARCTO, *PEMRARCTO,
1986 EMRCHORD, *PEMRCHORD,
1987 EMRPIE, *PEMRPIE;
1988
1989 typedef struct {
1990     EMR      emr;
1991     RECTL    rclBounds;
1992     LONG     xDest;
1993     LONG     yDest;
1994     LONG     cxDest;
1995     LONG     cyDest;
1996     DWORD    dwRop;
1997     LONG     xSrc;
1998     LONG     ySrc;
1999     XFORM     xformSrc;
2000     COLORREF  crBkColorSrc;
2001     DWORD    iUsageSrc;
2002     DWORD    offBmiSrc;
2003     DWORD    cbBmiSrc;
2004     DWORD    offBitsSrc;
2005     DWORD    cbBitsSrc;
2006 } EMRBITBLT, *PEMRBITBLT;
2007
2008 typedef struct {
2009     EMR      emr;
2010     DWORD    ihBrush;
2011     LOGBRUSH lb;
2012 } EMRCREATEBRUSHINDIRECT, *PEMRCREATEBRUSHINDIRECT;
2013
2014 typedef struct {
2015     EMR      emr;
2016     DWORD    ihCS;
2017     LOGCOLORSPACEA lcs;
2018 } EMRCREATECOLORSPACE, *PEMRCREATECOLORSPACE;
2019
2020 typedef struct {
2021     EMR      emr;
2022     DWORD    ihCS;
2023     LOGCOLORSPACEW lcs;
2024     DWORD    dwFlags;
2025     DWORD    cbData;
2026     BYTE     Data[1];
2027 } EMRCREATECOLORSPACEW, *PEMRCREATECOLORSPACEW;
2028
2029 typedef struct {
2030     EMR      emr;
2031     DWORD    ihBrush;
2032     DWORD    iUsage;
2033     DWORD    offBmi;
2034     DWORD    cbBmi;
2035     DWORD    offBits;
2036     DWORD    cbBits;
2037 } EMRCREATEDIBPATTERNBRUSHPT, *PEMRCREATEDIBPATTERNBRUSHPT;
2038
2039 typedef struct {
2040     EMR      emr;
2041     DWORD    ihBrush;
2042     DWORD    iUsage;
2043     DWORD    offBmi;
2044     DWORD    cbBmi;
2045     DWORD    offBits;
2046     DWORD    cbBits;
2047 } EMRCREATEMONOBRUSH, *PEMRCREATEMONOBRUSH;
2048
2049 typedef struct {
2050     EMR      emr;
2051     DWORD    ihPal;
2052     LOGPALETTE lgpl;
2053 } EMRCREATEPALETTE, *PEMRCREATEPALETTE;
2054
2055 typedef struct {
2056     EMR      emr;
2057     DWORD    ihPen;
2058     LOGPEN   lopn;
2059 } EMRCREATEPEN, *PEMRCREATEPEN;
2060
2061 typedef struct {
2062     EMR      emr;
```

```

2063     DWORD          ihCS;
2064 } EMRDELETECOLORSPACE, *PEMRDELETECOLORSPACE,
2065 EMRSELECTCOLORSPACE, *PEMRSELECTCOLORSPACE,
2066 EMRSETCOLORSPACE, *PEMRSETCOLORSPACE;
2067
2068 typedef struct {
2069     EMR    emr;
2070     DWORD  ihObject;
2071 } EMRDELETEOBJECT, *PEMRDELETEOBJECT,
2072 EMRSELECTOBJECT, *PEMRSELECTOBJECT;
2073
2074 typedef struct {
2075     EMR    emr;
2076     RECTL  rclBox;
2077 } EMRELLIPSE, *PEMRELLIPSE,
2078 EMRRECTANGLE, *PEMRRECTANGLE;
2079
2080 typedef struct {
2081     EMR    emr;
2082     DWORD  nPalEntries;
2083     DWORD  offPalEntries;
2084     DWORD  nSizeLast;
2085 } EMREOF, *PEMREOF;
2086
2087 typedef struct {
2088     EMR    emr;
2089     RECTL  rclClip;
2090 } EMREXCLUDECLIPRECT, *PEMREXCLUDECLIPRECT,
2091 EMRINTERSECTCLIPRECT, *PEMRINTERSECTCLIPRECT;
2092
2093 typedef struct {
2094     EMR    emr;
2095     DWORD  ihFont;
2096     EXTLOGFONTW elfw;
2097 } EMREXTCREATEFONTINDIRECTW, *PEMREXTCREATEFONTINDIRECTW;
2098
2099 typedef struct {
2100     EMR    emr;
2101     DWORD  ihPen;
2102     DWORD  offBmi;
2103     DWORD  cbBmi;
2104     DWORD  offBits;
2105     DWORD  cbBits;
2106     EXTLOGPEN elp;
2107 } EMREXTCREATEPEN, *PEMREXTCREATEPEN;
2108
2109 typedef struct {
2110     EMR    emr;
2111     POINTL  ptlStart;
2112     COLORREF crColor;
2113     DWORD  iMode;
2114 } EMREXTFLOODFILL, *PEMREXTFLOODFILL;
2115
2116 typedef struct {
2117     EMR    emr;
2118     DWORD  cbRgnData;
2119     DWORD  iMode;
2120     BYTE  RgnData[1];
2121 } EMREXTSELECTCLIPRGN, *PEMREXTSELECTCLIPRGN;
2122
2123 typedef struct {
2124     EMR    emr;
2125     RECTL  rclBounds;
2126     DWORD  iGraphicsMode;
2127     FLOAT  exScale;
2128     FLOAT  eyScale;
2129     EMRTEXT emrtext;
2130 } EMREXTTEXTOUTA, *PEMREXTTEXTOUTA,
2131 EMREXTTEXTOUTW, *PEMREXTTEXTOUTW;
2132
2133 typedef struct {
2134     EMR    emr;
2135     RECTL  rclBounds;
2136 } EMRFILLPATH, *PEMRFILLPATH,
2137 EMRSTROKEANDFILLPATH, *PEMRSTROKEANDFILLPATH,
2138 EMRSTROKEPATH, *PEMRSTROKEPATH;
2139
2140 typedef struct {
2141     EMR    emr;
2142     RECTL  rclBounds;
2143     DWORD  cbRgnData;
2144     DWORD  ihBrush;
2145     BYTE  RgnData[1];
2146 } EMRFILLRGN, *PEMRFILLRGN;
2147
2148 typedef struct {
2149     DWORD  signature;

```



```

2150     DWORD nVersion;
2151     DWORD cbData;
2152     DWORD offData;
2153 } EMRFORMAT, *PEMRFORMAT;
2154
2155 typedef struct {
2156     EMR     emr;
2157     RECTL   rclBounds;
2158     DWORD   cbRgnData;
2159     DWORD   ihBrush;
2160     SIZEL   szlStroke;
2161     BYTE    RgnData[1];
2162 } EMRFRAMERGN, *PEMRFRAMERGN;
2163
2164 typedef struct {
2165     EMR     emr;
2166     DWORD   cbData;
2167     BYTE    Data[1];
2168 } EMRGDICOMMENT, *PEMRGDICOMMENT;
2169
2170 #if 0
2171 typedef struct {
2172     EMR     emr;
2173     RECTL   rclBounds;
2174     DWORD   nVer;
2175     DWORD   nTri;
2176     ULONG   ulMode;
2177     TRIVERTEX Ver[1];
2178 } EMRGRADIENTFILL, *PEMRGRADIENTFILL;
2179 #endif
2180
2181 typedef struct {
2182     EMR     emr;
2183     RECTL   rclBounds;
2184     DWORD   cbRgnData;
2185     BYTE    RgnData[1];
2186 } EMRINVERTRG, *PEMRINVERTRG,
2187 EMRPAINTRGN, *PEMRPAINTRGN;
2188
2189 typedef struct {
2190     EMR     emr;
2191     POINTL   ptl;
2192 } EMRLINETO, *PEMRLINETO,
2193 EMRMOVETOEX, *PEMRMOVETOEX;
2194
2195 typedef struct {
2196     EMR     emr;
2197     RECTL   rclBounds;
2198     LONG     xDest;
2199     LONG     yDest;
2200     LONG     cxDest;
2201     LONG     cyDest;
2202     DWORD    dwRop;
2203     LONG     xSrc;
2204     LONG     ySrc;
2205     XFORM     xformSrc;
2206     COLORREF  crBkColorSrc;
2207     DWORD     iUsageSrc;
2208     DWORD     offBmiSrc;
2209     DWORD     cbBmiSrc;
2210     DWORD     offBitsSrc;
2211     DWORD     cbBitsSrc;
2212     LONG     xMask;
2213     LONG     yMask;
2214     DWORD     iUsageMask;
2215     DWORD     offBmiMask;
2216     DWORD     cbBmiMask;
2217     DWORD     offBitsMask;
2218     DWORD     cbBitsMask;
2219 } EMRMASKBLT, *PEMRMASKBLT;
2220
2221 typedef struct {
2222     EMR     emr;
2223     XFORM     xform;
2224     DWORD     iMode;
2225 } EMRMODIFYWORLDTRANSFORM, *PEMRMODIFYWORLDTRANSFORM;
2226
2227 typedef struct {
2228     EMR     emr;
2229     POINTL   ptlOffset;
2230 } EMROFFSETCLIPRGN, *PEMROFFSETCLIPRGN;
2231
2232 typedef struct {
2233     EMR     emr;
2234     RECTL   rclBounds;
2235     POINTL   aptlDst[3];
2236     LONG     xSrc;

```

```
2237     LONG     ySrc;
2238     LONG     cxSrc;
2239     LONG     cySrc;
2240     XFORM     xformSrc;
2241     COLORREF  crBkColorSrc;
2242     DWORD     iUsageSrc;
2243     DWORD     offBmiSrc;
2244     DWORD     cbBmiSrc;
2245     DWORD     offBitsSrc;
2246     DWORD     cbBitsSrc;
2247     LONG     xMask;
2248     LONG     yMask;
2249     DWORD     iUsageMask;
2250     DWORD     offBmiMask;
2251     DWORD     cbBmiMask;
2252     DWORD     offBitsMask;
2253     DWORD     cbBitsMask;
2254 } EMRPLGBLT, *PEMRPLGBLT;
2255
2256 typedef struct {
2257     EMR     emr;
2258     RECTL   rclBounds;
2259     DWORD   cptl;
2260     POINTL  aptl[1];
2261 } EMRPOLYLINE, *PEMRPOLYLINE,
2262 EMRPOLYBEZIER, *PEMRPOLYBEZIER,
2263 EMRPOLYGON, *PEMRPOLYGON,
2264 EMRPOLYBEZIERTO, *PEMRPOLYBEZIERTO,
2265 EMRPOLYLINETO, *PEMRPOLYLINETO;
2266
2267 typedef struct {
2268     EMR     emr;
2269     RECTL   rclBounds;
2270     DWORD   cptl;
2271     POINTL  aptl[1];
2272     BYTE    abTypes[1];
2273 } EMRPOLYDRAW, *PEMRPOLYDRAW;
2274
2275 typedef struct {
2276     EMR     emr;
2277     RECTL   rclBounds;
2278     DWORD   nPolys;
2279     DWORD   cptl;
2280     DWORD   aPolyCounts[1];
2281     POINTL  aptl[1];
2282 } EMRPOLYPOLYLINE, *PEMRPOLYPOLYLINE,
2283 EMRPOLYPOLYGON, *PEMRPOLYPOLYGON;
2284
2285 typedef struct {
2286     EMR     emr;
2287     RECTL   rclBounds;
2288     DWORD   iGraphicsMode;
2289     FLOAT   exScale;
2290     FLOAT   eyScale;
2291     LONG    cStrings;
2292     EMRTEXT aemrtext[1];
2293 } EMRPOLYTEXTOUTA, *PEMRPOLYTEXTOUTA,
2294 EMRPOLYTEXTOUTW, *PEMRPOLYTEXTOUTW;
2295
2296 typedef struct {
2297     EMR     emr;
2298     DWORD   ihPal;
2299     DWORD   cEntries;
2300 } EMRRESIZEPALETTE, *PEMRRESIZEPALETTE;
2301
2302 typedef struct {
2303     EMR     emr;
2304     LONG    iRelative;
2305 } EMRRESTOREDC, *PEMRRESTOREDC;
2306
2307 typedef struct {
2308     EMR     emr;
2309     RECTL   rclBox;
2310     SIZEL   szlCorner;
2311 } EMRROUNDRECT, *PEMRROUNDRECT;
2312
2313 typedef struct {
2314     EMR     emr;
2315     LONG    xNum;
2316     LONG    xDenom;
2317     LONG    yNum;
2318     LONG    yDenom;
2319 } EMRSCALEVIEWPORTEXT, *PEMRSCALEVIEWPORTEXT,
2320 EMRSCALEWINDOWEXT, *PEMRSCALEWINDOWEXT;
2321
2322 typedef struct {
2323     EMR     emr;
```

```

2324     DWORD iMode;
2325 } EMRSELECTCLIPPATH, *PEMRSELECTCLIPPATH,
2326 EMRSETBKMODE, *PEMRSETBKMODE,
2327 EMRSETMAPMODE, *PEMRSETMAPMODE,
2328 EMRSETPOLYFILLMODE, *PEMRSETPOLYFILLMODE,
2329 EMRSETROP2, *PEMRSETROP2,
2330 EMRSETSTRETCHBLTMODE, *PEMRSETSTRETCHBLTMODE,
2331 EMRSETTEXTALIGN, *PEMRSETTEXTALIGN,
2332 EMRSETICMMODE, *PEMRSETICMMODE,
2333 EMRSETLAYOUT, *PEMRSETLAYOUT;
2334
2335 typedef struct {
2336     EMR emr;
2337     DWORD ihPal;
2338 } EMRSELECTPALETTE, *PEMRSELECTPALETTE;
2339
2340 typedef struct {
2341     EMR emr;
2342     DWORD iArcDirection;
2343 } EMRSETARCDIRECTION, *PEMRSETARCDIRECTION;
2344
2345 typedef struct {
2346     EMR emr;
2347     COLORREF crColor;
2348 } EMRSETBKCOLOR, *PEMRSETBKCOLOR,
2349 EMRSETTEXTCOLOR, *PEMRSETTEXTCOLOR;
2350
2351 typedef struct {
2352     EMR emr;
2353     POINTL ptlOrigin;
2354 } EMRSETBRUSHORGE, *PEMRSETBRUSHORGE,
2355 EMRSETVIEWPORTORGE, *PEMRSETVIEWPORTORGE,
2356 EMRSETWINDOWORGE, *PEMRSETWINDOWORGE;
2357
2358 typedef struct {
2359     EMR emr;
2360     COLORADJUSTMENT ColorAdjustment;
2361 } EMRSETCOLORADJUSTMENT, *PEMRSETCOLORADJUSTMENT;
2362
2363 typedef struct {
2364     EMR emr;
2365     RECTL rclBounds;
2366     LONG xDest;
2367     LONG yDest;
2368     LONG xSrc;
2369     LONG ySrc;
2370     LONG cxSrc;
2371     LONG cySrc;
2372     DWORD offBmiSrc;
2373     DWORD cbBmiSrc;
2374     DWORD offBitsSrc;
2375     DWORD cbBitsSrc;
2376     DWORD iUsageSrc;
2377     DWORD iStartScan;
2378     DWORD cScans;
2379 } EMRSETDIBITSTODEVICE, *PEMRSETDIBITSTODEVICE;
2380
2381 typedef struct {
2382     EMR emr;
2383     DWORD dwFlags;
2384 } EMRSETMAPPERFLAGS, *PEMRSETMAPPERFLAGS;
2385
2386 typedef struct {
2387     EMR emr;
2388     FLOAT eMiterLimit;
2389 } EMRSETMITERLIMIT, *PEMRSETMITERLIMIT;
2390
2391 typedef struct {
2392     EMR emr;
2393     DWORD ihPal;
2394     DWORD iStart;
2395     DWORD cEntries;
2396     PALETTEENTRY aPalEntries[1];
2397 } EMRSETPALETTEENTRIES, *PEMRSETPALETTEENTRIES;
2398
2399 typedef struct {
2400     EMR emr;
2401     POINTL ptlPixel;
2402     COLORREF crColor;
2403 } EMRSETPIXELV, *PEMRSETPIXELV;
2404
2405 typedef struct {
2406     EMR emr;
2407     SIZEL szlExtent;
2408 } EMRSETVIEWPORTEXT, *PEMRSETVIEWPORTEXT,
2409 EMRSETWINDOWEXT, *PEMRSETWINDOWEXT;
2410

```

```

2411 typedef struct {
2412     EMR      emr;
2413     XFORM    xform;
2414 } EMRSETWORLDTRANSFORM, *PEMRSETWORLDTRANSFORM;
2415
2416 typedef struct {
2417     EMR      emr;
2418     RECTL    rclBounds;
2419     LONG     xDest;
2420     LONG     yDest;
2421     LONG     cxDest;
2422     LONG     cyDest;
2423     DWORD    dwRop;
2424     LONG     xSrc;
2425     LONG     ySrc;
2426     XFORM    xformSrc;
2427     COLORREF crBkColorSrc;
2428     DWORD    iUsageSrc;
2429     DWORD    offBmiSrc;
2430     DWORD    cbBmiSrc;
2431     DWORD    offBitsSrc;
2432     DWORD    cbBitsSrc;
2433     LONG     cxSrc;
2434     LONG     cySrc;
2435 } EMRSTRETCHBLT, *PEMRSTRETCHBLT;
2436
2437 typedef struct {
2438     EMR      emr;
2439     RECTL    rclBounds;
2440     LONG     xDest;
2441     LONG     yDest;
2442     LONG     xSrc;
2443     LONG     ySrc;
2444     LONG     cxSrc;
2445     LONG     cySrc;
2446     DWORD    offBmiSrc;
2447     DWORD    cbBmiSrc;
2448     DWORD    offBitsSrc;
2449     DWORD    cbBitsSrc;
2450     DWORD    iUsageSrc;
2451     DWORD    dwRop;
2452     LONG     cxDest;
2453     LONG     cyDest;
2454 } EMRSTRETCHDIBITS, *PEMRSTRETCHDIBITS;
2455
2456 typedef struct {
2457     EMR      emr;
2458     PIXELFORMATDESCRIPTOR pfd;
2459 } EMRPIXELFORMAT, *PEMRPIXELFORMAT;
2460
2461 typedef struct tagEMRGLSRECORD {
2462     EMR      emr;
2463     DWORD    cbData;
2464     BYTE     Data[1];
2465 } EMRGLSRECORD, *PEMRGLSRECORD;
2466
2467 typedef struct {
2468     EMR      emr;
2469     RECTL    rclBounds;
2470     DWORD    cbData;
2471     BYTE     Data[1];
2472 } EMRGLSBOUNDEDRECORD, *PEMRGLSBOUNDEDRECORD;
2473
2474 typedef INT CALLBACK (*ENHMFENUMPROC) (HDC, LPHANDLETABLE,
2475     LPENHMETARECORD, INT, LPVOID);
2476
2477 #define EMR_HEADER 1
2478 #define EMR_POLYBEZIER 2
2479 #define EMR_POLYGON 3
2480 #define EMR_POLYLINE 4
2481 #define EMR_POLYBEZIERTO 5
2482 #define EMR_POLYLINETO 6
2483 #define EMR_POLYPOLYLINE 7
2484 #define EMR_POLYPOLYGON 8
2485 #define EMR_SETWINDOWEXTTEX 9
2486 #define EMR_SETWINDOWORGEX 10
2487 #define EMR_SETVIEWPORTEXTTEX 11
2488 #define EMR_SETVIEWPORTORGEX 12
2489 #define EMR_SETBRUSHORGEX 13
2490 #define EMR_EOF 14
2491 #define EMR_SETPIXELV 15
2492 #define EMR_SETMAPPERFLAGS 16
2493 #define EMR_SETMAPMODE 17
2494 #define EMR_SETBKMODE 18
2495 #define EMR_SETPOLYFILLMODE 19
2496 #define EMR_SETROP2 20
2497 #define EMR_SETSTRETCHBLTMODE 21

```

```
2498 #define EMR_SETTEXTALIGN      22
2499 #define EMRSetColorADJUSTMENT  23
2500 #define EMR_SETTEXTCOLOR      24
2501 #define EMR_SETBKCOLOR        25
2502 #define EMR_OFFSETCLIPRGN     26
2503 #define EMR_MOVETOEX          27
2504 #define EMR_SETMETARGN        28
2505 #define EMR_EXCLUDECLIPRECT   29
2506 #define EMR_INTERSECTCLIPRECT 30
2507 #define EMR_SCALEVIEWPORTEXTEX 31
2508 #define EMR_SCALEWINDOWEXTEX  32
2509 #define EMR_SAVEDC            33
2510 #define EMR_RESTOREDC         34
2511 #define EMR_SETWORLDTRANSFORM  35
2512 #define EMR_MODIFYWORLDTRANSFORM 36
2513 #define EMR_SELECTOBJECT      37
2514 #define EMR_CREATEPEN         38
2515 #define EMR_CREATEBRUSHINDIRECT 39
2516 #define EMR_DELETEOBJECT      40
2517 #define EMR_ANGLEARC          41
2518 #define EMR_ELLIPSE           42
2519 #define EMR_RECTANGLE          43
2520 #define EMR_ROUNDRECT          44
2521 #define EMR_ARC               45
2522 #define EMR_CHORD             46
2523 #define EMR_PIE               47
2524 #define EMR_SELECTPALETTE     48
2525 #define EMR_CREATEPALETTE     49
2526 #define EMR_SETPALETTEENTRIES  50
2527 #define EMR_RESIZEPALETTE     51
2528 #define EMR_REALIZEPALETTE     52
2529 #define EMR_EXTFLOODFILL      53
2530 #define EMR_LINETO            54
2531 #define EMR_ARCTO             55
2532 #define EMR_POLYDRAW          56
2533 #define EMR_SETARCDIRECTION    57
2534 #define EMR_SETMITERLIMIT      58
2535 #define EMR_BEGINPATH          59
2536 #define EMR_ENDPATH           60
2537 #define EMR_CLOSEFIGURE       61
2538 #define EMR_FILLPATH           62
2539 #define EMR_STROKEANDFILLPATH  63
2540 #define EMR_STROKEPATH         64
2541 #define EMR_FLATTENPATH        65
2542 #define EMR_WIDENPATH          66
2543 #define EMR_SELECTCLIPPATH     67
2544 #define EMR_ABORTPATH         68
2545 #define EMR_GDI_COMMENT        70
2546 #define EMR_FILLRGN           71
2547 #define EMR_FRAMERGN          72
2548 #define EMR_INVERTRGN          73
2549 #define EMR_PAINTRGN           74
2550 #define EMR_EXTSELECTCLIPRGN   75
2551 #define EMR_BITBLT             76
2552 #define EMR_STRETCHBLT         77
2553 #define EMR_MASKBLT            78
2554 #define EMR_PLGBLT             79
2555 #define EMR_SETDIBITSTODEVICE  80
2556 #define EMR_STRETCHDIBITS      81
2557 #define EMR_EXTCREATEFONTINDIRECTW 82
2558 #define EMR_EXTTEXTOUTA        83
2559 #define EMR_EXTTEXTOUTW        84
2560 #define EMR_POLYBEZIER16       85
2561 #define EMR_POLYGON16          86
2562 #define EMR_POLYLINE16         87
2563 #define EMR_POLYBEZIER16TO16   88
2564 #define EMR_POLYLINETO16       89
2565 #define EMR_POLYPOLYLINE16     90
2566 #define EMR_POLYPOLYGON16      91
2567 #define EMR_POLYDRAW16         92
2568 #define EMR_CREATEMONOBRUSH    93
2569 #define EMR_CREATEDIBPATTERNBRUSHPT 94
2570 #define EMR_EXTCREATEPEN        95
2571 #define EMR_POLYTEXTOUTA        96
2572 #define EMR_POLYTEXTOUTW        97
2573 #define EMR_SETICMMODE          98
2574 #define EMR_CREATECOLORSPACE    99
2575 #define EMRSetColorSPACE        100
2576 #define EMR_DELETECOLORSPACE    101
2577 #define EMR_GLSRECORD           102
2578 #define EMR_GLSBOUNDEDRECORD    103
2579 #define EMR_PIXELFORMAT         104
2580
2581 #define EMR_MIN 1
2582 #define EMR_MAX 104
2583
2584 #define ENHMETA_SIGNATURE 1179469088
```

```
2585 #define ENHMETA_STOCK_OBJECT      0x80000000
2586
2587 #define GDICPMMENT_IDENTIFIER      0x43494447
2588 #define GDICOMMENT_WINDOWS_METAFILE 0x80000000
2589 #define GDICOMMENT_BEGINGROUP      0x80000001
2590 #define GDICOMMENT_ENDGROUP        0x80000002
2591 #define GDICOMMENT_MULTIFORMATS    0x80000003
2592 #define EPS_SIGNATURE               0x46535045
2593
2594 #define CCHDEVICENAME 32
2595 #define CCHFORMNAME 32
2596
2597 typedef struct
2598 {
2599     BYTE    dmDeviceName[CCHDEVICENAME];
2600     WORD    dmSpecVersion;
2601     WORD    dmDriverVersion;
2602     WORD    dmSize;
2603     WORD    dmDriverExtra;
2604     DWORD   dmFields;
2605     union u10 {
2606         struct snort {
2607             SHORT  dmOrientation;
2608             SHORT  dmPaperSize;
2609             SHORT  dmPaperLength;
2610             SHORT  dmPaperWidth;
2611             } DUMMYSTRUCTNAME1;
2612             POINTL dmPosition;
2613         } DUMMYUNIONNAME1;
2614         SHORT  dmScale;
2615         SHORT  dmCopies;
2616         SHORT  dmDefaultSource;
2617         SHORT  dmPrintQuality;
2618         SHORT  dmColor;
2619         SHORT  dmDuplex;
2620         SHORT  dmYResolution;
2621         SHORT  dmTTOption;
2622         SHORT  dmCollate;
2623         BYTE    dmFormName[CCHFORMNAME];
2624         WORD    dmLogPixels;
2625         DWORD   dmBitsPerPel;
2626         DWORD   dmPelsWidth;
2627         DWORD   dmPelsHeight;
2628         DWORD   dmDisplayFlags;
2629         DWORD   dmDisplayFrequency;
2630         DWORD   dmICMMethod;
2631         DWORD   dmICMIntent;
2632         DWORD   dmMediaType;
2633         DWORD   dmDitherType;
2634         DWORD   dmReserved1;
2635         DWORD   dmReserved2;
2636         DWORD   dmPanningWidth;
2637         DWORD   dmPanningHeight;
2638     } DEVMODEA, *PDEVMODEA, *LPDEVMODEA;
2639
2640 typedef struct
2641 {
2642     WCHAR    dmDeviceName[CCHDEVICENAME];
2643     WORD    dmSpecVersion;
2644     WORD    dmDriverVersion;
2645     WORD    dmSize;
2646     WORD    dmDriverExtra;
2647     DWORD   dmFields;
2648     union u20 {
2649         struct blorf {
2650             SHORT  dmOrientation;
2651             SHORT  dmPaperSize;
2652             SHORT  dmPaperLength;
2653             SHORT  dmPaperWidth;
2654             } DUMMYSTRUCTNAME1;
2655             POINTL dmPosition;
2656         } DUMMYUNIONNAME1;
2657         SHORT  dmScale;
2658         SHORT  dmCopies;
2659         SHORT  dmDefaultSource;
2660         SHORT  dmPrintQuality;
2661         SHORT  dmColor;
2662         SHORT  dmDuplex;
2663         SHORT  dmYResolution;
2664         SHORT  dmTTOption;
2665         SHORT  dmCollate;
2666         WCHAR    dmFormName[CCHFORMNAME];
2667         WORD    dmLogPixels;
2668         DWORD   dmBitsPerPel;
2669         DWORD   dmPelsWidth;
2670         DWORD   dmPelsHeight;
2671         DWORD   dmDisplayFlags;
```

```
2672     DWORD    dmDisplayFrequency;
2673     DWORD    dmICMMethod;
2674     DWORD    dmICMIntent;
2675     DWORD    dmMediaType;
2676     DWORD    dmDitherType;
2677     DWORD    dmReserved1;
2678     DWORD    dmReserved2;
2679     DWORD    dmPanningWidth;
2680     DWORD    dmPanningHeight;
2681 } DEVMODEW, *PDEVMODEW, *LPDEVMODEW;
2682
2683 DECL_WINELIB_TYPE_AW(DEVMODE)
2684 DECL_WINELIB_TYPE_AW(PDEVMODE)
2685 DECL_WINELIB_TYPE_AW(LPDEVMODE)
2686
2687 #define DM_SPECVERSION    0x401
2688 #define DM_UPDATE        1
2689 #define DM_COPY          2
2690 #define DM_PROMPT        4
2691 #define DM_MODIFY        8
2692
2693 #define DM_IN_BUFFER      DM_MODIFY
2694 #define DM_IN_PROMPT      DM_PROMPT
2695 #define DM_OUT_BUFFER     DM_COPY
2696 #define DM_OUT_DEFAULT    DM_UPDATE
2697
2698 #define DM_ORIENTATION    0x00000001L
2699 #define DM_PAPERSIZE      0x00000002L
2700 #define DM_PAPERLENGTH   0x00000004L
2701 #define DM_PAPERWIDTH     0x00000008L
2702 #define DM_SCALE          0x00000010L
2703 #define DM_POSITION       0x00000020L
2704 #define DM_COPIES         0x00000100L
2705 #define DM_DEFAULTSOURCE  0x00000200L
2706 #define DM_PRINTQUALITY   0x00000400L
2707 #define DM_COLOR          0x00000800L
2708 #define DM_DUPLEX         0x00001000L
2709 #define DM_YRESOLUTION    0x00002000L
2710 #define DM_TTOPTION       0x00004000L
2711 #define DM_COLLATE        0x00008000L
2712 #define DM_FORMNAME       0x00010000L
2713 #define DM_LOGPIXELS      0x00020000L
2714 #define DM_BITSPERPEL     0x00040000L
2715 #define DM_PELSWIDTH      0x00080000L
2716 #define DM_PELSHEIGHT     0x00100000L
2717 #define DM_DISPLAYFLAGS   0x00200000L
2718 #define DM_DISPLAYFREQUENCY 0x00400000L
2719 #define DM_ICMMETHOD     0x00800000L
2720 #define DM_ICMINTENT      0x01000000L
2721 #define DM_MEDIATYPE      0x02000000L
2722 #define DM_DITHERTYPE     0x04000000L
2723 #define DM_PANNINGWIDTH    0x08000000L
2724 #define DM_PANNINGHEIGHT  0x10000000L
2725
2726 #define DMORIENT_PORTRAIT 1
2727 #define DMORIENT_LANDSCAPE 2
2728
2729 #define DMPAPER_FIRST      DMPAPER_LETTER
2730 #define DMPAPER_LETTER    1
2731 #define DMPAPER_LETTERSMALL 2
2732 #define DMPAPER_TABLOID   3
2733 #define DMPAPER_LEDGER     4
2734 #define DMPAPER_LEGAL      5
2735 #define DMPAPER_STATEMENT  6
2736 #define DMPAPER_EXECUTIVE  7
2737 #define DMPAPER_A3         8
2738 #define DMPAPER_A4         9
2739 #define DMPAPER_A4SMALL    10
2740 #define DMPAPER_A5         11
2741 #define DMPAPER_B4         12
2742 #define DMPAPER_B5         13
2743 #define DMPAPER_FOLIO      14
2744 #define DMPAPER_QUARTO     15
2745 #define DMPAPER_10X14      16
2746 #define DMPAPER_11X17      17
2747 #define DMPAPER_NOTE       18
2748 #define DMPAPER_ENV_9      19
2749 #define DMPAPER_ENV_10     20
2750 #define DMPAPER_ENV_11     21
2751 #define DMPAPER_ENV_12     22
2752 #define DMPAPER_ENV_14     23
2753 #define DMPAPER_CSHEET     24
2754 #define DMPAPER_DSHEET     25
2755 #define DMPAPER_ESHEET     26
2756 #define DMPAPER_ENV_DL      27
2757 #define DMPAPER_ENV_C5      28
2758 #define DMPAPER_ENV_C3      29
```

```
2759 #define DMPAPER_ENV_C4 30
2760 #define DMPAPER_ENV_C6 31
2761 #define DMPAPER_ENV_C65 32
2762 #define DMPAPER_ENV_B4 33
2763 #define DMPAPER_ENV_B5 34
2764 #define DMPAPER_ENV_B6 35
2765 #define DMPAPER_ENV_ITALY 36
2766 #define DMPAPER_ENV_MONARCH 37
2767 #define DMPAPER_ENV_PERSONAL 38
2768 #define DMPAPER_FANFOLD_US 39
2769 #define DMPAPER_FANFOLD_STD_GERMAN 40
2770 #define DMPAPER_FANFOLD_LGL_GERMAN 41
2771 #define DMPAPER_ISO_B4 42
2772 #define DMPAPER_JAPANESE_POSTCARD 43
2773 #define DMPAPER_9X11 44
2774 #define DMPAPER_10X11 45
2775 #define DMPAPER_15X11 46
2776 #define DMPAPER_ENV_INVITE 47
2777 #define DMPAPER_RESERVED_48 48
2778 #define DMPAPER_RESERVED_49 49
2779 #define DMPAPER_LETTER_EXTRA 50
2780 #define DMPAPER_LEGAL_EXTRA 51
2781 #define DMPAPER_TABLOID_EXTRA 52
2782 #define DMPAPER_A4_EXTRA 53
2783 #define DMPAPER_LETTER_TRANSVERSE 54
2784 #define DMPAPER_A4_TRANSVERSE 55
2785 #define DMPAPER_LETTER_EXTRA_TRANSVERSE 56
2786 #define DMPAPER_A_PLUS 57
2787 #define DMPAPER_B_PLUS 58
2788 #define DMPAPER_LETTER_PLUS 59
2789 #define DMPAPER_A4_PLUS 60
2790 #define DMPAPER_A5_TRANSVERSE 61
2791 #define DMPAPER_B5_TRANSVERSE 62
2792 #define DMPAPER_A3_EXTRA 63
2793 #define DMPAPER_A5_EXTRA 64
2794 #define DMPAPER_B5_EXTRA 65
2795 #define DMPAPER_A2 66
2796 #define DMPAPER_A3_TRANSVERSE 67
2797 #define DMPAPER_A3_EXTRA_TRANSVERSE 68
2798 #define DMPAPER_DBL_JAPANESE_POSTCARD 69
2799 #define DMPAPER_A6 70
2800 #define DMPAPER_JENV_KAKU2 71
2801 #define DMPAPER_JENV_KAKU3 72
2802 #define DMPAPER_JENV_CHOU3 73
2803 #define DMPAPER_JENV_CHOU4 74
2804 #define DMPAPER_LETTER_ROTATED 75
2805 #define DMPAPER_A3_ROTATED 76
2806 #define DMPAPER_A4_ROTATED 77
2807 #define DMPAPER_A5_ROTATED 78
2808 #define DMPAPER_B4_JIS_ROTATED 79
2809 #define DMPAPER_B5_JIS_ROTATED 80
2810 #define DMPAPER_JAPANESE_POSTCARD_ROTATED 81
2811 #define DMPAPER_DBL_JAPANESE_POSTCARD_ROTATED 82
2812 #define DMPAPER_A6_ROTATED 83
2813 #define DMPAPER_JENV_KAKU2_ROTATED 84
2814 #define DMPAPER_JENV_KAKU3_ROTATED 85
2815 #define DMPAPER_JENV_CHOU3_ROTATED 86
2816 #define DMPAPER_JENV_CHOU4_ROTATED 87
2817 #define DMPAPER_B6_JIS 88
2818 #define DMPAPER_B6_JIS_ROTATED 89
2819 #define DMPAPER_12X11 90
2820 #define DMPAPER_JENV_YOU4 91
2821 #define DMPAPER_JENV_YOU4_ROTATED 92
2822 #define DMPAPER_P16K 93
2823 #define DMPAPER_P32K 94
2824 #define DMPAPER_P32KBIG 95
2825 #define DMPAPER_PENV_1 96
2826 #define DMPAPER_PENV_2 97
2827 #define DMPAPER_PENV_3 98
2828 #define DMPAPER_PENV_4 99
2829 #define DMPAPER_PENV_5 100
2830 #define DMPAPER_PENV_6 101
2831 #define DMPAPER_PENV_7 102
2832 #define DMPAPER_PENV_8 103
2833 #define DMPAPER_PENV_9 104
2834 #define DMPAPER_PENV_10 105
2835 #define DMPAPER_P16K_ROTATED 106
2836 #define DMPAPER_P32K_ROTATED 107
2837 #define DMPAPER_P32KBIG_ROTATED 108
2838 #define DMPAPER_PENV_1_ROTATED 109
2839 #define DMPAPER_PENV_2_ROTATED 110
2840 #define DMPAPER_PENV_3_ROTATED 111
2841 #define DMPAPER_PENV_4_ROTATED 112
2842 #define DMPAPER_PENV_5_ROTATED 113
2843 #define DMPAPER_PENV_6_ROTATED 114
2844 #define DMPAPER_PENV_7_ROTATED 115
2845 #define DMPAPER_PENV_8_ROTATED 116
```



```

2846 #define DMPAPER_PENV_9_ROTATED      117
2847 #define DMPAPER_PENV_10_ROTATED     118
2848
2849 #define DMPAPER_LAST                  DMPAPER_PENV_10_ROTATED
2850 #define DMPAPER_USER                  256
2851
2852 #define DMBIN_FIRST                   DMBIN_UPPER
2853 #define DMBIN_UPPER                   1
2854 #define DMBIN_ONLYONE                 1
2855 #define DMBIN_LOWER                   2
2856 #define DMBIN_MIDDLE                 3
2857 #define DMBIN_MANUAL                 4
2858 #define DMBIN_ENVELOPE               5
2859 #define DMBIN_ENVMANUAL               6
2860 #define DMBIN_AUTO                   7
2861 #define DMBIN_TRACTOR                8
2862 #define DMBIN_SMALLFMT               9
2863 #define DMBIN_LARGEFORMAT            10
2864 #define DMBIN_LARGECAPACITY          11
2865 #define DMBIN_CASSETTE               14
2866 #define DMBIN_FORMSOURCE             15
2867 #define DMBIN_LAST                   DMBIN_FORMSOURCE
2868 #define DMBIN_USER                   256
2869
2870 #define DMRES_DRAFT                  (-1)
2871 #define DMRES_LOW                    (-2)
2872 #define DMRES_MEDIUM                 (-3)
2873 #define DMRES_HIGH                   (-4)
2874
2875 #define DMCOLOR_MONOCHROME           1
2876 #define DMCOLOR_COLOR                2
2877
2878 #define DMDUP_SIMPLEX                 1
2879 #define DMDUP_VERTICAL                2
2880 #define DMDUP_HORIZONTAL              3
2881
2882 #define DMTT_BITMAP                   1
2883 #define DMTT_DOWNLOAD                 2
2884 #define DMTT_SUBDEV                   3
2885 #define DMTT_DOWNLOAD_OUTLINE         4
2886
2887 #define DMCOLLATE_FALSE                0
2888 #define DMCOLLATE_TRUE                 1
2889
2890 #define DMICMMETHOD_NONE             1
2891 #define DMICMMETHOD_SYSTEM           2
2892 #define DMICMMETHOD_DRIVER           3
2893 #define DMICMMETHOD_DEVICE           4
2894 #define DMICMMETHOD_USER             256
2895
2896 #define DMICM_SATURATE                 1
2897 #define DMICM_CONTRAST                 2
2898 #define DMICM_COLORMETRIC              3
2899 #define DMICM_USER                     256
2900
2901 #define DMMEDIA_STANDARD               1
2902 #define DMMEDIA_TRANSPARENCY           2
2903 #define DMMEDIA_GLOSSY                3
2904 #define DMMEDIA_USER                   256
2905
2906 #define DMDITHER_NONE                 1
2907 #define DMDITHER_COARSE                2
2908 #define DMDITHER_FINE                 3
2909 #define DMDITHER_LINEART              4
2910 #define DMDITHER_GRAYSCALE            5
2911 #define DMDITHER_USER                 256
2912
2913 typedef struct
2914 {
2915     INT      cbSize;
2916     LPCSTR   lpszDocName;
2917     LPCSTR   lpszOutput;
2918     LPCSTR   lpszDatatype;
2919     DWORD    fwType;
2920 } DOCINFOA, *LPDOCINFOA;
2921
2922 typedef struct
2923 {
2924     INT      cbSize;
2925     LPCWSTR  lpszDocName;
2926     LPCWSTR  lpszOutput;
2927     LPCWSTR  lpszDatatype;
2928     DWORD    fwType;
2929 } DOCINFOW, *LPDOCINFOW;
2930
2931 DECL_WINELIB_TYPE_AW(DOCINFO)
2932 DECL_WINELIB_TYPE_AW(LPDOCINFO)

```

```

2933
2934 #define DI_APPBANDING      0x0001
2935
2936 /* Flags for PolyDraw and GetPath */
2937 #define PT_CLOSEFIGURE      0x0001
2938 #define PT_LINETO           0x0002
2939 #define PT_BEZIERTO         0x0004
2940 #define PT_MOVETO           0x0006
2941
2942 #define RDH_RECTANGLES      1
2943
2944 typedef struct _RGNDATAHEADER {
2945     DWORD    dwSize;
2946     DWORD    iType;
2947     DWORD    nCount;
2948     DWORD    nRgnSize;
2949     RECT      rcBound;
2950 } RGNDATAHEADER, *PRGNDATAHEADER;
2951
2952 typedef struct _RGNDATA {
2953     RGNDATAHEADER  rdh;
2954     char            Buffer[1];
2955 } RGNDATA, *PRGNDATA, *LPRGNDATA;
2956
2957 typedef BOOL CALLBACK (*ABORTPROC) (HDC, INT);
2958
2959 typedef struct {
2960     DWORD    cb;
2961     CHAR      DeviceName[32];
2962     CHAR      DeviceString[128];
2963     DWORD    StateFlags;
2964     CHAR      DeviceID[128];
2965     CHAR      DeviceKey[128];
2966 } DISPLAY_DEVICEA, *PDISPLAY_DEVICEA, *LPDISPLAY_DEVICEA;
2967
2968 typedef struct {
2969     DWORD    cb;
2970     WCHAR     DeviceName[32];
2971     WCHAR     DeviceString[128];
2972     DWORD    StateFlags;
2973     WCHAR     DeviceID[128];
2974     WCHAR     DeviceKey[128];
2975 } DISPLAY_DEVICEW, *PDISPLAY_DEVICEW, *LPDISPLAY_DEVICEW;
2976 DECL_WINELIB_TYPE_AW(DISPLAY_DEVICE)
2977 DECL_WINELIB_TYPE_AW(PDISPLAY_DEVICE)
2978 DECL_WINELIB_TYPE_AW(LPDISPLAY_DEVICE)
2979
2980 /* DISPLAY_DEVICE.StateFlags (?) */
2981 #define DISPLAY_DEVICE_ATTACHED_TO_DESKTOP  0x00000001
2982 #define DISPLAY_DEVICE_MULTI_DRIVER         0x00000002
2983 #define DISPLAY_DEVICE_PRIMARY_DEVICE       0x00000004
2984 #define DISPLAY_DEVICE_MIRRORING_DRIVER     0x00000008
2985 #define DISPLAY_DEVICE_VGA_COMPATIBLE       0x00000010
2986
2987 #define GDI_ERROR                           (0xFFFFFFFFL)
2988 #define HGDI_ERROR                          ((HANDLE) 0xFFFFFFFFL)
2989
2990 INT      WINAPI AbortDoc(HDC);
2991 BOOL     WINAPI AbortPath(HDC);
2992 INT      WINAPI AddFontResourceA(LPCSTR);
2993 INT      WINAPI AddFontResourceW(LPCWSTR);
2994 #define    AddFontResource WINELIB_NAME_AW(AddFontResource)
2995 BOOL     WINAPI AngleArc(HDC, INT, INT, DWORD, FLOAT, FLOAT);
2996 BOOL     WINAPI AnimatePalette(HPALETTE, UINT, UINT, const PALETTEENTRY*);
2997 BOOL     WINAPI Arc(HDC, INT, INT, INT, INT, INT, INT, INT);
2998 BOOL     WINAPI ArcTo(HDC, INT, INT, INT, INT, INT, INT, INT, INT);
2999 BOOL     WINAPI BeginPath(HDC);
3000 BOOL     WINAPI BitBlt(HDC, INT, INT, INT, INT, HDC, INT, INT, DWORD);
3001 INT      WINAPI ChoosePixelFormat(HDC, const LPPIXELFORMATDESCRIPTOR);
3002 BOOL     WINAPI Chord(HDC, INT, INT, INT, INT, INT, INT, INT);
3003 HENHMETAFILE WINAPI CloseEnhMetaFile(HDC);
3004 BOOL     WINAPI CloseFigure(HDC);
3005 HMETAFILE WINAPI CloseMetaFile(HDC);
3006 INT      WINAPI CombineRgn(HRGN, HRGN, HRGN, INT);
3007 BOOL     WINAPI CombineTransform(LPXFORM, const XFORM *, const XFORM *);
3008 HENHMETAFILE WINAPI CopyEnhMetaFileA(HENHMETAFILE, LPCSTR);
3009 HENHMETAFILE WINAPI CopyEnhMetaFileW(HENHMETAFILE, LPCWSTR);
3010 #define    CopyEnhMetaFile WINELIB_NAME_AW(CopyEnhMetaFile)
3011 HMETAFILE WINAPI CopyMetaFileA(HMETAFILE, LPCSTR);
3012 HMETAFILE WINAPI CopyMetaFileW(HMETAFILE, LPCWSTR);
3013 #define    CopyMetaFile WINELIB_NAME_AW(CopyMetaFile)
3014 HBITMAP  WINAPI CreateBitmap(INT, INT, UINT, UINT, LPCVOID);
3015 HBITMAP  WINAPI CreateBitmapIndirect(const BITMAP*);
3016 HBRUSH   WINAPI CreateBrushIndirect(const LOGBRUSH*);
3017 HCOLORSPACE WINAPI CreateColorSpaceA(LPLOGCOLORSPACEA);
3018 HCOLORSPACE WINAPI CreateColorSpaceW(LPLOGCOLORSPACEW);
3019 #define    CreateColorSpace WINELIB_NAME_AW(CreateColorSpace)

```

```

3020 HBITMAP    WINAPI CreateCompatibleBitmap(HDC, INT, INT);
3021 HDC        WINAPI CreateCompatibleDC(HDC);
3022 HDC        WINAPI CreateDCA(LPCSTR, LPCSTR, LPCSTR, const DEVMODEA*);
3023 HDC        WINAPI CreateDCW(LPCWSTR, LPCWSTR, LPCWSTR, const DEVMODEW*);
3024 #define      CreateDC WINELIB_NAME_AW(CreateDC)
3025 HBITMAP    WINAPI CreateDIBitmap(HDC, const BITMAPINFOHEADER*, DWORD,
3026                                  LPVOID, const BITMAPINFO*, UINT);
3027 HBRUSH      WINAPI CreateDIBPatternBrush(HGLOBAL, UINT);
3028 HBRUSH      WINAPI CreateDIBPatternBrushPt(const void*, UINT);
3029 HBITMAP    WINAPI CreateDIBSection(HDC, BITMAPINFO*, UINT,
3030                                   LPVOID*, HANDLE, DWORD offset);
3031 HBITMAP    WINAPI CreateDiscardableBitmap(HDC, INT, INT);
3032 HRGN        WINAPI CreateEllipticRgn(INT, INT, INT, INT);
3033 HRGN        WINAPI CreateEllipticRgnIndirect(const RECT*);
3034 HDC         WINAPI CreateEnhMetaFileA(HDC, LPCSTR, const RECT*, LPCSTR);
3035 HDC         WINAPI CreateEnhMetaFileW(HDC, LPCWSTR, const RECT*, LPCWSTR);
3036 #define      CreateEnhMetaFile WINELIB_NAME_AW(CreateEnhMetaFile)
3037 HFONT       WINAPI CreateFontA(INT, INT, INT, INT, INT, DWORD, DWORD,
3038                                DWORD, DWORD, DWORD, DWORD, LPCSTR);
3039 HFONT       WINAPI CreateFontW(INT, INT, INT, INT, INT, DWORD, DWORD,
3040                                DWORD, DWORD, DWORD, DWORD, LPCWSTR);
3041 #define      CreateFont WINELIB_NAME_AW(CreateFont)
3042 HFONT       WINAPI CreateFontIndirectA(const LOGFONTA*);
3043 HFONT       WINAPI CreateFontIndirectW(const LOGFONTW*);
3044 #define      CreateFontIndirect WINELIB_NAME_AW(CreateFontIndirect)
3045 HPALETTE    WINAPI CreateHalftonePalette(HDC);
3046 HBRUSH      WINAPI CreateHatchBrush(INT, COLORREF);
3047 HDC         WINAPI CreateICA(LPCSTR, LPCSTR, LPCSTR, const DEVMODEA*);
3048 HDC         WINAPI CreateICW(LPCWSTR, LPCWSTR, LPCWSTR, const DEVMODEW*);
3049 #define      CreateIC WINELIB_NAME_AW(CreateIC)
3050 HDC         WINAPI CreateMetaFileA(LPCSTR);
3051 HDC         WINAPI CreateMetaFileW(LPCWSTR);
3052 #define      CreateMetaFile WINELIB_NAME_AW(CreateMetaFile)
3053 HPALETTE    WINAPI CreatePalette(const LOGPALETTE*);
3054 HBRUSH      WINAPI CreatePatternBrush(HBITMAP);
3055 HPEN        WINAPI CreatePen(INT, INT, COLORREF);
3056 HPEN        WINAPI CreatePenIndirect(const LOGPEN*);
3057 HRGN        WINAPI CreatePolyPolygonRgn(const POINT*, const INT*, INT, INT);
3058 HRGN        WINAPI CreatePolygonRgn(const POINT*, INT, INT);
3059 HRGN        WINAPI CreateRectRgn(INT, INT, INT, INT);
3060 HRGN        WINAPI CreateRectRgnIndirect(const RECT*);
3061 HRGN        WINAPI CreateRoundRectRgn(INT, INT, INT, INT, INT, INT);
3062 BOOL        WINAPI CreateScalableFontResourceA(DWORD, LPCSTR, LPCSTR, LPCSTR);
3063 BOOL        WINAPI CreateScalableFontResourceW(DWORD, LPCWSTR, LPCWSTR, LPCWSTR);
3064 #define      CreateScalableFontResource WINELIB_NAME_AW(CreateScalableFontResource)
3065 HBRUSH      WINAPI CreateSolidBrush(COLORREF);
3066 BOOL        WINAPI DPToLP(HDC, LPPOINT, INT);
3067 BOOL        WINAPI DeleteColorSpace(HCOLORSPACE);
3068 BOOL        WINAPI DeleteDC(HDC);
3069 BOOL        WINAPI DeleteEnhMetaFile(HENHMETAFILE);
3070 BOOL        WINAPI DeleteMetaFile(HMETAFILE);
3071 BOOL        WINAPI DeleteObject(HGDIOBJ);
3072 INT         WINAPI DescribePixelFormat(HDC, int, UINT,
3073                                       LPPIXELFORMATDESCRIPTOR);
3074 INT         WINAPI DrawEscape(HDC, INT, INT, LPCSTR);
3075 BOOL        WINAPI Ellipse(HDC, INT, INT, INT, INT);
3076 INT         WINAPI EndDoc(HDC);
3077 BOOL        WINAPI EndPath(HDC);
3078 BOOL        WINAPI EnumEnhMetaFile(HDC, HENHMETAFILE, ENHMFENUMPROC, LPVOID, const RECT*);
3079 INT         WINAPI EnumFontFamiliesA(HDC, LPCSTR, FONTENUMPROCA, LPARAM);
3080 INT         WINAPI EnumFontFamiliesW(HDC, LPCWSTR, FONTENUMPROCW, LPARAM);
3081 #define      EnumFontFamilies WINELIB_NAME_AW(EnumFontFamilies)
3082 INT         WINAPI EnumFontFamiliesExA(HDC, LPLOGFONTA, FONTENUMPROCEXA, LPARAM, DWORD);
3083 INT         WINAPI EnumFontFamiliesExW(HDC, LPLOGFONTW, FONTENUMPROCEXW, LPARAM, DWORD);
3084 #define      EnumFontFamiliesEx WINELIB_NAME_AW(EnumFontFamiliesEx)
3085 INT         WINAPI EnumFontsWithA(HDC, LPCSTR, FONTENUMPROCA, LPARAM);
3086 INT         WINAPI EnumFontsWithW(HDC, LPCWSTR, FONTENUMPROCW, LPARAM);
3087 #define      EnumFonts WINELIB_NAME_AW(EnumFonts)
3088 BOOL        WINAPI EnumMetaFile(HDC, HMETAFILE, MFENUMPROC, LPARAM);
3089 INT         WINAPI EnumObjects(HDC, INT, GOBJENUMPROC, LPARAM);
3090 BOOL        WINAPI EqualRgn(HRGN, HRGN);
3091 INT         WINAPI Escape(HDC, INT, INT, LPCSTR, LPVOID);
3092 INT         WINAPI ExcludeClipRect(HDC, INT, INT, INT, INT);
3093 HPEN        WINAPI ExtCreatePen(DWORD, DWORD, const LOGBRUSH*, DWORD, const DWORD*);
3094 HRGN        WINAPI ExtCreateRegion(const XFORM*, DWORD, const RGNDATA*);
3095 INT         WINAPI ExtEscape(HDC, INT, INT, LPCSTR, INT, LPSTR);
3096 BOOL        WINAPI ExtFloodFill(HDC, INT, INT, COLORREF, UINT);
3097 INT         WINAPI ExtSelectClipRgn(HDC, HRGN, INT);
3098 BOOL        WINAPI ExtTextOutA(HDC, INT, INT, UINT, const RECT*,
3099                                LPCSTR, UINT, const INT*);
3100 BOOL        WINAPI ExtTextOutW(HDC, INT, INT, UINT, const RECT*,
3101                                LPCWSTR, UINT, const INT*);
3102 #define      ExtTextOut WINELIB_NAME_AW(ExtTextOut)
3103 BOOL        WINAPI FillPath(HDC);
3104 BOOL        WINAPI FillRgn(HDC, HRGN, HBRUSH);
3105 BOOL        WINAPI FixBrushOrgEx(HDC, INT, INT, LPPOINT);
3106 BOOL        WINAPI FlattenPath(HDC);

```

```

3107 BOOL        WINAPI FloodFill(HDC, INT, INT, COLORREF);
3108 BOOL        WINAPI FrameRgn(HDC, HRGN, HBRUSH, INT, INT);
3109 BOOL        WINAPI GdiComment(HDC, UINT, const BYTE *);
3110 BOOL        WINAPI GdiFlush(void);
3111 INT         WINAPI GetArcDirection(HDC);
3112 BOOL        WINAPI GetAspectRatioFilterEx(HDC, LPCTSTR);
3113 LONG        WINAPI GetBitmapBits(HBITMAP, LONG, LPVOID);
3114 BOOL        WINAPI GetBitmapDimensionEx(HBITMAP, LPCTSTR);
3115 BOOL        WINAPI GetBrushOrgEx(HDC, LPPOINT);
3116 COLORREF    WINAPI GetBkColor(HDC);
3117 INT         WINAPI GetBkMode(HDC);
3118 UINT        WINAPI GetBoundsRect(HDC, LPRECT, UINT);
3119 BOOL        WINAPI GetCharABCWidthsA(HDC, UINT, UINT, LPABC);
3120 BOOL        WINAPI GetCharABCWidthsW(HDC, UINT, UINT, LPABC);
3121 #define      GetCharABCWidths WINELIB_NAME_AW(GetCharABCWidths)
3122 BOOL        WINAPI GetCharABCWidthsFloatA(HDC, UINT, UINT, LPABCFLOAT);
3123 BOOL        WINAPI GetCharABCWidthsFloatW(HDC, UINT, UINT, LPABCFLOAT);
3124 #define      GetCharABCWidthsFloat WINELIB_NAME_AW(GetCharABCWidthsFloat)
3125 DWORD       WINAPI GetCharacterPlacementA(HDC, LPCSTR, INT, INT, GCP_RESULTS*, DWORD);
3126 DWORD       WINAPI GetCharacterPlacementW(HDC, LPCWSTR, INT, INT, GCP_RESULTS*, DWORD);
3127 #define      GetCharacterPlacement WINELIB_NAME_AW(GetCharacterPlacement)
3128 BOOL        WINAPI GetCharWidth32A(HDC, UINT, UINT, LPINT);
3129 BOOL        WINAPI GetCharWidth32W(HDC, UINT, UINT, LPINT);
3130 #define      GetCharWidthA GetCharWidth32A
3131 #define      GetCharWidthW GetCharWidth32W
3132 #define      GetCharWidth32 WINELIB_NAME_AW(GetCharWidth32)
3133 #define      GetCharWidth WINELIB_NAME_AW(GetCharWidth)
3134 BOOL        WINAPI GetCharWidthFloatA(HDC, UINT, UINT, PFLOAT);
3135 BOOL        WINAPI GetCharWidthFloatW(HDC, UINT, UINT, PFLOAT);
3136 #define      GetCharWidthFloat WINELIB_NAME_AW(GetCharWidthFloat)
3137 INT         WINAPI GetClipBox(HDC, LPRECT);
3138 INT         WINAPI GetClipRgn(HDC, HRGN);
3139 BOOL        WINAPI GetColorAdjustment(HDC, LPCOLORADJUSTMENT);
3140 HANDLE      WINAPI GetCurrentObject(HDC, UINT);
3141 BOOL        WINAPI GetCurrentPositionEx(HDC, LPPOINT);
3142 INT         WINAPI GetDeviceCaps(HDC, INT);
3143 BOOL        WINAPI GetDeviceGammaRamp(HDC, LPVOID);
3144 COLORREF    WINAPI GetDCBrushColor(HDC);
3145 BOOL        WINAPI GetDCOrgEx(HDC, LPPOINT);
3146 COLORREF    WINAPI GetDCPenColor(HDC);
3147 UINT        WINAPI GetDIBColorTable(HDC, UINT, UINT, RGBQUAD*);
3148 INT         WINAPI GetDIBits(HDC, HBITMAP, UINT, UINT, LPVOID, LPBITMAPINFO, UINT);
3149 HENHMETAFILE WINAPI GetEnhMetaFileA(LPCSTR);
3150 HENHMETAFILE WINAPI GetEnhMetaFileW(LPCWSTR);
3151 #define      GetEnhMetaFile WINELIB_NAME_AW(GetEnhMetaFile)
3152 UINT        WINAPI GetEnhMetaFileBits(HENHMETAFILE, UINT, LPBYTE);
3153 UINT        WINAPI GetEnhMetaFileDescriptionA(HENHMETAFILE, UINT, LPSTR);
3154 UINT        WINAPI GetEnhMetaFileDescriptionW(HENHMETAFILE, UINT, LPWSTR);
3155 #define      GetEnhMetaFileDescription WINELIB_NAME_AW(GetEnhMetaFileDescription)
3156 UINT        WINAPI GetEnhMetaFileHeader(HENHMETAFILE, UINT, LPENHMETAHEADER);
3157 UINT        WINAPI GetEnhMetaFilePaletteEntries(HENHMETAFILE, UINT, LPPALETTEENTRY);
3158 DWORD       WINAPI GetFontData(HDC, DWORD, DWORD, LPVOID, DWORD);
3159 DWORD       WINAPI GetFontLanguageInfo(HDC);
3160 DWORD       WINAPI GetGlyphOutlineA(HDC, UINT, UINT, LPGLYPHMETRICS, DWORD, LPVOID, const MAT2*);
3161 DWORD       WINAPI GetGlyphOutlineW(HDC, UINT, UINT, LPGLYPHMETRICS, DWORD, LPVOID, const MAT2*);
3162 #define      GetGlyphOutline WINELIB_NAME_AW(GetGlyphOutline)
3163 INT         WINAPI GetGraphicsMode(HDC);
3164 DWORD       WINAPI GetKerningPairsA(HDC, DWORD, LPKERNINGPAIR);
3165 DWORD       WINAPI GetKerningPairsW(HDC, DWORD, LPKERNINGPAIR);
3166 #define      GetKerningPairs WINELIB_NAME_AW(GetKerningPairs)
3167 DWORD       WINAPI GetLayout(HDC);
3168 BOOL        WINAPI GetLogColorSpaceA(HCOLORSPACE, LPLOGCOLORSPACEA, DWORD);
3169 BOOL        WINAPI GetLogColorSpaceW(HCOLORSPACE, LPLOGCOLORSPACEW, DWORD);
3170 #define      GetLogColorSpace WINELIB_NAME_AW(GetLogColorSpace)
3171 INT         WINAPI GetMapMode(HDC);
3172 HMETAFILE   WINAPI GetMetaFileA(LPCSTR);
3173 HMETAFILE   WINAPI GetMetaFileW(LPCWSTR);
3174 #define      GetMetaFile WINELIB_NAME_AW(GetMetaFile)
3175 UINT        WINAPI GetMetaFileBitsEx(HMETAFILE, UINT, LPVOID);
3176 INT         WINAPI GetMetaRgn(HDC, HRGN);
3177 BOOL        WINAPI GetMiterLimit(HDC, PFLOAT);
3178 DWORD       WINAPI GetNearestColor(HDC, DWORD);
3179 UINT        WINAPI GetNearestPaletteIndex(HPALETTE, COLORREF);
3180 INT         WINAPI GetObjectA(HANDLE, INT, LPVOID);
3181 INT         WINAPI GetObjectW(HANDLE, INT, LPVOID);
3182 #define      GetObject WINELIB_NAME_AW(GetObject)
3183 DWORD       WINAPI GetObjectType(HANDLE);
3184 UINT        WINAPI GetOutlineTextMetricsA(HDC, UINT, LPOUTLINETEXTMETRICA);
3185 UINT        WINAPI GetOutlineTextMetricsW(HDC, UINT, LPOUTLINETEXTMETRICW);
3186 #define      GetOutlineTextMetrics WINELIB_NAME_AW(GetOutlineTextMetrics)
3187 UINT        WINAPI GetPaletteEntries(HPALETTE, UINT, LPPALETTEENTRY);
3188 INT         WINAPI GetPath(HDC, LPPOINT, LPBYTE, INT);
3189 COLORREF    WINAPI GetPixel(HDC, INT, INT);
3190 INT         WINAPI GetPixelFormat(HDC);
3191 INT         WINAPI GetPolyFillMode(HDC);
3192 BOOL        WINAPI GetRasterizerCaps(LPRASTERIZER_STATUS, UINT);
3193 DWORD       WINAPI GetRegionData(HRGN, DWORD, LPRGNDATA);

```

```

3194 INT      WINAPI GetRelAbs (HDC, DWORD);
3195 INT      WINAPI GetRgnBox (HRGN, LPRECT);
3196 INT      WINAPI GetROP2 (HDC);
3197 HGDIOBJ   WINAPI GetStockObject (INT);
3198 INT      WINAPI GetStretchBltMode (HDC);
3199 UINT      WINAPI GetSystemPaletteEntries (HDC, UINT, UINT, LPPALETTEENTRY);
3200 UINT      WINAPI GetSystemPaletteUse (HDC);
3201 UINT      WINAPI GetTextAlign (HDC);
3202 INT      WINAPI GetTextCharacterExtra (HDC);
3203 UINT      WINAPI GetTextCharset (HDC);
3204 UINT      WINAPI GetTextCharsetInfo (HDC, LPFONTSIGNATURE, DWORD);
3205 COLORREF  WINAPI GetTextColor (HDC);
3206 BOOL      WINAPI GetTextExtentExPointA (HDC, LPCSTR, INT, INT,
3207                                          LPINT, LPINT, LPSIZE);
3208 BOOL      WINAPI GetTextExtentExPointW (HDC, LPCWSTR, INT, INT,
3209                                          LPINT, LPINT, LPSIZE);
3210 BOOL      WINAPI GetTextExtentPointA (HDC, LPCSTR, INT, LPSIZE);
3211 BOOL      WINAPI GetTextExtentPointW (HDC, LPCWSTR, INT, LPSIZE);
3212 #define    GetTextExtentPoint WINELIB_NAME_AW(GetTextExtentPoint)
3213 BOOL      WINAPI GetTextExtentPoint32A (HDC, LPCSTR, INT, LPSIZE);
3214 BOOL      WINAPI GetTextExtentPoint32W (HDC, LPCWSTR, INT, LPSIZE);
3215 #define    GetTextExtentPoint32 WINELIB_NAME_AW(GetTextExtentPoint32)
3216 #define    GetTextExtentExPoint WINELIB_NAME_AW(GetTextExtentExPoint)
3217 INT      WINAPI GetTextFaceA (HDC, INT, LPSTR);
3218 INT      WINAPI GetTextFaceW (HDC, INT, LPWSTR);
3219 #define    GetTextFace WINELIB_NAME_AW(GetTextFace)
3220 BOOL      WINAPI GetTextMetricsA (HDC, LPTEXTMETRICA);
3221 BOOL      WINAPI GetTextMetricsW (HDC, LPTEXTMETRICW);
3222 #define    GetTextMetrics WINELIB_NAME_AW(GetTextMetrics)
3223 BOOL      WINAPI GetViewportExtEx (HDC, LPSIZE);
3224 BOOL      WINAPI GetViewportOrgEx (HDC, LPPOINT);
3225 BOOL      WINAPI GetWindowExtEx (HDC, LPSIZE);
3226 BOOL      WINAPI GetWindowOrgEx (HDC, LPPOINT);
3227 BOOL      WINAPI GetWorldTransform (HDC, LPXFORM);
3228 INT      WINAPI IntersectClipRect (HDC, INT, INT, INT, INT);
3229 BOOL      WINAPI InvertRgn (HDC, HRGN);
3230 BOOL      WINAPI LineDDA (INT, INT, INT, INT, LINEDDAPROC, LPARAM);
3231 BOOL      WINAPI LineTo (HDC, INT, INT);
3232 BOOL      WINAPI LPtoDP (HDC, LPPOINT, INT);
3233 BOOL      WINAPI MaskBlt (HDC, INT, INT, INT, INT, HDC, INT, INT, HBITMAP, INT, INT, DWORD);
3234 BOOL      WINAPI ModifyWorldTransform (HDC, const XFORM *, DWORD);
3235 BOOL      WINAPI MoveToEx (HDC, INT, INT, LPPOINT);
3236 /* FIXME This is defined in kernel32.spec !?*/
3237 INT      WINAPI MulDiv (INT, INT, INT);
3238 INT      WINAPI OffsetClipRgn (HDC, INT, INT);
3239 INT      WINAPI OffsetRgn (HRGN, INT, INT);
3240 BOOL      WINAPI OffsetViewportOrgEx (HDC, INT, INT, LPPOINT);
3241 BOOL      WINAPI OffsetWindowOrgEx (HDC, INT, INT, LPPOINT);
3242 BOOL      WINAPI PaintRgn (HDC, HRGN);
3243 BOOL      WINAPI PatBlt (HDC, INT, INT, INT, INT, DWORD);
3244 HRGN      WINAPI PathToRegion (HDC);
3245 BOOL      WINAPI Pie (HDC, INT, INT, INT, INT, INT, INT, INT, INT);
3246 BOOL      WINAPI PlayEnhMetaFile (HDC, HENHMETAFILE, const RECT*);
3247 BOOL      WINAPI PlayEnhMetaFileRecord (HDC, LPHANDLETABLE, const ENHMETARECORD*, UINT);
3248 BOOL      WINAPI PlayMetaFile (HDC, HMETAFILE);
3249 BOOL      WINAPI PlayMetaFileRecord (HDC, LPHANDLETABLE, LPMETARECORD, UINT);
3250 BOOL      WINAPI PlgBlt (HDC, const POINT*, HDC, INT, INT, INT, INT, HBITMAP, INT, INT);
3251 BOOL      WINAPI PolyBezier (HDC, const POINT*, DWORD);
3252 BOOL      WINAPI PolyBezierTo (HDC, const POINT*, DWORD);
3253 BOOL      WINAPI PolyDraw (HDC, const POINT*, const BYTE*, DWORD);
3254 BOOL      WINAPI PolyPolygon (HDC, const POINT*, const INT*, UINT);
3255 BOOL      WINAPI PolyPolyline (HDC, const POINT*, const DWORD*, DWORD);
3256 BOOL      WINAPI Polygon (HDC, const POINT*, INT);
3257 BOOL      WINAPI Polyline (HDC, const POINT*, INT);
3258 BOOL      WINAPI PolylineTo (HDC, const POINT*, DWORD);
3259 BOOL      WINAPI PtInRegion (HRGN, INT, INT);
3260 BOOL      WINAPI PtVisible (HDC, INT, INT);
3261 UINT      WINAPI RealizePalette (HDC);
3262 BOOL      WINAPI Rectangle (HDC, INT, INT, INT, INT);
3263 BOOL      WINAPI RectInRegion (HRGN, const RECT *);
3264 BOOL      WINAPI RectVisible (HDC, const RECT*);
3265 BOOL      WINAPI RemoveFontResourceA (LPCSTR);
3266 BOOL      WINAPI RemoveFontResourceW (LPCWSTR);
3267 #define    RemoveFontResource WINELIB_NAME_AW(RemoveFontResource)
3268 HDC      WINAPI ResetDCA (HDC, const DEVMODEA *);
3269 HDC      WINAPI ResetDCW (HDC, const DEVMODEW *);
3270 #define    ResetDC WINELIB_NAME_AW(ResetDC)
3271 BOOL      WINAPI ResizePalette (HPALETTE, UINT);
3272 BOOL      WINAPI RestoreDC (HDC, INT);
3273 BOOL      WINAPI RoundRect (HDC, INT, INT, INT, INT, INT, INT);
3274 INT      WINAPI SaveDC (HDC);
3275 BOOL      WINAPI ScaleViewportExtEx (HDC, INT, INT, INT, INT, LPSIZE);
3276 BOOL      WINAPI ScaleWindowExtEx (HDC, INT, INT, INT, INT, LPSIZE);
3277 BOOL      WINAPI SelectClipPath (HDC, INT);
3278 INT      WINAPI SelectClipRgn (HDC, HRGN);
3279 HGDIOBJ   WINAPI SelectObject (HDC, HGDIOBJ);
3280 HPALETTE  WINAPI SelectPalette (HDC, HPALETTE, BOOL);

```

```

3281 INT      WINAPI SetAbortProc(HDC, ABORTPROC);
3282 INT      WINAPI SetArcDirection(HDC, INT);
3283 LONG      WINAPI SetBitmapBits(HBITMAP, LONG, LPCVOID);
3284 BOOL      WINAPI SetBitmapDimensionEx(HBITMAP, INT, INT, LPCTSTR);
3285 COLORREF  WINAPI SetBkColor(HDC, COLORREF);
3286 INT      WINAPI SetBkMode(HDC, INT);
3287 UINT      WINAPI SetBoundsRect(HDC, const RECT*, UINT);
3288 BOOL      WINAPI SetBrushOrgEx(HDC, INT, INT, LPPOINT);
3289 BOOL      WINAPI SetColorAdjustment(HDC, const COLORADJUSTMENT*);
3290 HCOLORSPACE WINAPI SetColorSpace(HDC, HCOLORSPACE);
3291 BOOL      WINAPI SetDeviceGammaRamp(HDC, LPVOID);
3292 UINT      WINAPI SetDIBColorTable(HDC, UINT, UINT, RGBQUAD*);
3293 INT      WINAPI SetDIBits(HDC, HBITMAP, UINT, UINT, LPCVOID, const BITMAPINFO*, UINT);
3294 INT      WINAPI SetDIBitsToDevice(HDC, INT, INT, DWORD, DWORD, INT,
3295 INT, UINT, UINT, LPCVOID, const BITMAPINFO*, UINT);
3296 HENHMETAFILE WINAPI SetEnhMetaFileBits(UINT, const BYTE *);
3297 INT      WINAPI SetGraphicsMode(HDC, INT);
3298 INT      WINAPI SetICMMode(HDC, INT);
3299 DWORD     WINAPI SetLayout(HDC, DWORD);
3300 INT      WINAPI SetMapMode(HDC, INT);
3301 DWORD     WINAPI SetMapperFlags(HDC, DWORD);
3302 HMETAFILE  WINAPI SetMetaFileBitsEx(UINT, const BYTE*);
3303 INT      WINAPI SetMetaRgn(HDC);
3304 BOOL      WINAPI SetMiterLimit(HDC, FLOAT, PFLOAT);
3305 UINT      WINAPI SetPaletteEntries(HPALETTE, UINT, LPPALETTEENTRY);
3306 COLORREF  WINAPI SetPixel(HDC, INT, INT, COLORREF);
3307 BOOL      WINAPI SetPixelV(HDC, INT, INT, COLORREF);
3308 BOOL      WINAPI SetPixelFormat(HDC, int, const PIXELFORMATDESCRIPTOR*);
3309 INT      WINAPI SetPolyFillMode(HDC, INT);
3310 BOOL      WINAPI SetRectRgn(HRGN, INT, INT, INT, INT);
3311 INT      WINAPI SetRelAbs(HDC, INT);
3312 INT      WINAPI SetROP2(HDC, INT);
3313 INT      WINAPI SetStretchBltMode(HDC, INT);
3314 UINT      WINAPI SetSystemPaletteUse(HDC, UINT);
3315 UINT      WINAPI SetTextAlign(HDC, UINT);
3316 INT      WINAPI SetTextCharacterExtra(HDC, INT);
3317 COLORREF  WINAPI SetTextColor(HDC, COLORREF);
3318 BOOL      WINAPI SetTextJustification(HDC, INT, INT);
3319 BOOL      WINAPI SetViewportExtEx(HDC, INT, INT, LPCTSTR);
3320 BOOL      WINAPI SetViewportOrgEx(HDC, INT, INT, LPPOINT);
3321 BOOL      WINAPI SetWindowExtEx(HDC, INT, INT, LPCTSTR);
3322 BOOL      WINAPI SetWindowOrgEx(HDC, INT, INT, LPPOINT);
3323 HENHMETAFILE WINAPI SetWinMetaFileBits(UINT, CONST BYTE*, HDC, CONST METAFILEPICT *);
3324 BOOL      WINAPI SetWorldTransform(HDC, const XFORM*);
3325 INT      WINAPI StartDocA(HDC, const DOCINFOA*);
3326 INT      WINAPI StartDocW(HDC, const DOCINFOW*);
3327 #define StartDoc WINELIB_NAME_AW(StartDoc)
3328 INT      WINAPI StartPage(HDC);
3329 INT      WINAPI EndPage(HDC);
3330 BOOL      WINAPI StretchBlt(HDC, INT, INT, INT, INT, HDC, INT,
3331 INT, INT, INT, DWORD);
3332 INT      WINAPI StretchDIBits(HDC, INT, INT, INT, INT, INT, INT,
3333 INT, INT, const VOID*, const BITMAPINFO*, UINT, DWORD);
3334 BOOL      WINAPI StrokeAndFillPath(HDC);
3335 BOOL      WINAPI StrokePath(HDC);
3336 BOOL      WINAPI SwapBuffers(HDC);
3337 BOOL      WINAPI TextOutA(HDC, INT, INT, LPCSTR, INT);
3338 BOOL      WINAPI TextOutW(HDC, INT, INT, LPCWSTR, INT);
3339 #define TextOut WINELIB_NAME_AW(TextOut)
3340 BOOL      WINAPI TranslateCharsetInfo(LPDWORD, LPCCHARSETINFO, DWORD);
3341 BOOL      WINAPI UnrealizeObject(HGDIOBJ);
3342 BOOL      WINAPI UpdateColors(HDC);
3343 BOOL      WINAPI WidenPath(HDC);
3344 BOOL      WINAPI PolyTextOutA(HDC, PPOLYTEXTA, INT);
3345 BOOL      WINAPI PolyTextOutW(HDC, PPOLYTEXTW, INT);
3346 #define PolyTextOut WINELIB_NAME_AW(PolyTextOut)
3347
3348 #ifdef __cplusplus
3349 }
3350 #endif
3351
3352 #endif /* !NOGDI */
3353 #endif /* _WINGDI_ */

```

5.12 winnt.h

```

1 /*
2  * Win32 definitions for Windows NT
3  *
4  * Copyright 1996 Alexandre Julliard
5  */
6
7 #ifndef __WINE_WINNT_H
8 #define __WINE_WINNT_H

```

```

9
10 #include "basetsd.h"
11
12 #ifndef RC_INVOKED
13 #include <ctype.h>
14 #include <stddef.h>
15 #include <string.h>
16 #endif
17
18
19 /* On Windows winnt.h depends on a few windef.h types and macros and thus
20 * is not self-contained. Furthermore windef.h includes winnt.h so that it
21 * would be pointless to try to use winnt.h directly.
22 * But for Wine and Winelib I decided to make winnt.h self-contained by
23 * moving these definitions to winnt.h. It makes no difference to Winelib
24 * programs since they are not using winnt.h directly anyway, and it allows
25 * us to use winnt.h and get a minimal set of definitions.
26 */
27
28 /**** Some Wine specific definitions *****/
29
30 /* Architecture dependent settings. */
31 /* These are hardcoded to avoid dependencies on config.h in Winelib apps. */
32 #if defined(__i386__)
33 # undef WORDS_BIGENDIAN
34 # undef BITFIELDS_BIGENDIAN
35 # define ALLOW_UNALIGNED_ACCESS
36 #elif defined(__x86_64__)
37 # undef WORDS_BIGENDIAN
38 # undef BITFIELDS_BIGENDIAN
39 # define ALLOW_UNALIGNED_ACCESS
40 #elif defined(__alpha__)
41 # undef WORDS_BIGENDIAN
42 # undef BITFIELDS_BIGENDIAN
43 # undef ALLOW_UNALIGNED_ACCESS
44 #elif defined(__arm__)
45 # undef WORDS_BIGENDIAN
46 # undef BITFIELDS_BIGENDIAN
47 # undef ALLOW_UNALIGNED_ACCESS
48 #elif defined(__aarch64__)
49 # undef WORDS_BIGENDIAN
50 # undef BITFIELDS_BIGENDIAN
51 # undef ALLOW_UNALIGNED_ACCESS
52 #elif defined(__sparc__)
53 # define WORDS_BIGENDIAN
54 # define BITFIELDS_BIGENDIAN
55 # undef ALLOW_UNALIGNED_ACCESS
56 #elif defined(__PPC__)
57 # define WORDS_BIGENDIAN
58 # define BITFIELDS_BIGENDIAN
59 # undef ALLOW_UNALIGNED_ACCESS
60 #elif defined(__s390__)
61 # define WORDS_BIGENDIAN
62 # define BITFIELDS_BIGENDIAN
63 # undef ALLOW_UNALIGNED_ACCESS
64 #elif defined(__e2k__)
65 # undef WORDS_BIGENDIAN
66 # undef BITFIELDS_BIGENDIAN
67 # undef ALLOW_UNALIGNED_ACCESS
68 #elif defined(__MIPSEB__)
69 # define WORDS_BIGENDIAN
70 # define BITFIELDS_BIGENDIAN
71 # undef ALLOW_UNALIGNED_ACCESS
72 #elif defined(__riscv) && defined(__riscv_xlen) && __riscv_xlen == 64
73 # undef WORDS_BIGENDIAN
74 # undef BITFIELDS_BIGENDIAN
75 # undef ALLOW_UNALIGNED_ACCESS
76 #elif !defined(RC_INVOKED)
77 # error Unknown CPU architecture!
78 #endif
79
80
81 #ifndef DECLSPEC_ALIGN
82 # if defined(_MSC_VER) && (_MSC_VER >= 1300) && !defined(MIDL_PASS)
83 #   define DECLSPEC_ALIGN(x) __declspec(align(x))
84 # elif defined(__GNUC__)
85 #   define DECLSPEC_ALIGN(x) __attribute__((aligned(x)))
86 # else
87 #   define DECLSPEC_ALIGN(x)
88 # endif
89 #endif
90
91
92 /* Calling conventions definitions */
93
94 #ifdef __i386__
95 # ifdef _X86_

```



```

96 # define _X86_
97 # endif
98 # if defined(__GNUC__) && ((__GNUC__ > 2) || ((__GNUC__ == 2) && (__GNUC_MINOR__ >= 7)))
99 # define __stdcall __attribute__((__stdcall__))
100 # define __cdecl __attribute__((__cdecl__))
101 # else
102 # error You need gcc >= 2.7 to build Wine on a 386
103 # endif /* __GNUC__ */
104 #else /* __i386__ */
105 # define __stdcall
106 # define __cdecl
107 #endif /* __i386__ */
108
109 #ifndef __WINE__
110 #define pascal __stdcall
111 #define _pascal __stdcall
112 #ifndef __stdcall
113 #define __stdcall __stdcall
114 #endif
115 #ifndef __fastcall
116 #define __fastcall __stdcall
117 #endif
118 #ifndef __fastcall
119 #define __fastcall __stdcall
120 #endif
121 #define __export __stdcall
122 #define cdecl __cdecl
123 #ifndef __cdecl
124 #define __cdecl __cdecl
125 #endif
126
127 #define near
128 #define far
129 #define _near
130 #define _far
131 #define NEAR
132 #define FAR
133
134 #ifndef __cdeclspec
135 #define __cdeclspec(x)
136 #endif
137 #ifndef __cdeclspec
138 #define __cdeclspec(x)
139 #endif
140 #endif /* __WINE__ */
141
142 #define CALLBACK __stdcall
143 #if 0
144 #define WINAPI __stdcall
145 #else
146 #define WINAPI __attribute__((visibility("default")))
147 #endif
148 #define APIPRIVATE __stdcall
149 #define PASCAL __stdcall
150 #define CDECL __cdecl
151 #define _CDECL __cdecl
152 #define WINAPIV __cdecl
153 #define APIENTRY WINAPI
154 #define CONST const
155
156 /* Macro for structure packing and more. */
157
158 #ifdef __GNUC__
159 #define WINE_PACKED __attribute__((packed))
160 #define WINE_UNUSED __attribute__((unused))
161 #define WINE_NORETURN __attribute__((noreturn))
162 #else
163 #define WINE_PACKED /* nothing */
164 #define WINE_UNUSED /* nothing */
165 #define WINE_NORETURN /* nothing */
166 #endif
167
168 /* Anonymous union/struct handling */
169
170 #ifdef __WINE__
171 # define NONAMELESSSTRUCT
172 # define NONAMELESSUNION
173 #else
174 #if !defined(__cplusplus)
175 /* for c we can keep the anonymous version (to avoid compiler warnings) */
176 #define NONAMELESSSTRUCT
177 #define NONAMELESSUNION
178 #else
179 /* Anonymous struct support starts with gcc/g++ 2.96 */
180 #if !defined(NONAMELESSSTRUCT) && defined(__GNUC__) && ((__GNUC__ < 2) || ((__GNUC__ == 2) &&
181 /* && !defined(__cplusplus) */

```



```

182 # define NONAMELESSSTRUCT
183 # endif
184 /* Anonymous unions support starts with gcc 2.96/g++ 2.95 */
185 # if !defined(NONAMELESSUNION) && defined(__GNUC__) && ((__GNUC__ < 2) || ((__GNUC__ == 2) &&
    (__GNUC_MINOR__ < 95) || ((__GNUC_MINOR__ == 95) && !defined(__cplusplus))))
186 # define NONAMELESSUNION
187 # endif
188 #endif
189 #endif
190
191 #ifndef NONAMELESSSTRUCT
192 #define DUMMYSTRUCTNAME
193 #define DUMMYSTRUCTNAME1
194 #define DUMMYSTRUCTNAME2
195 #define DUMMYSTRUCTNAME3
196 #define DUMMYSTRUCTNAME4
197 #define DUMMYSTRUCTNAME5
198 #else /* !defined(NONAMELESSSTRUCT) */
199 #define DUMMYSTRUCTNAME s
200 #define DUMMYSTRUCTNAME1 s1
201 #define DUMMYSTRUCTNAME2 s2
202 #define DUMMYSTRUCTNAME3 s3
203 #define DUMMYSTRUCTNAME4 s4
204 #define DUMMYSTRUCTNAME5 s5
205 #endif /* !defined(NONAMELESSSTRUCT) */
206
207 #ifndef NONAMELESSUNION
208 #define DUMMYUNIONNAME
209 #define DUMMYUNIONNAME1
210 #define DUMMYUNIONNAME2
211 #define DUMMYUNIONNAME3
212 #define DUMMYUNIONNAME4
213 #define DUMMYUNIONNAME5
214 #define DUMMYUNIONNAME6
215 #define DUMMYUNIONNAME7
216 #define DUMMYUNIONNAME8
217 #else /* !defined(NONAMELESSUNION) */
218 #define DUMMYUNIONNAME u
219 #define DUMMYUNIONNAME1 u1
220 #define DUMMYUNIONNAME2 u2
221 #define DUMMYUNIONNAME3 u3
222 #define DUMMYUNIONNAME4 u4
223 #define DUMMYUNIONNAME5 u5
224 #define DUMMYUNIONNAME6 u6
225 #define DUMMYUNIONNAME7 u7
226 #define DUMMYUNIONNAME8 u8
227 #endif /* !defined(NONAMELESSUNION) */
228
229
230 /**** Parts of windef.h that are needed here *****/
231
232 /* Misc. constants. */
233
234 #undef NULL
235 #ifdef __cplusplus
236 #define NULL 0
237 #else
238 #define NULL ((void*)0)
239 #endif
240
241 #ifdef FALSE
242 #undef FALSE
243 #endif
244 #define FALSE 0
245
246 #ifdef TRUE
247 #undef TRUE
248 #endif
249 #define TRUE 1
250
251 #ifdef IN
252 #define IN
253 #endif
254
255 #ifdef OUT
256 #define OUT
257 #endif
258
259 #ifdef OPTIONAL
260 #define OPTIONAL
261 #endif
262
263 /* Standard data types */
264 typedef const void PCVOID, *LPCVOID;
265 typedef int BOOL, *PBOOL, *LPBOOL;
266 typedef unsigned char BYTE, *PBYTE, *LPBYTE;
267 typedef unsigned char UCHAR, *PUCHAR;

```

```

268 typedef unsigned short USHORT,      *PUSHORT,  *LPUSHORT;
269 typedef unsigned short WORD,         *PWORD,    *LPWORD;
270 typedef int INT,                     *PINT,     *LPINT;
271 typedef unsigned int UINT,           *PUINT,    *LPUINT;
272 /* Not sure this is correct. Probably should depend on the compiler, too. */
273 #if defined(__LP64__) || defined(__alpha__)
274 typedef unsigned int DWORD,          *PDWORD,   *LPDWORD;
275 typedef unsigned int ULONG,          *PULONG,   *LPULONG;
276 #else
277 typedef unsigned long DWORD,          *PDWORD,   *LPDWORD;
278 typedef unsigned long ULONG,          *PULONG,   *LPULONG;
279 #endif
280 typedef float FLOAT,                 *PFLOAT,   *LPFLOAT;
281 typedef double DOUBLE,               *PDOUBLE,  *LPDOUBLE;
282 typedef double DATE;
283
284
285 /**** winnt.h proper *****/
286
287 /* Microsoft's macros for declaring functions */
288
289 #ifdef __cplusplus
290 # define EXTERN_C      extern "C"
291 #else
292 # define EXTERN_C      extern
293 #endif
294
295 #ifndef __WINE__
296 #define STDMETHODCALLTYPE      __stdcall
297 #define STDMETHODVCALLTYPE     __cdecl
298 #define STDAPICALLTYPE        __stdcall
299 #define STDAPIVCALLTYPE        __cdecl
300
301 #define STDAPI                EXTERN_C HRESULT STDAPICALLTYPE
302 #define STDAPI_(type)         EXTERN_C type STDAPICALLTYPE
303 #define STDMETHODCALLTYPEIMP  HRESULT STDMETHODCALLTYPE
304 #define STDMETHODCALLTYPEIMP_(type)  type STDMETHODCALLTYPE
305 #define STDAPIV                EXTERN_C HRESULT STDAPIVCALLTYPE
306 #define STDAPIV_(type)         EXTERN_C type STDAPIVCALLTYPE
307 #define STDMETHODCALLTYPEIMPV  HRESULT STDMETHODCALLTYPE
308 #define STDMETHODCALLTYPEIMPV_(type)  type STDMETHODCALLTYPE
309 #endif
310
311 /* Define the basic types */
312 #ifndef VOID
313 #define VOID void
314 #endif
315 typedef VOID *PVOID, *LPVOID;
316 typedef BYTE BOOLEAN, *PBOOLEAN;
317 typedef char CHAR, *PCHAR;
318 typedef short SHORT, *PSHORT;
319 #if defined(__LP64__) || defined(__alpha__)
320 typedef int LONG, *PLONG, *LPLONG;
321 #else
322 typedef long LONG, *PLONG, *LPLONG;
323 #endif
324
325 /* Some systems might have wchar_t, but we really need 16 bit characters */
326 #ifndef WINE_WCHAR_DEFINED
327 #ifdef WINE_UNICODE_NATIVE
328 typedef wchar_t WCHAR, *PWCHAR;
329 #else
330 typedef unsigned short WCHAR, *PWCHAR;
331 #endif
332 #define WINE_WCHAR_DEFINED
333 #endif
334
335 /* 'Extended/Wide' numerical types */
336 #ifndef _ULONGLONG_
337 #define _ULONGLONG_
338 typedef __int64 LONGLONG, *PLONGLONG;
339 typedef __uint64 ULONGLONG, *PULONGLONG;
340 #endif
341
342 #ifndef _DWORDLONG_
343 #define _DWORDLONG_
344 typedef ULONGLONG DWORDLONG, *PDWORDLONG;
345 #endif
346
347 /* ANSI string types */
348 typedef CHAR *PCH, *LPCH;
349 typedef const CHAR *PCCH, *LPCCH;
350 typedef CHAR *PSTR, *LPSTR;
351 typedef const CHAR *PCSTR, *LPCSTR;
352
353 /* Unicode string types */
354 typedef WCHAR *PWCH, *LPWCH;

```

```

355 typedef const WCHAR      *PCWCH,      *LPCWCH;
356 typedef WCHAR            *PWSTR,      *LPWSTR;
357 typedef const WCHAR      *PCWSTR,     *LPCWSTR;
358
359 /* Neutral character and string types */
360 /* These are only defined for Winelib, i.e. _not_ defined for
361 * the emulator. The reason is they depend on the UNICODE
362 * macro which only exists in the user's code.
363 */
364 #ifndef __WINE__
365 # ifdef WINE_UNICODE_REWRITE
366
367 /* Use this if your compiler does not provide a 16bit wchar_t type.
368 * Note that you will need to specify -fwritable-strings or an option
369 * to this effect.
370 * In C++ both WINE_UNICODE_TEXT('c') and WINE_UNICODE_TEXT("str") are
371 * supported, but only the string form can be supported in C.
372 */
373 EXTERN_C unsigned short* wine_rewrite_s4tos2(const wchar_t* str4);
374 #  ifdef __cplusplus
375 inline WCHAR* wine_unicode_text(const wchar_t* str4)
376 {
377     return (WCHAR*)wine_rewrite_s4tos2(str4);
378 }
379 inline WCHAR wine_unicode_text(wchar_t chr4)
380 {
381     return (WCHAR)chr4;
382 }
383 #  define WINE_UNICODE_TEXT(x)          wine_unicode_text(L##x)
384 #  else /* __cplusplus */
385 #  define WINE_UNICODE_TEXT(x)          ((WCHAR*)wine_rewrite_s4tos2(L##x))
386 #  endif /* __cplusplus */
387
388 # else /* WINE_UNICODE_REWRITE */
389
390 /* Define WINE_UNICODE_NATIVE if:
391 * - your compiler provides a 16bit wchar_t type, e.g. gcc >= 2.96 with
392 *   -fshort-wchar option
393 * - or if you decide to use the native 32bit Unix wchar_t type. Be aware
394 *   though that the Wine APIs only support 16bit WCHAR characters for
395 *   binary compatibility reasons.
396 * - or define nothing at all if you don't use Unicode, and blissfully
397 *   ignore the issue :-)
398 */
399 #  define WINE_UNICODE_TEXT(string)     L##string
400
401 #  endif /* WINE_UNICODE_REWRITE */
402
403 #  ifdef UNICODE
404 typedef WCHAR            TCHAR,      *PTCHAR;
405 typedef LPWSTR          PTSTR,      *LPTSTR;
406 typedef LPCWSTR         PCTSTR,     *LPCTSTR;
407 #  define __TEXT(string) WINE_UNICODE_TEXT(string)
408 #  else /* UNICODE */
409 typedef CHAR            TCHAR,      *PTCHAR;
410 typedef LPSTR           PTSTR,      *LPTSTR;
411 typedef LPCSTR          PCTSTR,     *LPCTSTR;
412 #  define __TEXT(string) string
413 #  endif /* UNICODE */
414 #  define TEXT(quote)    __TEXT(quote)
415 #endif /* __WINE__ */
416
417 /* Misc common WIN32 types */
418 typedef LONG            HRESULT;
419 typedef DWORD           LCID,      *PLCID;
420 typedef WORD            LANGID;
421 typedef DWORD           EXECUTION_STATE;
422
423 /* Handle type */
424
425 /* FIXME: Wine does not compile with strict on, therefore strict
426 * handles are presently only usable on machines where sizeof(UINT) ==
427 * sizeof(void*). HANDLES are supposed to be void* but a large amount
428 * of WINE code operates on HANDLES as if they are UINTs. So to WINE
429 * they exist as UINTs but to the Winelib user who turns on strict,
430 * they exist as void*. If there is a size difference between UINT and
431 * void* then things get ugly.
432 *
433 * Here is the plan to convert Wine to STRICT:
434 *
435 * Types will be converted one at a time by volunteers who will compile
436 * Wine with STRICT turned on. Handles that have not been converted yet
437 * will be declared with DECLARE_OLD_HANDLE. Converted handles are
438 * declared with DECLARE_HANDLE.
439 * See the bug report 90 for more details:
440 * http://wine.codeweavers.com/bugzilla/show_bug.cgi?id=90
441 */

```

```

442 /*
443 * when compiling Wine we always treat HANDLE as an UINT. Then when
444 * we're ready we'll remove the '!defined(__WINE__)' (the equivalent
445 * of converting it from DECLARE_OLD_HANDLE to DECLARE_HANDLE).
446 */
447 #if defined(STRICT) && !defined(__WINE__)
448 typedef VOID* HANDLE;
449 #define DECLARE_OLD_HANDLE(a) \
450 typedef struct a##_ { int unused; } *a; \
451 typedef a *P##a, *LP##a
452 #else
453 typedef UINT HANDLE;
454 #define DECLARE_OLD_HANDLE(a) \
455 typedef HANDLE a; \
456 typedef a *P##a, *LP##a
457 #endif
458 #endif
459 typedef HANDLE *PHANDLE, *LPHANDLE;
460
461 #ifdef STRICT
462 #define DECLARE_HANDLE(a) \
463 typedef struct a##_ { int unused; } *a; \
464 typedef a *P##a, *LP##a
465 #else /*STRICT*/
466 #define DECLARE_HANDLE(a) \
467 typedef HANDLE a; \
468 typedef a *P##a, *LP##a
469 #endif /*STRICT*/
470
471 /* Defines */
472
473 /* Argument 1 passed to the DllEntryProc. */
474 #define DLL_PROCESS_DETACH 0 /* detach process (unload library) */
475 #define DLL_PROCESS_ATTACH 1 /* attach process (load library) */
476 #define DLL_THREAD_ATTACH 2 /* attach new thread */
477 #define DLL_THREAD_DETACH 3 /* detach thread */
478
479
480 /* u.x.wProcessorArchitecture (NT) */
481 #define PROCESSOR_ARCHITECTURE_INTEL 0
482 #define PROCESSOR_ARCHITECTURE_MIPS 1
483 #define PROCESSOR_ARCHITECTURE_ALPHA 2
484 #define PROCESSOR_ARCHITECTURE_PPC 3
485 #define PROCESSOR_ARCHITECTURE_SHX 4
486 #define PROCESSOR_ARCHITECTURE_ARM 5
487 #define PROCESSOR_ARCHITECTURE_UNKNOWN 0xFFFF
488
489 /* dwProcessorType */
490 #define PROCESSOR_INTEL_386 386
491 #define PROCESSOR_INTEL_486 486
492 #define PROCESSOR_INTEL_PENTIUM 586
493 #define PROCESSOR_INTEL_860 860
494 #define PROCESSOR_MIPS_R2000 2000
495 #define PROCESSOR_MIPS_R3000 3000
496 #define PROCESSOR_MIPS_R4000 4000
497 #define PROCESSOR_ALPHA_21064 21064
498 #define PROCESSOR_PPC_601 601
499 #define PROCESSOR_PPC_603 603
500 #define PROCESSOR_PPC_604 604
501 #define PROCESSOR_PPC_620 620
502 #define PROCESSOR_HITACHI_SH3 10003
503 #define PROCESSOR_HITACHI_SH3E 10004
504 #define PROCESSOR_HITACHI_SH4 10005
505 #define PROCESSOR_MOTOROLA_821 821
506 #define PROCESSOR_SHx_SH3 103
507 #define PROCESSOR_SHx_SH4 104
508 #define PROCESSOR_STRONGARM 2577
509 #define PROCESSOR_ARM720 1824 /* 0x720 */
510 #define PROCESSOR_ARM820 2080 /* 0x820 */
511 #define PROCESSOR_ARM920 2336 /* 0x920 */
512 #define PROCESSOR_ARM_7TDMI 70001
513
514 typedef struct _MEMORY_BASIC_INFORMATION
515 {
516     LPVOID BaseAddress;
517     LPVOID AllocationBase;
518     DWORD AllocationProtect;
519     DWORD RegionSize;
520     DWORD State;
521     DWORD Protect;
522     DWORD Type;
523 } MEMORY_BASIC_INFORMATION, *LPMEMORY_BASIC_INFORMATION, *PMEMORY_BASIC_INFORMATION;
524
525 #define PAGE_NOACCESS 0x01
526 #define PAGE_READONLY 0x02
527 #define PAGE_READWRITE 0x04
528 #define PAGE_WRITECOPY 0x08

```

```

529 #define PAGE_EXECUTE          0x10
530 #define PAGE_EXECUTE_READ     0x20
531 #define PAGE_EXECUTE_READWRITE 0x40
532 #define PAGE_EXECUTE_WRITECOPY 0x80
533 #define PAGE_GUARD            0x100
534 #define PAGE_NOCACHE          0x200
535
536 #define MEM_COMMIT              0x00001000
537 #define MEM_RESERVE            0x00002000
538 #define MEM_DECOMMIT           0x00004000
539 #define MEM_RELEASE            0x00008000
540 #define MEM_FREE               0x00010000
541 #define MEM_PRIVATE            0x00020000
542 #define MEM_MAPPED             0x00040000
543 #define MEM_RESET              0x00080000
544 #define MEM_TOP_DOWN           0x00100000
545 #ifdef __WINE__
546 #define MEM_SYSTEM              0x80000000
547 #endif
548
549 #define SEC_FILE                0x00800000
550 #define SEC_IMAGE              0x01000000
551 #define SEC_RESERVE            0x04000000
552 #define SEC_COMMIT             0x08000000
553 #define SEC_NOCACHE            0x10000000
554 #define MEM_IMAGE              SEC_IMAGE
555
556
557 #define MINCHAR                0x80
558 #define MAXCHAR                0x7f
559 #define MINSHORT               0x8000
560 #define MAXSHORT               0x7fff
561 #define MINLONG                0x80000000
562 #define MAXLONG                0x7fffffff
563 #define MAXBYTE                0xff
564 #define MAXWORD                0xffff
565 #define MAXDWORD               0xffffffff
566
567 #define FIELD_OFFSET(type, field) \
568 ((LONG)(INT)&((type *)0)->field)
569
570 #define CONTAINING_RECORD(address, type, field) \
571 ((type *)((PCHAR)(address) - (PCHAR)&((type *)0)->field))
572
573 /* Types */
574
575 typedef struct _LIST_ENTRY {
576     struct _LIST_ENTRY *Flink;
577     struct _LIST_ENTRY *Blink;
578 } LIST_ENTRY, *PLIST_ENTRY;
579
580 typedef struct _SINGLE_LIST_ENTRY {
581     struct _SINGLE_LIST_ENTRY *Next;
582 } SINGLE_LIST_ENTRY, *PSINGLE_LIST_ENTRY;
583
584 /* Heap flags */
585
586 #define HEAP_NO_SERIALIZE      0x00000001
587 #define HEAP_GROWABLE         0x00000002
588 #define HEAP_GENERATE_EXCEPTIONS 0x00000004
589 #define HEAP_ZERO_MEMORY      0x00000008
590 #define HEAP_REALLOC_IN_PLACE_ONLY 0x00000010
591 #define HEAP_TAIL_CHECKING_ENABLED 0x00000020
592 #define HEAP_FREE_CHECKING_ENABLED 0x00000040
593 #define HEAP_DISABLE_COALESCE_ON_FREE 0x00000080
594 #define HEAP_CREATE_ALIGN_16  0x00010000
595 #define HEAP_CREATE_ENABLE_TRACING 0x00020000
596
597 /* This flag allows it to create heaps shared by all processes under win95,
598 FIXME: correct name */
599 #define HEAP_SHARED            0x04000000
600
601 /* Processor feature flags. */
602 #define PF_FLOATING_POINT_PRECISION_ERRATA 0
603 #define PF_FLOATING_POINT_EMULATED        1
604 #define PF_COMPARE_EXCHANGE_DOUBLE        2
605 #define PF_MMX_INSTRUCTIONS_AVAILABLE      3
606 #define PF_PPC_MOVEMEM_64BIT_OK           4
607 #define PF_ALPHA_BYTE_INSTRUCTIONS        5
608 #define PF_XMMI_INSTRUCTIONS_AVAILABLE     6
609 #define PF_AMD3D_INSTRUCTIONS_AVAILABLE    7
610 #define PF_RDTSC_INSTRUCTION_AVAILABLE    8
611
612
613 /* Execution state flags */
614 #define ES_SYSTEM_REQUIRED            0x00000001
615 #define ES_DISPLAY_REQUIRED           0x00000002

```

```

616 #define ES_USER_PRESENT      0x00000004
617 #define ES_CONTINUOUS        0x80000000
618
619 /* The Win32 register context */
620
621 /* CONTEXT is the CPU-dependent context; it should be used
622 /* wherever a platform-specific context is needed (e.g. exception */
623 /* handling, Win32 register functions). */
624
625 /* CONTEXT86 is the i386-specific context; it should be used
626 /* wherever only a 386 context makes sense (e.g. DOS interrupts, */
627 /* Win16 register functions), so that this code can be compiled */
628 /* on all platforms. */
629
630 #define SIZE_OF_80387_REGISTERS 80
631
632 typedef struct _FLOATING_SAVE_AREA
633 {
634     DWORD    ControlWord;
635     DWORD    StatusWord;
636     DWORD    TagWord;
637     DWORD    ErrorOffset;
638     DWORD    ErrorSelector;
639     DWORD    DataOffset;
640     DWORD    DataSelector;
641     BYTE     RegisterArea[SIZE_OF_80387_REGISTERS];
642     DWORD    Cr0NpxState;
643 } FLOATING_SAVE_AREA, *PFLOATING_SAVE_AREA;
644
645 #define MAXIMUM_SUPPORTED_EXTENSION 512
646
647 typedef struct _CONTEXT86
648 {
649     DWORD    ContextFlags;
650
651     /* These are selected by CONTEXT_DEBUG_REGISTERS */
652     DWORD    Dr0;
653     DWORD    Dr1;
654     DWORD    Dr2;
655     DWORD    Dr3;
656     DWORD    Dr6;
657     DWORD    Dr7;
658
659     /* These are selected by CONTEXT_FLOATING_POINT */
660     FLOATING_SAVE_AREA FloatSave;
661
662     /* These are selected by CONTEXT_SEGMENTS */
663     DWORD    SegGs;
664     DWORD    SegFs;
665     DWORD    SegEs;
666     DWORD    SegDs;
667
668     /* These are selected by CONTEXT_INTEGER */
669     DWORD    Edi;
670     DWORD    Esi;
671     DWORD    Ebx;
672     DWORD    Edx;
673     DWORD    Ecx;
674     DWORD    Eax;
675
676     /* These are selected by CONTEXT_CONTROL */
677     DWORD    Ebp;
678     DWORD    Eip;
679     DWORD    SegCs;
680     DWORD    EFlags;
681     DWORD    Esp;
682     DWORD    SegSs;
683
684     BYTE     ExtendedRegisters[MAXIMUM_SUPPORTED_EXTENSION];
685 } CONTEXT86;
686
687 #define CONTEXT_X86      0x00010000
688 #define CONTEXT_i386     CONTEXT_X86
689 #define CONTEXT_i486     CONTEXT_X86
690
691 #define CONTEXT86_CONTROL (CONTEXT_i386 | 0x0001) /* SS:SP, CS:IP, FLAGS, BP */
692 #define CONTEXT86_INTEGER (CONTEXT_i386 | 0x0002) /* AX, BX, CX, DX, SI, DI */
693 #define CONTEXT86_SEGMENTS (CONTEXT_i386 | 0x0004) /* DS, ES, FS, GS */
694 #define CONTEXT86_FLOATING_POINT (CONTEXT_i386 | 0x0008) /* 387 state */
695 #define CONTEXT86_DEBUG_REGISTERS (CONTEXT_i386 | 0x0010) /* DB 0-3,6,7 */
696 #define CONTEXT86_FULL (CONTEXT86_CONTROL | CONTEXT86_INTEGER | CONTEXT86_SEGMENTS)
697
698 /* i386 context definitions */
699 #ifdef __i386__
700
701 #define CONTEXT_CONTROL      CONTEXT86_CONTROL
702 #define CONTEXT_INTEGER      CONTEXT86_INTEGER

```

```

703 #define CONTEXT_SEGMENTS          CONTEXT86_SEGMENTS
704 #define CONTEXT_FLOATING_POINT    CONTEXT86_FLOATING_POINT
705 #define CONTEXT_DEBUG_REGISTERS   CONTEXT86_DEBUG_REGISTERS
706 #define CONTEXT_FULL              CONTEXT86_FULL
707
708 typedef CONTEXT86 CONTEXT;
709
710 #endif /* __i386__ */
711
712 /* x86-64 context definitions */
713 #if defined(__x86_64__)
714
715 #define CONTEXT_AMD64             0x00100000
716
717 #define CONTEXT_CONTROL           (CONTEXT_AMD64 | 0x0001)
718 #define CONTEXT_INTEGER           (CONTEXT_AMD64 | 0x0002)
719 #define CONTEXT_SEGMENTS         (CONTEXT_AMD64 | 0x0004)
720 #define CONTEXT_FLOATING_POINT   (CONTEXT_AMD64 | 0x0008L)
721 #define CONTEXT_DEBUG_REGISTERS (CONTEXT_AMD64 | 0x0010L)
722 #define CONTEXT_FULL             (CONTEXT_CONTROL | CONTEXT_INTEGER | CONTEXT_FLOATING_POINT)
723 #define CONTEXT_ALL              (CONTEXT_CONTROL | CONTEXT_INTEGER | CONTEXT_SEGMENTS | CONTEXT_FLOATING_POINT |
    CONTEXT_DEBUG_REGISTERS)
724
725 #define EXCEPTION_READ_FAULT      0
726 #define EXCEPTION_WRITE_FAULT    1
727 #define EXCEPTION_EXECUTE_FAULT  8
728
729 typedef struct DECLSPEC_ALIGN(16) _M128A {
730     ULONGLONG Low;
731     LONGLONG High;
732 } M128A, *PM128A;
733
734 typedef struct _XMM_SAVE_AREA32 {
735     WORD ControlWord; /* 000 */
736     WORD StatusWord; /* 002 */
737     BYTE TagWord; /* 004 */
738     BYTE Reserved1; /* 005 */
739     WORD ErrorOpcode; /* 006 */
740     DWORD ErrorOffset; /* 008 */
741     WORD ErrorSelector; /* 00c */
742     WORD Reserved2; /* 00e */
743     DWORD DataOffset; /* 010 */
744     WORD DataSelector; /* 014 */
745     WORD Reserved3; /* 016 */
746     DWORD MxCsr; /* 018 */
747     DWORD MxCsr_Mask; /* 01c */
748     M128A FloatRegisters[8]; /* 020 */
749     M128A XmmRegisters[16]; /* 0a0 */
750     BYTE Reserved4[96]; /* 1a0 */
751 } XMM_SAVE_AREA32, *PXMM_SAVE_AREA32;
752
753 typedef struct DECLSPEC_ALIGN(16) _CONTEXT {
754     DWORD64 P1Home; /* 000 */
755     DWORD64 P2Home; /* 008 */
756     DWORD64 P3Home; /* 010 */
757     DWORD64 P4Home; /* 018 */
758     DWORD64 P5Home; /* 020 */
759     DWORD64 P6Home; /* 028 */
760
761     /* Control flags */
762     DWORD ContextFlags; /* 030 */
763     DWORD MxCsr; /* 034 */
764
765     /* Segment */
766     WORD SegCs; /* 038 */
767     WORD SegDs; /* 03a */
768     WORD SegEs; /* 03c */
769     WORD SegFs; /* 03e */
770     WORD SegGs; /* 040 */
771     WORD SegSs; /* 042 */
772     DWORD EFlags; /* 044 */
773
774     /* Debug */
775     DWORD64 Dr0; /* 048 */
776     DWORD64 Dr1; /* 050 */
777     DWORD64 Dr2; /* 058 */
778     DWORD64 Dr3; /* 060 */
779     DWORD64 Dr6; /* 068 */
780     DWORD64 Dr7; /* 070 */
781
782     /* Integer */
783     DWORD64 Rax; /* 078 */
784     DWORD64 Rcx; /* 080 */
785     DWORD64 Rdx; /* 088 */
786     DWORD64 Rbx; /* 090 */
787     DWORD64 Rsp; /* 098 */
788     DWORD64 Rbp; /* 0a0 */

```

```

789     DWORD64 Rsi;                /* 0a8 */
790     DWORD64 Rdi;                /* 0b0 */
791     DWORD64 R8;                 /* 0b8 */
792     DWORD64 R9;                 /* 0c0 */
793     DWORD64 R10;               /* 0c8 */
794     DWORD64 R11;               /* 0d0 */
795     DWORD64 R12;               /* 0d8 */
796     DWORD64 R13;               /* 0e0 */
797     DWORD64 R14;               /* 0e8 */
798     DWORD64 R15;               /* 0f0 */
799
800     /* Counter */
801     DWORD64 Rip;                /* 0f8 */
802
803     /* Floating point */
804     union {
805         XMM_SAVE_AREA32 FltSave; /* 100 */
806         struct {
807             M128A Header[2];      /* 100 */
808             M128A Legacy[8];      /* 120 */
809             M128A Xmm0;           /* 1a0 */
810             M128A Xmm1;           /* 1b0 */
811             M128A Xmm2;           /* 1c0 */
812             M128A Xmm3;           /* 1d0 */
813             M128A Xmm4;           /* 1e0 */
814             M128A Xmm5;           /* 1f0 */
815             M128A Xmm6;           /* 200 */
816             M128A Xmm7;           /* 210 */
817             M128A Xmm8;           /* 220 */
818             M128A Xmm9;           /* 230 */
819             M128A Xmm10;          /* 240 */
820             M128A Xmm11;          /* 250 */
821             M128A Xmm12;          /* 260 */
822             M128A Xmm13;          /* 270 */
823             M128A Xmm14;          /* 280 */
824             M128A Xmm15;          /* 290 */
825         } DUMMYSTRUCTNAME;
826     } DUMMYUNIONNAME;
827
828     /* Vector */
829     M128A VectorRegister[26];    /* 300 */
830     DWORD64 VectorControl;       /* 4a0 */
831
832     /* Debug control */
833     DWORD64 DebugControl;        /* 4a8 */
834     DWORD64 LastBranchToRip;     /* 4b0 */
835     DWORD64 LastBranchFromRip;   /* 4b8 */
836     DWORD64 LastExceptionToRip;  /* 4c0 */
837     DWORD64 LastExceptionFromRip; /* 4c8 */
838 } CONTEXT;
839
840 typedef struct _RUNTIME_FUNCTION
841 {
842     DWORD BeginAddress;
843     DWORD EndAddress;
844     DWORD UnwindData;
845 } RUNTIME_FUNCTION, *PRUNTIME_FUNCTION;
846
847 #define UNWIND_HISTORY_TABLE_SIZE 12
848
849 typedef struct _UNWIND_HISTORY_TABLE_ENTRY
850 {
851     ULONG64 ImageBase;
852     PRUNTIME_FUNCTION FunctionEntry;
853 } UNWIND_HISTORY_TABLE_ENTRY, *PUNWIND_HISTORY_TABLE_ENTRY;
854
855 #define UNWIND_HISTORY_TABLE_NONE 0
856 #define UNWIND_HISTORY_TABLE_GLOBAL 1
857 #define UNWIND_HISTORY_TABLE_LOCAL 2
858
859 typedef struct _UNWIND_HISTORY_TABLE
860 {
861     ULONG Count;
862     UCHAR Search;
863     ULONG64 LowAddress;
864     ULONG64 HighAddress;
865     UNWIND_HISTORY_TABLE_ENTRY Entry[UNWIND_HISTORY_TABLE_SIZE];
866 } UNWIND_HISTORY_TABLE, *PUNWIND_HISTORY_TABLE;
867
868 typedef struct _KNONVOLATILE_CONTEXT_POINTERS
869 {
870     union
871     {
872         PM128A FloatingContext[16];
873         struct
874         {
875             PM128A Xmm0;

```



```

876         PM128A Xmm1;
877         PM128A Xmm2;
878         PM128A Xmm3;
879         PM128A Xmm4;
880         PM128A Xmm5;
881         PM128A Xmm6;
882         PM128A Xmm7;
883         PM128A Xmm8;
884         PM128A Xmm9;
885         PM128A Xmm10;
886         PM128A Xmm11;
887         PM128A Xmm12;
888         PM128A Xmm13;
889         PM128A Xmm14;
890         PM128A Xmm15;
891     } DUMMYSTRUCTNAME1;
892 } DUMMYUNIONNAME1;
893
894 union
895 {
896     PULONG64 IntegerContext[16];
897     struct
898     {
899         PULONG64 Rax;
900         PULONG64 Rcx;
901         PULONG64 Rdx;
902         PULONG64 Rbx;
903         PULONG64 Rsp;
904         PULONG64 Rbp;
905         PULONG64 Rsi;
906         PULONG64 Rdi;
907         PULONG64 R8;
908         PULONG64 R9;
909         PULONG64 R10;
910         PULONG64 R11;
911         PULONG64 R12;
912         PULONG64 R13;
913         PULONG64 R14;
914         PULONG64 R15;
915     } DUMMYSTRUCTNAME2;
916 } DUMMYUNIONNAME2;
917 } KNONVOLATILE_CONTEXT_POINTERS, *PKNONVOLATILE_CONTEXT_POINTERS;
918
919 BOOLEAN CDECL RtlAddFunctionTable(RUNTIME_FUNCTION*, DWORD, DWORD64);
920 BOOLEAN CDECL RtlDeleteFunctionTable(RUNTIME_FUNCTION*);
921 PRUNTIME_FUNCTION WINAPI RtlLookupFunctionEntry(DWORD64, DWORD64*, UNWIND_HISTORY_TABLE*);
922 PVOID WINAPI
923     RtlVirtualUnwind(ULONG, ULONG64, ULONG64, RUNTIME_FUNCTION*, CONTEXT*, PVOID*, ULONG64*, KNONVOLATILE_CONTEXT_POINTERS*);
924
924 #define UNW_FLAG_NHANDLER 0
925 #define UNW_FLAG_EHANDLER 1
926 #define UNW_FLAG_UHANDLER 2
927 #define UNW_FLAG_CHAININFO 4
928
929 #endif /* __x86_64__ */
930
931 /* Alpha context definitions */
932 #if defined(__alpha__)
933
934 #define CONTEXT_ALPHA 0x00020000
935
936 #define CONTEXT_CONTROL (CONTEXT_ALPHA | 0x00000001L)
937 #define CONTEXT_FLOATING_POINT (CONTEXT_ALPHA | 0x00000002L)
938 #define CONTEXT_INTEGER (CONTEXT_ALPHA | 0x00000004L)
939 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_FLOATING_POINT | CONTEXT_INTEGER)
940
941 typedef struct _CONTEXT
942 {
943     /* selected by CONTEXT_FLOATING_POINT */
944     ULONGLONG FltF0;
945     ULONGLONG FltF1;
946     ULONGLONG FltF2;
947     ULONGLONG FltF3;
948     ULONGLONG FltF4;
949     ULONGLONG FltF5;
950     ULONGLONG FltF6;
951     ULONGLONG FltF7;
952     ULONGLONG FltF8;
953     ULONGLONG FltF9;
954     ULONGLONG FltF10;
955     ULONGLONG FltF11;
956     ULONGLONG FltF12;
957     ULONGLONG FltF13;
958     ULONGLONG FltF14;
959     ULONGLONG FltF15;
960     ULONGLONG FltF16;
961     ULONGLONG FltF17;

```

```

962     ULONGLONG FltF18;
963     ULONGLONG FltF19;
964     ULONGLONG FltF20;
965     ULONGLONG FltF21;
966     ULONGLONG FltF22;
967     ULONGLONG FltF23;
968     ULONGLONG FltF24;
969     ULONGLONG FltF25;
970     ULONGLONG FltF26;
971     ULONGLONG FltF27;
972     ULONGLONG FltF28;
973     ULONGLONG FltF29;
974     ULONGLONG FltF30;
975     ULONGLONG FltF31;
976
977     /* selected by CONTEXT_INTEGER */
978     ULONGLONG IntV0;
979     ULONGLONG IntT0;
980     ULONGLONG IntT1;
981     ULONGLONG IntT2;
982     ULONGLONG IntT3;
983     ULONGLONG IntT4;
984     ULONGLONG IntT5;
985     ULONGLONG IntT6;
986     ULONGLONG IntT7;
987     ULONGLONG IntS0;
988     ULONGLONG IntS1;
989     ULONGLONG IntS2;
990     ULONGLONG IntS3;
991     ULONGLONG IntS4;
992     ULONGLONG IntS5;
993     ULONGLONG IntFp;
994     ULONGLONG IntA0;
995     ULONGLONG IntA1;
996     ULONGLONG IntA2;
997     ULONGLONG IntA3;
998     ULONGLONG IntA4;
999     ULONGLONG IntA5;
1000     ULONGLONG IntT8;
1001     ULONGLONG IntT9;
1002     ULONGLONG IntT10;
1003     ULONGLONG IntT11;
1004     ULONGLONG IntRa;
1005     ULONGLONG IntT12;
1006     ULONGLONG IntAt;
1007     ULONGLONG IntGp;
1008     ULONGLONG IntSp;
1009     ULONGLONG IntZero;
1010
1011     /* selected by CONTEXT_FLOATING_POINT */
1012     ULONGLONG FpCr;
1013     ULONGLONG SoftFpCr;
1014
1015     /* selected by CONTEXT_CONTROL */
1016     ULONGLONG Fir;
1017     DWORD Psr;
1018     DWORD ContextFlags;
1019     DWORD Fill[4];
1020 } CONTEXT;
1021
1022 #define _QUAD_PSR_OFFSET    HighSoftFpCr
1023 #define _QUAD_FLAGS_OFFSET HighFir
1024
1025 #endif /* _ALPHA */
1026
1027 /* Mips context definitions */
1028 #if defined(_MIPS_) || defined(__MIPS__) || defined(__mips__)
1029
1030 #define CONTEXT_R4000    0x00010000
1031
1032 #define CONTEXT_CONTROL    (CONTEXT_R4000 | 0x00000001)
1033 #define CONTEXT_FLOATING_POINT (CONTEXT_R4000 | 0x00000002)
1034 #define CONTEXT_INTEGER    (CONTEXT_R4000 | 0x00000004)
1035
1036 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_FLOATING_POINT | CONTEXT_INTEGER)
1037
1038 typedef struct _CONTEXT
1039 {
1040     DWORD Argument[4];
1041     /* These are selected by CONTEXT_FLOATING_POINT */
1042     DWORD FltF0;
1043     DWORD FltF1;
1044     DWORD FltF2;
1045     DWORD FltF3;
1046     DWORD FltF4;
1047     DWORD FltF5;
1048     DWORD FltF6;

```

```

1049     DWORD FltF7;
1050     DWORD FltF8;
1051     DWORD FltF9;
1052     DWORD FltF10;
1053     DWORD FltF11;
1054     DWORD FltF12;
1055     DWORD FltF13;
1056     DWORD FltF14;
1057     DWORD FltF15;
1058     DWORD FltF16;
1059     DWORD FltF17;
1060     DWORD FltF18;
1061     DWORD FltF19;
1062     DWORD FltF20;
1063     DWORD FltF21;
1064     DWORD FltF22;
1065     DWORD FltF23;
1066     DWORD FltF24;
1067     DWORD FltF25;
1068     DWORD FltF26;
1069     DWORD FltF27;
1070     DWORD FltF28;
1071     DWORD FltF29;
1072     DWORD FltF30;
1073     DWORD FltF31;
1074
1075     /* These are selected by CONTEXT_INTEGER */
1076     DWORD IntZero;
1077     DWORD IntAt;
1078     DWORD IntV0;
1079     DWORD IntV1;
1080     DWORD IntA0;
1081     DWORD IntA1;
1082     DWORD IntA2;
1083     DWORD IntA3;
1084     DWORD IntT0;
1085     DWORD IntT1;
1086     DWORD IntT2;
1087     DWORD IntT3;
1088     DWORD IntT4;
1089     DWORD IntT5;
1090     DWORD IntT6;
1091     DWORD IntT7;
1092     DWORD IntS0;
1093     DWORD IntS1;
1094     DWORD IntS2;
1095     DWORD IntS3;
1096     DWORD IntS4;
1097     DWORD IntS5;
1098     DWORD IntS6;
1099     DWORD IntS7;
1100     DWORD IntT8;
1101     DWORD IntT9;
1102     DWORD IntK0;
1103     DWORD IntK1;
1104     DWORD IntGp;
1105     DWORD IntSp;
1106     DWORD IntS8;
1107     DWORD IntRa;
1108     DWORD IntLo;
1109     DWORD IntHi;
1110
1111     /* These are selected by CONTEXT_FLOATING_POINT */
1112     DWORD Fsr;
1113
1114     /* These are selected by CONTEXT_CONTROL */
1115     DWORD Fir;
1116     DWORD Psr;
1117
1118     DWORD ContextFlags;
1119     DWORD Fill[2];
1120 } CONTEXT;
1121
1122 #endif /* _MIPS_ */
1123
1124 /* PowerPC context definitions */
1125 #ifdef __PPC__
1126
1127 #define CONTEXT_CONTROL      0x0001
1128 #define CONTEXT_FLOATING_POINT 0x0002
1129 #define CONTEXT_INTEGER      0x0004
1130 #define CONTEXT_DEBUG_REGISTERS 0x0008
1131 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_FLOATING_POINT | CONTEXT_INTEGER)
1132
1133 typedef struct
1134 {
1135     /* These are selected by CONTEXT_FLOATING_POINT */

```

```
1136     double Fpr0;
1137     double Fpr1;
1138     double Fpr2;
1139     double Fpr3;
1140     double Fpr4;
1141     double Fpr5;
1142     double Fpr6;
1143     double Fpr7;
1144     double Fpr8;
1145     double Fpr9;
1146     double Fpr10;
1147     double Fpr11;
1148     double Fpr12;
1149     double Fpr13;
1150     double Fpr14;
1151     double Fpr15;
1152     double Fpr16;
1153     double Fpr17;
1154     double Fpr18;
1155     double Fpr19;
1156     double Fpr20;
1157     double Fpr21;
1158     double Fpr22;
1159     double Fpr23;
1160     double Fpr24;
1161     double Fpr25;
1162     double Fpr26;
1163     double Fpr27;
1164     double Fpr28;
1165     double Fpr29;
1166     double Fpr30;
1167     double Fpr31;
1168     double Fpscr;
1169
1170     /* These are selected by CONTEXT_INTEGER */
1171     DWORD Gpr0;
1172     DWORD Gpr1;
1173     DWORD Gpr2;
1174     DWORD Gpr3;
1175     DWORD Gpr4;
1176     DWORD Gpr5;
1177     DWORD Gpr6;
1178     DWORD Gpr7;
1179     DWORD Gpr8;
1180     DWORD Gpr9;
1181     DWORD Gpr10;
1182     DWORD Gpr11;
1183     DWORD Gpr12;
1184     DWORD Gpr13;
1185     DWORD Gpr14;
1186     DWORD Gpr15;
1187     DWORD Gpr16;
1188     DWORD Gpr17;
1189     DWORD Gpr18;
1190     DWORD Gpr19;
1191     DWORD Gpr20;
1192     DWORD Gpr21;
1193     DWORD Gpr22;
1194     DWORD Gpr23;
1195     DWORD Gpr24;
1196     DWORD Gpr25;
1197     DWORD Gpr26;
1198     DWORD Gpr27;
1199     DWORD Gpr28;
1200     DWORD Gpr29;
1201     DWORD Gpr30;
1202     DWORD Gpr31;
1203
1204     DWORD Cr;
1205     DWORD Xer;
1206
1207     /* These are selected by CONTEXT_CONTROL */
1208     DWORD Msr;
1209     DWORD Iar;
1210     DWORD Lr;
1211     DWORD Ctr;
1212
1213     DWORD ContextFlags;
1214     DWORD Fill[3];
1215
1216     /* These are selected by CONTEXT_DEBUG_REGISTERS */
1217     DWORD Dr0;
1218     DWORD Dr1;
1219     DWORD Dr2;
1220     DWORD Dr3;
1221     DWORD Dr4;
1222     DWORD Dr5;
```

```

1223     DWORD Dr6;
1224     DWORD Dr7;
1225 } CONTEXT;
1226
1227 typedef struct _STACK_FRAME_HEADER
1228 {
1229     DWORD BackChain;
1230     DWORD GlueSaved1;
1231     DWORD GlueSaved2;
1232     DWORD Reserved1;
1233     DWORD Spare1;
1234     DWORD Spare2;
1235
1236     DWORD Parameter0;
1237     DWORD Parameter1;
1238     DWORD Parameter2;
1239     DWORD Parameter3;
1240     DWORD Parameter4;
1241     DWORD Parameter5;
1242     DWORD Parameter6;
1243     DWORD Parameter7;
1244 } STACK_FRAME_HEADER, *PSTACK_FRAME_HEADER;
1245
1246 #endif /* __PPC__ */
1247
1248 #ifdef __sparc__
1249
1250 /*
1251  * FIXME:
1252  *
1253  * There is no official CONTEXT structure defined for the SPARC
1254  * architecture, so I just made one up.
1255  *
1256  * This structure is valid only for 32-bit SPARC architectures,
1257  * not for 64-bit SPARC.
1258  *
1259  * Note that this structure contains only the 'top-level' registers;
1260  * the rest of the register window chain is not visible.
1261  *
1262  * The layout follows the Solaris 'prgregset_t' structure.
1263  */
1264
1265 #define CONTEXT_SPARC          0x10000000
1266
1267 #define CONTEXT_CONTROL        (CONTEXT_SPARC | 0x00000001)
1268 #define CONTEXT_FLOATING_POINT (CONTEXT_SPARC | 0x00000002)
1269 #define CONTEXT_INTEGER        (CONTEXT_SPARC | 0x00000004)
1270
1271 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_FLOATING_POINT | CONTEXT_INTEGER)
1272
1273 typedef struct _CONTEXT
1274 {
1275     DWORD ContextFlags;
1276
1277     /* These are selected by CONTEXT_INTEGER */
1278     DWORD g0;
1279     DWORD g1;
1280     DWORD g2;
1281     DWORD g3;
1282     DWORD g4;
1283     DWORD g5;
1284     DWORD g6;
1285     DWORD g7;
1286     DWORD o0;
1287     DWORD o1;
1288     DWORD o2;
1289     DWORD o3;
1290     DWORD o4;
1291     DWORD o5;
1292     DWORD o6;
1293     DWORD o7;
1294     DWORD i0;
1295     DWORD i1;
1296     DWORD i2;
1297     DWORD i3;
1298     DWORD i4;
1299     DWORD i5;
1300     DWORD i6;

```

```

1310     DWORD i7;
1311
1312     /* These are selected by CONTEXT_CONTROL */
1313     DWORD psr;
1314     DWORD pc;
1315     DWORD npc;
1316     DWORD y;
1317     DWORD wim;
1318     DWORD tbr;
1319
1320     /* FIXME: floating point registers missing */
1321
1322 } CONTEXT;
1323
1324 #endif /* __sparc__ */
1325
1326 #ifdef __s390__
1327
1328 /*
1329  * FIXME:
1330  *
1331  * There is no official CONTEXT structure defined for the S390
1332  * architecture, so I just made one up.
1333  *
1334  * Note that this structure contains only the 'top-level' registers;
1335  * the rest of the register window chain is not visible.
1336  *
1337  * The layout is based on the sparc one.
1338  *
1339  */
1340
1341 #define CONTEXT_S390C          0x20000000
1342
1343 #define CONTEXT_CONTROL        (CONTEXT_S390 | 0x00000001)
1344 #define CONTEXT_FLOATING_POINT (CONTEXT_S390 | 0x00000002)
1345 #define CONTEXT_INTEGER        (CONTEXT_S390 | 0x00000004)
1346
1347 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_FLOATING_POINT | CONTEXT_INTEGER)
1348
1349 typedef struct _CONTEXT
1350 {
1351     DWORD ContextFlags;
1352
1353     /* These are selected by CONTEXT_INTEGER */
1354     DWORD r0;
1355     DWORD r1;
1356     DWORD r2;
1357     DWORD r3;
1358     DWORD r4;
1359     DWORD r5;
1360     DWORD r6;
1361     DWORD r7;
1362     DWORD r8;
1363     DWORD r9;
1364     DWORD r10;
1365     DWORD r11;
1366     DWORD r12;
1367     DWORD r13;
1368     DWORD r14;
1369     DWORD r15;
1370
1371     /* FIXME: this section is fictional (copied from sparc) */
1372     DWORD psr;
1373     DWORD pc;
1374     DWORD npc;
1375     DWORD y;
1376     DWORD wim;
1377     DWORD tbr;
1378
1379     /* FIXME: floating point registers missing */
1380
1381 } CONTEXT;
1382
1383 #endif /* __s390__ */
1384
1385 #ifdef __arm__
1386
1387 /* These definitions are taken directly from wine
1388  * http://source.winehq.org/git/wine.git/blob_plain/HEAD:/include/winnt.h */
1389
1390 /* The following flags control the contents of the CONTEXT structure. */
1391
1392 #define CONTEXT_ARM          0x02000000
1393 #define CONTEXT_CONTROL        (CONTEXT_ARM | 0x00000001)
1394 #define CONTEXT_INTEGER        (CONTEXT_ARM | 0x00000002)
1395 #define CONTEXT_FLOATING_POINT (CONTEXT_ARM | 0x00000004)
1396 #define CONTEXT_DEBUG_REGISTERS (CONTEXT_ARM | 0x00000008)

```

```

1397
1398 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_INTEGER)
1399
1400 #define EXCEPTION_READ_FAULT    0
1401 #define EXCEPTION_WRITE_FAULT   1
1402 #define EXCEPTION_EXECUTE_FAULT 8
1403
1404 typedef struct _CONTEXT {
1405     /* The flags values within this flag control the contents of
1406     * a CONTEXT record.
1407     *
1408     * If the context record is used as an input parameter, then
1409     * for each portion of the context record controlled by a flag
1410     * whose value is set, it is assumed that that portion of the
1411     * context record contains valid context.  If the context record
1412     * is being used to modify a thread's context, then only that
1413     * portion of the threads context will be modified.
1414     *
1415     * If the context record is used as an IN OUT parameter to capture
1416     * the context of a thread, then only those portions of the thread's
1417     * context corresponding to set flags will be returned.
1418     *
1419     * The context record is never used as an OUT only parameter.  */
1420
1421     ULONG ContextFlags;
1422
1423     /* This section is specified/returned if the ContextFlags word contains
1424     * the flag CONTEXT_INTEGER.  */
1425     ULONG R0;
1426     ULONG R1;
1427     ULONG R2;
1428     ULONG R3;
1429     ULONG R4;
1430     ULONG R5;
1431     ULONG R6;
1432     ULONG R7;
1433     ULONG R8;
1434     ULONG R9;
1435     ULONG R10;
1436     ULONG Fp;
1437     ULONG Ip;
1438
1439     /* These are selected by CONTEXT_CONTROL */
1440     ULONG Sp;
1441     ULONG Lr;
1442     ULONG Pc;
1443     ULONG Cpsr;
1444 } CONTEXT;
1445
1446 #endif /* __arm__ */
1447
1448 #ifdef __aarch64__
1449     /*
1450     * FIXME:
1451     *
1452     * There is not yet an official CONTEXT structure defined for the AArch64
1453     * architecture, so I just made one up.
1454     *
1455     */
1456
1457     /* These definitions are taken directly from wine
1458     * http://source.winehq.org/git/wine.git/blob\_plain/HEAD:/include/winnt.h */
1459
1460     #define CONTEXT_ARM64            0x20000000
1461     #define CONTEXT_CONTROL          (CONTEXT_ARM64 | 0x00000001)
1462     #define CONTEXT_INTEGER          (CONTEXT_ARM64 | 0x00000002)
1463     #define CONTEXT_FLOATING_POINT  (CONTEXT_ARM64 | 0x00000004)
1464     #define CONTEXT_DEBUG_REGISTERS (CONTEXT_ARM64 | 0x00000008)
1465
1466     #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_INTEGER)
1467
1468     #define EXCEPTION_READ_FAULT    0
1469     #define EXCEPTION_WRITE_FAULT   1
1470     #define EXCEPTION_EXECUTE_FAULT 8
1471
1472     typedef struct _CONTEXT {
1473         ULONG ContextFlags;
1474
1475         /* This section is specified/returned if the ContextFlags word contains
1476         the flag CONTEXT_INTEGER.  */
1477         ULONGLONG X0;
1478         ULONGLONG X1;
1479         ULONGLONG X2;
1480         ULONGLONG X3;
1481         ULONGLONG X4;
1482         ULONGLONG X5;
1483         ULONGLONG X6;

```

```
1484     ULONGLONG X7;
1485     ULONGLONG X8;
1486     ULONGLONG X9;
1487     ULONGLONG X10;
1488     ULONGLONG X11;
1489     ULONGLONG X12;
1490     ULONGLONG X13;
1491     ULONGLONG X14;
1492     ULONGLONG X15;
1493     ULONGLONG X16;
1494     ULONGLONG X17;
1495     ULONGLONG X18;
1496     ULONGLONG X19;
1497     ULONGLONG X20;
1498     ULONGLONG X21;
1499     ULONGLONG X22;
1500     ULONGLONG X23;
1501     ULONGLONG X24;
1502     ULONGLONG X25;
1503     ULONGLONG X26;
1504     ULONGLONG X27;
1505     ULONGLONG X28;
1506     ULONGLONG X29;
1507     ULONGLONG X30;
1508
1509     /* These are selected by CONTEXT_CONTROL */
1510     ULONGLONG Sp;
1511     ULONGLONG Pc;
1512     ULONGLONG PState;
1513
1514     /* These are selected by CONTEXT_FLOATING_POINT */
1515     /* FIXME */
1516 } CONTEXT;
1517
1518 #endif /* __aarch64__ */
1519
1520 #ifdef __e2k__
1521 /*
1522  * FIXME:
1523  *
1524  * There is not yet an official CONTEXT structure defined for the
1525  * Elbrus 2000 architecture (64-bit LE), so I just made one up.
1526  */
1527 */
1528
1529 #define CONTEXT_E2K          0x4000000
1530 #define CONTEXT_CONTROL     (CONTEXT_E2K | 0x00000001)
1531 #define CONTEXT_INTEGER     (CONTEXT_E2K | 0x00000002)
1532 #define CONTEXT_FLOATING_POINT (CONTEXT_E2K | 0x00000004)
1533 #define CONTEXT_DEBUG_REGISTERS (CONTEXT_E2K | 0x00000008)
1534
1535 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_INTEGER)
1536
1537 #define EXCEPTION_READ_FAULT 0
1538 #define EXCEPTION_WRITE_FAULT 1
1539 #define EXCEPTION_EXECUTE_FAULT 8
1540
1541 typedef struct _CONTEXT {
1542     ULONG ContextFlags;
1543
1544     /* This section is specified/returned if the ContextFlags word contains
1545     the flag CONTEXT_INTEGER. */
1546     ULONGLONG X0;
1547     ULONGLONG X1;
1548     ULONGLONG X2;
1549     ULONGLONG X3;
1550     ULONGLONG X4;
1551     ULONGLONG X5;
1552     ULONGLONG X6;
1553     ULONGLONG X7;
1554     ULONGLONG X8;
1555     ULONGLONG X9;
1556     ULONGLONG X10;
1557     ULONGLONG X11;
1558     ULONGLONG X12;
1559     ULONGLONG X13;
1560     ULONGLONG X14;
1561     ULONGLONG X15;
1562     ULONGLONG X16;
1563     ULONGLONG X17;
1564     ULONGLONG X18;
1565     ULONGLONG X19;
1566     ULONGLONG X20;
1567     ULONGLONG X21;
1568     ULONGLONG X22;
1569     ULONGLONG X23;
1570     ULONGLONG X24;
```



```

1571     ULONGLONG X25;
1572     ULONGLONG X26;
1573     ULONGLONG X27;
1574     ULONGLONG X28;
1575     ULONGLONG X29;
1576     ULONGLONG X30;
1577
1578     /* These are selected by CONTEXT_CONTROL */
1579     ULONGLONG Sp;
1580     ULONGLONG Pc;
1581     ULONGLONG PState;
1582
1583     /* These are selected by CONTEXT_FLOATING_POINT */
1584     /* FIXME */
1585 } CONTEXT;
1586
1587 #endif /* __e2k__ */
1588
1589 #ifdef __riscv && __riscv_xlen==64
1590 /*
1591  * FIXME:
1592  *
1593  * There is not yet an official CONTEXT structure defined for the
1594  * riscv64 architecture (64-bit LE), so I just made one up.
1595  */
1596 */
1597
1598 #define CONTEXT_RISCV64          0x40000000
1599 #define CONTEXT_CONTROL          (CONTEXT_RISCV64 | 0x00000001)
1600 #define CONTEXT_INTEGER          (CONTEXT_RISCV64 | 0x00000002)
1601 #define CONTEXT_FLOATING_POINT  (CONTEXT_RISCV64 | 0x00000004)
1602 #define CONTEXT_DEBUG_REGISTERS (CONTEXT_RISCV64 | 0x00000008)
1603
1604 #define CONTEXT_FULL (CONTEXT_CONTROL | CONTEXT_INTEGER)
1605
1606 #define EXCEPTION_READ_FAULT    0
1607 #define EXCEPTION_WRITE_FAULT   1
1608 #define EXCEPTION_EXECUTE_FAULT 8
1609
1610 typedef struct _CONTEXT {
1611     ULONG ContextFlags;
1612
1613     /* This section is specified/returned if the ContextFlags word contains
1614     the flag CONTEXT_INTEGER. */
1615     ULONGLONG X0;
1616     ULONGLONG X1;
1617     ULONGLONG X2;
1618     ULONGLONG X3;
1619     ULONGLONG X4;
1620     ULONGLONG X5;
1621     ULONGLONG X6;
1622     ULONGLONG X7;
1623     ULONGLONG X8;
1624     ULONGLONG X9;
1625     ULONGLONG X10;
1626     ULONGLONG X11;
1627     ULONGLONG X12;
1628     ULONGLONG X13;
1629     ULONGLONG X14;
1630     ULONGLONG X15;
1631     ULONGLONG X16;
1632     ULONGLONG X17;
1633     ULONGLONG X18;
1634     ULONGLONG X19;
1635     ULONGLONG X20;
1636     ULONGLONG X21;
1637     ULONGLONG X22;
1638     ULONGLONG X23;
1639     ULONGLONG X24;
1640     ULONGLONG X25;
1641     ULONGLONG X26;
1642     ULONGLONG X27;
1643     ULONGLONG X28;
1644     ULONGLONG X29;
1645     ULONGLONG X30;
1646     ULONGLONG X31;
1647
1648     /* These are selected by CONTEXT_CONTROL */
1649     ULONGLONG Sp;
1650     ULONGLONG Pc;
1651     ULONGLONG PState;
1652
1653     /* These are selected by CONTEXT_FLOATING_POINT */
1654     /* FIXME */
1655 } CONTEXT;
1656
1657 #endif /* __riscv64__ */

```

```

1658
1659
1660 #if !defined(CONTEXT_FULL) && !defined(RC_INVOKED)
1661 #error You need to define a CONTEXT for your CPU
1662 #endif
1663
1664 typedef CONTEXT *PCONTEXT;
1665
1666 #ifdef __WINE__
1667
1668 /* Macros for easier access to i386 context registers */
1669
1670 #define AX_reg(context)      (*(WORD*)&(context)->Eax)
1671 #define BX_reg(context)      (*(WORD*)&(context)->Ebx)
1672 #define CX_reg(context)      (*(WORD*)&(context)->Ecx)
1673 #define DX_reg(context)      (*(WORD*)&(context)->Edx)
1674 #define SI_reg(context)      (*(WORD*)&(context)->Esi)
1675 #define DI_reg(context)      (*(WORD*)&(context)->Edi)
1676 #define BP_reg(context)      (*(WORD*)&(context)->Ebp)
1677
1678 #define AL_reg(context)      (*(BYTE*)&(context)->Eax)
1679 #define AH_reg(context)      (*( (BYTE*)&(context)->Eax + 1))
1680 #define BL_reg(context)      (*(BYTE*)&(context)->Ebx)
1681 #define BH_reg(context)      (*( (BYTE*)&(context)->Ebx + 1))
1682 #define CL_reg(context)      (*(BYTE*)&(context)->Ecx)
1683 #define CH_reg(context)      (*( (BYTE*)&(context)->Ecx + 1))
1684 #define DL_reg(context)      (*(BYTE*)&(context)->Edx)
1685 #define DH_reg(context)      (*( (BYTE*)&(context)->Edx + 1))
1686
1687 #define SET_CFLAG(context)    ((context)->EFlags |= 0x0001)
1688 #define RESET_CFLAG(context) ((context)->EFlags &= ~0x0001)
1689 #define SET_ZFLAG(context)    ((context)->EFlags |= 0x0040)
1690 #define RESET_ZFLAG(context) ((context)->EFlags &= ~0x0040)
1691 #define ISV86(context)        ((context)->EFlags & 0x00020000)
1692
1693
1694 /* Macros to retrieve the current context */
1695
1696 #ifdef NEED_UNDERSCORE_PREFIX
1697 # define __ASM_NAME(name) "_" name
1698 #else
1699 # define __ASM_NAME(name) name
1700 #endif
1701
1702 #ifdef NEED_TYPE_IN_DEF
1703 # define __ASM_FUNC(name) ".def " __ASM_NAME(name) "; .scl 2; .type 32; .endef"
1704 #else
1705 # define __ASM_FUNC(name) ".type " __ASM_NAME(name) ",@function"
1706 #endif
1707
1708 #ifdef __GNUC__
1709 # define __ASM_GLOBAL_FUNC(name,code) \
1710 __asm__( ".align 4\n\t" \
1711          ".globl " __ASM_NAME(#name) "\n\t" \
1712          __ASM_FUNC(#name) "\n\t" \
1713          __ASM_NAME(#name) ":\n\t" \
1714          code );
1715 #else /* __GNUC__ */
1716 # define __ASM_GLOBAL_FUNC(name,code) \
1717 void __asm_dummy_##name(void) { \
1718 asm( ".align 4\n\t" \
1719      ".globl " __ASM_NAME(#name) "\n\t" \
1720      __ASM_FUNC(#name) "\n\t" \
1721      __ASM_NAME(#name) ":\n\t" \
1722      code ); \
1723 }
1724 #endif /* __GNUC__ */
1725
1726 #ifdef __i386__
1727
1728 #define _DEFINE_REGS_ENTRYPOINT( name, fn, args ) \
1729 __ASM_GLOBAL_FUNC( name, \
1730 "call " __ASM_NAME("__wine_call_from_32_regs") "\n\t" \
1731 ".long " __ASM_NAME(#fn) "\n\t" \
1732 ".byte " #args ", " #args )
1733 #define DEFINE_REGS_ENTRYPOINT_0( name, fn ) \
1734 extern void WINAPI name(void); \
1735 _DEFINE_REGS_ENTRYPOINT( name, fn, 0 )
1736 #define DEFINE_REGS_ENTRYPOINT_1( name, fn, t1 ) \
1737 extern void WINAPI name( t1 al ); \
1738 _DEFINE_REGS_ENTRYPOINT( name, fn, 4 )
1739 #define DEFINE_REGS_ENTRYPOINT_2( name, fn, t1, t2 ) \
1740 extern void WINAPI name( t1 al, t2 a2 ); \
1741 _DEFINE_REGS_ENTRYPOINT( name, fn, 8 )
1742 #define DEFINE_REGS_ENTRYPOINT_3( name, fn, t1, t2, t3 ) \
1743 extern void WINAPI name( t1 al, t2 a2, t3 a3 ); \
1744 _DEFINE_REGS_ENTRYPOINT( name, fn, 12 )

```

```

1745 #define DEFINE_REGS_ENTRYPOINT_4( name, fn, t1, t2, t3, t4 ) \
1746 extern void WINAPI name( t1 a1, t2 a2, t3 a3, t4 a4 ); \
1747 _DEFINE_REGS_ENTRYPOINT( name, fn, 16 )
1748
1749 #endif /* __i386__ */
1750
1751 #ifdef __sparc__
1752 /* FIXME: use getcontext() to retrieve full context */
1753 #define _GET_CONTEXT \
1754 CONTEXT context; \
1755 do { memset(&context, 0, sizeof(CONTEXT)); \
1756 context.ContextFlags = CONTEXT_CONTROL; \
1757 context.pc = (DWORD)__builtin_return_address(0); \
1758 } while (0)
1759
1760 #define DEFINE_REGS_ENTRYPOINT_0( name, fn ) \
1761 void WINAPI name ( void ) \
1762 { _GET_CONTEXT; fn( &context ); }
1763 #define DEFINE_REGS_ENTRYPOINT_1( name, fn, t1 ) \
1764 void WINAPI name ( t1 a1 ) \
1765 { _GET_CONTEXT; fn( a1, &context ); }
1766 #define DEFINE_REGS_ENTRYPOINT_2( name, fn, t1, t2 ) \
1767 void WINAPI name ( t1 a1, t2 a2 ) \
1768 { _GET_CONTEXT; fn( a1, a2, &context ); }
1769 #define DEFINE_REGS_ENTRYPOINT_3( name, fn, t1, t2, t3 ) \
1770 void WINAPI name ( t1 a1, t2 a2, t3 a3 ) \
1771 { _GET_CONTEXT; fn( a1, a2, a3, &context ); }
1772 #define DEFINE_REGS_ENTRYPOINT_4( name, fn, t1, t2, t3, t4 ) \
1773 void WINAPI name ( t1 a1, t2 a2, t3 a3, t4 a4 ) \
1774 { _GET_CONTEXT; fn( a1, a2, a3, a4, &context ); }
1775
1776 #endif /* __sparc__ */
1777
1778 #ifdef __s390__
1779 /* FIXME: use getcontext() to retrieve full context */
1780 #define _GET_CONTEXT \
1781 CONTEXT context; \
1782 do { memset(&context, 0, sizeof(CONTEXT)); \
1783 context.ContextFlags = CONTEXT_CONTROL; \
1784 context.pc = (DWORD)__builtin_return_address(0); \
1785 } while (0)
1786
1787 #define DEFINE_REGS_ENTRYPOINT_0( name, fn ) \
1788 void WINAPI name ( void ) \
1789 { _GET_CONTEXT; fn( &context ); }
1790 #define DEFINE_REGS_ENTRYPOINT_1( name, fn, t1 ) \
1791 void WINAPI name ( t1 a1 ) \
1792 { _GET_CONTEXT; fn( a1, &context ); }
1793 #define DEFINE_REGS_ENTRYPOINT_2( name, fn, t1, t2 ) \
1794 void WINAPI name ( t1 a1, t2 a2 ) \
1795 { _GET_CONTEXT; fn( a1, a2, &context ); }
1796 #define DEFINE_REGS_ENTRYPOINT_3( name, fn, t1, t2, t3 ) \
1797 void WINAPI name ( t1 a1, t2 a2, t3 a3 ) \
1798 { _GET_CONTEXT; fn( a1, a2, a3, &context ); }
1799 #define DEFINE_REGS_ENTRYPOINT_4( name, fn, t1, t2, t3, t4 ) \
1800 void WINAPI name ( t1 a1, t2 a2, t3 a3, t4 a4 ) \
1801 { _GET_CONTEXT; fn( a1, a2, a3, a4, &context ); }
1802
1803 #endif /* __s390__ */
1804
1805 #ifdef __PPC__
1806 /* FIXME: use getcontext() to retrieve full context */
1807 #define _GET_CONTEXT \
1808 CONTEXT context; \
1809 do { memset(&context, 0, sizeof(CONTEXT)); \
1810 context.ContextFlags = CONTEXT_CONTROL; \
1811 } while (0)
1812
1813 #define DEFINE_REGS_ENTRYPOINT_0( name, fn ) \
1814 void WINAPI name ( void ) \
1815 { _GET_CONTEXT; fn( &context ); }
1816 #define DEFINE_REGS_ENTRYPOINT_1( name, fn, t1 ) \
1817 void WINAPI name ( t1 a1 ) \
1818 { _GET_CONTEXT; fn( a1, &context ); }
1819 #define DEFINE_REGS_ENTRYPOINT_2( name, fn, t1, t2 ) \
1820 void WINAPI name ( t1 a1, t2 a2 ) \
1821 { _GET_CONTEXT; fn( a1, a2, &context ); }
1822 #define DEFINE_REGS_ENTRYPOINT_3( name, fn, t1, t2, t3 ) \
1823 void WINAPI name ( t1 a1, t2 a2, t3 a3 ) \
1824 { _GET_CONTEXT; fn( a1, a2, a3, &context ); }
1825 #define DEFINE_REGS_ENTRYPOINT_4( name, fn, t1, t2, t3, t4 ) \
1826 void WINAPI name ( t1 a1, t2 a2, t3 a3, t4 a4 ) \
1827 { _GET_CONTEXT; fn( a1, a2, a3, a4, &context ); }
1828
1829 #endif /* __PPC__ */
1830
1831

```

```

1832
1833 #ifndef DEFINE_REGS_ENTRYPOINT_0
1834 #error You need to define DEFINE_REGS_ENTRYPOINT macros for your CPU
1835 #endif
1836
1837 /* Constructor functions */
1838
1839 #ifdef __GNUC__
1840 # define DECL_GLOBAL_CONSTRUCTOR(func) \
1841 static void func(void) __attribute__((constructor)); \
1842 static void func(void)
1843 #else /* __GNUC__ */
1844 # ifdef __i386__
1845 #  define DECL_GLOBAL_CONSTRUCTOR(func) \
1846 static void __dummy_init_##func(void) { \
1847 asm(".section .init,\"ax\"\\n\\t\" \\"
1848      "call \" #func\" \\n\\t\" \\"
1849      ".previous"); } \
1850      static void func(void)
1851 # else /* __i386__ */
1852 #  error You must define the DECL_GLOBAL_CONSTRUCTOR macro for your platform
1853 # endif
1854 #endif /* __GNUC__ */
1855
1856 /* Segment register access */
1857
1858 #ifdef __i386__
1859 # ifdef __GNUC__
1860 #  define __DEFINE_GET_SEG(seg) \
1861 extern inline unsigned short __get_##seg(void) \
1862 { unsigned short res; __asm__("movw %%\" #seg\",%w0\" : \"=r\"(res)); return res; }
1863 #  define __DEFINE_SET_SEG(seg) \
1864 extern inline void __set_##seg(int val) { __asm__("movw %w0,%%\" #seg : : \"r\" (val)); }
1865 # else /* __GNUC__ */
1866 #  define __DEFINE_GET_SEG(seg) extern unsigned short __get_##seg(void);
1867 #  define __DEFINE_SET_SEG(seg) extern void __set_##seg(unsigned int);
1868 # endif /* __GNUC__ */
1869 #else /* __i386__ */
1870 # define __DEFINE_GET_SEG(seg) inline static unsigned short __get_##seg(void) { return 0; }
1871 # define __DEFINE_SET_SEG(seg) /* nothing */
1872 #endif /* __i386__ */
1873
1874 __DEFINE_GET_SEG(cs)
1875 __DEFINE_GET_SEG(ds)
1876 __DEFINE_GET_SEG(es)
1877 __DEFINE_GET_SEG(fs)
1878 __DEFINE_GET_SEG(gs)
1879 __DEFINE_GET_SEG(ss)
1880 __DEFINE_SET_SEG(fs)
1881 __DEFINE_SET_SEG(gs)
1882 #undef __DEFINE_GET_SEG
1883 #undef __DEFINE_SET_SEG
1884
1885 #endif /* __WINE__ */
1886
1887
1888
1889 /*
1890 * Language IDs
1891 */
1892
1893 #define MAKELCID(l, s)      (MAKELONG(l, s))
1894
1895 #define MAKELANGID(p, s)    (((WORD)(s))<10) | (WORD)(p))
1896 #define PRIMARYLANGID(l)   ((WORD)(l) & 0x3ff)
1897 #define SUBLANGID(l)       ((WORD)(l) > 10)
1898
1899 #define LANGIDFROMLCID(lcid) ((WORD)(lcid))
1900 #define SORTIDFROMLCID(lcid) ((WORD)(((DWORD)(lcid)) >> 16) & 0x0f))
1901
1902 #define LANG_SYSTEM_DEFAULT (MAKELANGID(LANG_NEUTRAL, SUBLANG_SYS_DEFAULT))
1903 #define LANG_USER_DEFAULT   (MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT))
1904 #define LOCALE_SYSTEM_DEFAULT (MAKELCID(LANG_SYSTEM_DEFAULT, SORT_DEFAULT))
1905 #define LOCALE_USER_DEFAULT  (MAKELCID(LANG_USER_DEFAULT, SORT_DEFAULT))
1906 #define LOCALE_NEUTRAL       (MAKELCID(MAKELANGID(LANG_NEUTRAL, SUBLANG_NEUTRAL), SORT_DEFAULT))
1907
1908 /* FIXME: are the symbolic names correct for LIDs: 0x17, 0x20, 0x28,
1909 * 0x2a, 0x2b, 0x2c, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35,
1910 * 0x37, 0x39, 0x3a, 0x3b, 0x3c, 0x3e, 0x3f, 0x41, 0x43, 0x44,
1911 * 0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e,
1912 * 0x4f, 0x57
1913 */
1914 #define LANG_NEUTRAL          0x00
1915 #define LANG_Afrikaans        0x36
1916 #define LANG_Albanian         0x1c
1917 #define LANG_Arabic           0x01
1918 #define LANG_Armenian         0x2b

```

```

1919 #define LANG_ASSAMESE          0x4d
1920 #define LANG_AZERI              0x2c
1921 #define LANG_BASQUE             0x2d
1922 #define LANG_BENGALI            0x45
1923 #define LANG_BULGARIAN          0x02
1924 #define LANG_BYELORUSSIAN       0x23
1925 #define LANG_CATALAN            0x03
1926 #define LANG_CHINESE            0x04
1927 #define LANG_SERBO_CROATIAN      0x1a
1928 #define LANG_CROATIAN           LANG_SERBO_CROATIAN
1929 #define LANG_SERBIAN             LANG_SERBO_CROATIAN
1930 #define LANG_CZECH              0x05
1931 #define LANG_DANISH             0x06
1932 #define LANG_DUTCH              0x13
1933 #define LANG_ENGLISH            0x09
1934 #define LANG_ESTONIAN           0x25
1935 #define LANG_FAEROESE           0x38
1936 #define LANG_FARSI             0x29
1937 #define LANG_FINNISH            0x0b
1938 #define LANG_FRENCH             0x0c
1939 #define LANG_GAELIC             0x3c
1940 #define LANG_GEORGIAN           0x37
1941 #define LANG_GERMAN             0x07
1942 #define LANG_GREEK              0x08
1943 #define LANG_GUJARATI           0x47
1944 #define LANG_HEBREW             0x0D
1945 #define LANG_HINDI              0x39
1946 #define LANG_HUNGARIAN          0x0e
1947 #define LANG_ICELANDIC          0x0f
1948 #define LANG_INDONESIAN         0x21
1949 #define LANG_ITALIAN            0x10
1950 #define LANG_JAPANESE           0x11
1951 #define LANG_KANNADA            0x4b
1952 #define LANG_KAZAKH            0x3f
1953 #define LANG_KONKANI            0x57
1954 #define LANG_KOREAN             0x12
1955 #define LANG_LATVIAN            0x26
1956 #define LANG_LITHUANIAN         0x27
1957 #define LANG_MACEDONIAN         0x2f
1958 #define LANG_MALAY              0x3e
1959 #define LANG_MALAYALAM          0x4c
1960 #define LANG_MALTESE            0x3a
1961 #define LANG_MAORI              0x28
1962 #define LANG_MARATHI            0x4e
1963 #define LANG_NORWEGIAN          0x14
1964 #define LANG_ORIYA              0x48
1965 #define LANG_POLISH             0x15
1966 #define LANG_PORTUGUESE         0x16
1967 #define LANG_PUNJABI            0x46
1968 #define LANG_RHAETO_ROMANCE     0x17
1969 #define LANG_ROMANIAN           0x18
1970 #define LANG_RUSSIAN            0x19
1971 #define LANG_SAAMI              0x3b
1972 #define LANG_SANSKRIT           0x4f
1973 #define LANG_SLOVAK             0x1b
1974 #define LANG_SLOVENIAN          0x24
1975 #define LANG_SORBIAN            0x2e
1976 #define LANG_SPANISH            0x0a
1977 #define LANG_SUTU               0x30
1978 #define LANG_SWAHILI            0x41
1979 #define LANG_SWEDISH            0x1d
1980 #define LANG_TAMIL              0x49
1981 #define LANG_TATAR              0x44
1982 #define LANG_TELUGU             0x4a
1983 #define LANG_THAI               0x1e
1984 #define LANG_TSONGA             0x31
1985 #define LANG_TSWANA             0x32
1986 #define LANG_TURKISH            0x1f
1987 #define LANG_UKRAINIAN          0x22
1988 #define LANG_URDU               0x20
1989 #define LANG_UZBEK              0x43
1990 #define LANG_VENDA              0x33
1991 #define LANG_VIETNAMESE         0x2a
1992 #define LANG_XHOSA              0x34
1993 #define LANG_ZULU               0x35
1994 /* non standard; keep the number high enough (but < 0xff) */
1995 #define LANG_ESPERANTO           0x8f
1996 #define LANG_WALON               0x90
1997 #define LANG_CORNISH             0x91
1998 #define LANG_WELSH              0x92
1999 #define LANG_BRETON              0x93
2000
2001 /* Sublanguage definitions */
2002 #define SUBLANG_NEUTRAL           0x00 /* language neutral */
2003 #define SUBLANG_DEFAULT          0x01 /* user default */
2004 #define SUBLANG_SYS_DEFAULT      0x02 /* system default */
2005

```

```
2006 #define SUBLANG_ARABIC 0x01
2007 #define SUBLANG_ARABIC_SAUDI_ARABIA 0x01
2008 #define SUBLANG_ARABIC_IRAQ 0x02
2009 #define SUBLANG_ARABIC_EGYPT 0x03
2010 #define SUBLANG_ARABIC_LIBYA 0x04
2011 #define SUBLANG_ARABIC_ALGERIA 0x05
2012 #define SUBLANG_ARABIC_MOROCCO 0x06
2013 #define SUBLANG_ARABIC_TUNISIA 0x07
2014 #define SUBLANG_ARABIC_OMAN 0x08
2015 #define SUBLANG_ARABIC_YEMEN 0x09
2016 #define SUBLANG_ARABIC_SYRIA 0x0a
2017 #define SUBLANG_ARABIC_JORDAN 0x0b
2018 #define SUBLANG_ARABIC_LEBANON 0x0c
2019 #define SUBLANG_ARABIC_KUWAIT 0x0d
2020 #define SUBLANG_ARABIC_UAE 0x0e
2021 #define SUBLANG_ARABIC_BAHRAIN 0x0f
2022 #define SUBLANG_ARABIC_QATAR 0x10
2023 #define SUBLANG_CHINESE_TRADITIONAL 0x01
2024 #define SUBLANG_CHINESE_SIMPLIFIED 0x02
2025 #define SUBLANG_CHINESE_HONGKONG 0x03
2026 #define SUBLANG_CHINESE_SINGAPORE 0x04
2027 #define SUBLANG_CHINESE_MACAU 0x05
2028 #define SUBLANG_DUTCH 0x01
2029 #define SUBLANG_DUTCH_BELGIAN 0x02
2030 #define SUBLANG_DUTCH_SURINAM 0x03
2031 #define SUBLANG_ENGLISH_US 0x01
2032 #define SUBLANG_ENGLISH_UK 0x02
2033 #define SUBLANG_ENGLISH_AUS 0x03
2034 #define SUBLANG_ENGLISH_CAN 0x04
2035 #define SUBLANG_ENGLISH_NZ 0x05
2036 #define SUBLANG_ENGLISH_EIRE 0x06
2037 #define SUBLANG_ENGLISH_SAFRICA 0x07
2038 #define SUBLANG_ENGLISH_JAMAICA 0x08
2039 #define SUBLANG_ENGLISH_CARRIBEAN 0x09
2040 #define SUBLANG_ENGLISH_BELIZE 0x0a
2041 #define SUBLANG_ENGLISH_TRINIDAD 0x0b
2042 #define SUBLANG_ENGLISH_ZIMBABWE 0x0c
2043 #define SUBLANG_ENGLISH_PHILIPPINES 0x0d
2044 #define SUBLANG_FRENCH 0x01
2045 #define SUBLANG_FRENCH_BELGIAN 0x02
2046 #define SUBLANG_FRENCH_CANADIAN 0x03
2047 #define SUBLANG_FRENCH_SWISS 0x04
2048 #define SUBLANG_FRENCH_LUXEMBOURG 0x05
2049 #define SUBLANG_FRENCH_MONACO 0x06
2050 #define SUBLANG_GERMAN 0x01
2051 #define SUBLANG_GERMAN_SWISS 0x02
2052 #define SUBLANG_GERMAN_AUSTRIAN 0x03
2053 #define SUBLANG_GERMAN_LUXEMBOURG 0x04
2054 #define SUBLANG_GERMAN_LIECHTENSTEIN 0x05
2055 #define SUBLANG_ITALIAN 0x01
2056 #define SUBLANG_ITALIAN_SWISS 0x02
2057 #define SUBLANG_KOREAN 0x01
2058 #define SUBLANG_KOREAN_JOHAB 0x02
2059 #define SUBLANG_NORWEGIAN_BOKMAL 0x01
2060 #define SUBLANG_NORWEGIAN_NYNORSK 0x02
2061 #define SUBLANG_PORTUGUESE 0x02
2062 #define SUBLANG_PORTUGUESE_BRAZILIAN 0x01
2063 #define SUBLANG_SPANISH 0x01
2064 #define SUBLANG_SPANISH_MEXICAN 0x02
2065 #define SUBLANG_SPANISH_MODERN 0x03
2066 #define SUBLANG_SPANISH_GUATEMALA 0x04
2067 #define SUBLANG_SPANISH_COSTARICA 0x05
2068 #define SUBLANG_SPANISH_PANAMA 0x06
2069 #define SUBLANG_SPANISH_DOMINICAN 0x07
2070 #define SUBLANG_SPANISH_VENEZUELA 0x08
2071 #define SUBLANG_SPANISH_COLOMBIA 0x09
2072 #define SUBLANG_SPANISH_PERU 0x0a
2073 #define SUBLANG_SPANISH_ARGENTINA 0x0b
2074 #define SUBLANG_SPANISH_ECUADOR 0x0c
2075 #define SUBLANG_SPANISH_CHILE 0x0d
2076 #define SUBLANG_SPANISH_URUGUAY 0x0e
2077 #define SUBLANG_SPANISH_PARAGUAY 0x0f
2078 #define SUBLANG_SPANISH_BOLIVIA 0x10
2079 #define SUBLANG_SPANISH_EL_SALVADOR 0x11
2080 #define SUBLANG_SPANISH_HONDURAS 0x12
2081 #define SUBLANG_SPANISH_NICARAGUA 0x13
2082 #define SUBLANG_SPANISH_PUERTO_RICO 0x14
2083 /* FIXME: I don't know the symbolic names for those */
2084 #define SUBLANG_ROMANIAN 0x01
2085 #define SUBLANG_ROMANIAN_MOLDAVIA 0x02
2086 #define SUBLANG_RUSSIAN 0x01
2087 #define SUBLANG_RUSSIAN_MOLDAVIA 0x02
2088 #define SUBLANG_CROATIAN 0x01
2089 #define SUBLANG_SERBIAN 0x02
2090 #define SUBLANG_SERBIAN_LATIN 0x03
2091 #define SUBLANG_SWEDISH 0x01
2092 #define SUBLANG_SWEDISH_FINLAND 0x02
```

```
2093 #define SUBLANG_LITHUANIAN 0x01
2094 #define SUBLANG_LITHUANIAN_CLASSIC 0x02
2095 #define SUBLANG_AZERI 0x01
2096 #define SUBLANG_AZERI_CYRILLIC 0x02
2097 #define SUBLANG_GAELIC 0x01
2098 #define SUBLANG_GAELIC_SCOTTISH 0x02
2099 #define SUBLANG_GAELIC_MANX 0x03
2100 #define SUBLANG_MALAY 0x01
2101 #define SUBLANG_MALAY_BRUNEI_DARUSSALAM 0x02
2102 #define SUBLANG_UZBEK 0x01
2103 #define SUBLANG_UZBEK_CYRILLIC 0x02
2104 #define SUBLANG_URDU_PAKISTAN 0x01
2105
2106
2107
2108 /*
2109  * Sort definitions
2110  */
2111
2112 #define SORT_DEFAULT 0x0
2113 #define SORT_JAPANESE_XJIS 0x0
2114 #define SORT_JAPANESE_UNICODE 0x1
2115 #define SORT_CHINESE_BIG5 0x0
2116 #define SORT_CHINESE_UNICODE 0x1
2117 #define SORT_KOREAN_KSC 0x0
2118 #define SORT_KOREAN_UNICODE 0x1
2119
2120
2121
2122 /*
2123  * Definitions for IsTextUnicode()
2124  */
2125
2126 #define IS_TEXT_UNICODE_ASCII16 0x0001
2127 #define IS_TEXT_UNICODE_STATISTICS 0x0002
2128 #define IS_TEXT_UNICODE_CONTROLS 0x0004
2129 #define IS_TEXT_UNICODE_SIGNATURE 0x0008
2130 #define IS_TEXT_UNICODE_UNICODE_MASK 0x000F
2131 #define IS_TEXT_UNICODE_REVERSE_ASCII16 0x0010
2132 #define IS_TEXT_UNICODE_REVERSE_STATISTICS 0x0020
2133 #define IS_TEXT_UNICODE_REVERSE_CONTROLS 0x0040
2134 #define IS_TEXT_UNICODE_REVERSE_SIGNATURE 0x0080
2135 #define IS_TEXT_UNICODE_REVERSE_MASK 0x00F0
2136 #define IS_TEXT_UNICODE_ILLEGAL_CHARS 0x0100
2137 #define IS_TEXT_UNICODE_ODD_LENGTH 0x0200
2138 #define IS_TEXT_UNICODE_DBCS_LEADBYTE 0x0400
2139 #define IS_TEXT_UNICODE_NOT_UNICODE_MASK 0x0F00
2140 #define IS_TEXT_UNICODE_NULL_BYTES 0x1000
2141 #define IS_TEXT_UNICODE_NOT_ASCII_MASK 0xF000
2142
2143
2144
2145 /*
2146  * Exception codes
2147  */
2148
2149 #define STATUS_SUCCESS 0x00000000
2150 #define STATUS_WAIT_0 0x00000000
2151 #define STATUS_ABANDONED_WAIT_0 0x00000080
2152 #define STATUS_ABANDONED_WAIT_63 0x000000BF
2153 #define STATUS_USER_APC 0x000000C0
2154 #define STATUS_ALERTED 0x00000101
2155 #define STATUS_TIMEOUT 0x00000102
2156 #define STATUS_PENDING 0x00000103
2157 #define STATUS_REPARSE 0x00000104
2158 #define STATUS_MORE_ENTRIES 0x00000105
2159 #define STATUS_NOT_ALL_ASSIGNED 0x00000106
2160 #define STATUS_SOME_NOT_MAPPED 0x00000107
2161 #define STATUS_OPLOCK_BREAK_IN_PROGRESS 0x00000108
2162 #define STATUS_VOLUME_MOUNTED 0x00000109
2163 #define STATUS_RXACT_COMMITTED 0x0000010A
2164 #define STATUS_NOTIFY_CLEANUP 0x0000010B
2165 #define STATUS_NOTIFY_ENUM_DIR 0x0000010C
2166 #define STATUS_NO_QUOTAS_FOR_ACCOUNT 0x0000010D
2167 #define STATUS_PRIMARY_TRANSPORT_CONNECT_FAILED 0x0000010E
2168 #define STATUS_PAGE_FAULT_TRANSITION 0x00000110
2169 #define STATUS_PAGE_FAULT_DEMAND_ZERO 0x00000111
2170 #define STATUS_PAGE_FAULT_COPY_ON_WRITE 0x00000112
2171 #define STATUS_PAGE_FAULT_GUARD_PAGE 0x00000113
2172 #define STATUS_PAGE_FAULT_PAGING_FILE 0x00000114
2173 #define STATUS_CACHE_PAGE_LOCKED 0x00000115
2174 #define STATUS_CRASH_DUMP 0x00000116
2175 #define STATUS_BUFFER_ALL_ZEROS 0x00000117
2176 #define STATUS_REPARSE_OBJECT 0x00000118
2177
2178 #define STATUS_THREAD_WAS_SUSPENDED 0x40000001
2179 #define STATUS_WORKING_SET_LIMIT_RANGE 0x40000002
```

```
2180 #define STATUS_IMAGE_NOT_AT_BASE 0x40000003
2181 #define STATUS_RXACT_STATE_CREATED 0x40000004
2182 #define STATUS_SEGMENT_NOTIFICATION 0x40000005
2183 #define STATUS_LOCAL_USER_SESSION_KEY 0x40000006
2184 #define STATUS_BAD_CURRENT_DIRECTORY 0x40000007
2185 #define STATUS_SERIAL_MORE_WRITES 0x40000008
2186 #define STATUS_REGISTRY_RECOVERED 0x40000009
2187 #define STATUS_FT_READ_RECOVERY_FROM_BACKUP 0x4000000A
2188 #define STATUS_FT_WRITE_RECOVERY 0x4000000B
2189 #define STATUS_SERIAL_COUNTER_TIMEOUT 0x4000000C
2190 #define STATUS_NULL_IM_PASSWORD 0x4000000D
2191 #define STATUS_IMAGE_MACHINE_TYPE_MISMATCH 0x4000000E
2192 #define STATUS_RECEIVE_PARTIAL 0x4000000F
2193 #define STATUS_RECEIVE_EXPEDITED 0x40000010
2194 #define STATUS_RECEIVE_PARTIAL_EXPEDITED 0x40000011
2195 #define STATUS_EVENT_DONE 0x40000012
2196 #define STATUS_EVENT_PENDING 0x40000013
2197 #define STATUS_CHECKING_FILE_SYSTEM 0x40000014
2198 #define STATUS_FATAL_APP_EXIT 0x40000015
2199 #define STATUS_PREDEFINED_HANDLE 0x40000016
2200 #define STATUS_WAS_UNLOCKED 0x40000017
2201 #define STATUS_SERVICE_NOTIFICATION 0x40000018
2202 #define STATUS_WAS_LOCKED 0x40000019
2203 #define STATUS_LOG_HARD_ERROR 0x4000001A
2204 #define STATUS_ALREADY_WIN32 0x4000001B
2205 #define STATUS_WX86_UNSIMULATE 0x4000001C
2206 #define STATUS_WX86_CONTINUE 0x4000001D
2207 #define STATUS_WX86_SINGLE_STEP 0x4000001E
2208 #define STATUS_WX86_BREAKPOINT 0x4000001F
2209 #define STATUS_WX86_EXCEPTION_CONTINUE 0x40000020
2210 #define STATUS_WX86_EXCEPTION_LASTCHANCE 0x40000021
2211 #define STATUS_WX86_EXCEPTION_CHAIN 0x40000022
2212 #define STATUS_IMAGE_MACHINE_TYPE_MISMATCH_EXE 0x40000023
2213 #define STATUS_NO_YIELD_PERFORMED 0x40000024
2214 #define STATUS_TIMER_RESUME_IGNORED 0x40000025
2215
2216 #define STATUS_GUARD_PAGE_VIOLATION 0x80000001
2217 #define STATUS_DATATYPE_MISALIGNMENT 0x80000002
2218 #define STATUS_BREAKPOINT 0x80000003
2219 #define STATUS_SINGLE_STEP 0x80000004
2220 #define STATUS_BUFFER_OVERFLOW 0x80000005
2221 #define STATUS_NO_MORE_FILES 0x80000006
2222 #define STATUS_WAKE_SYSTEM_DEBUGGER 0x80000007
2223
2224 #define STATUS_HANDLES_CLOSED 0x8000000A
2225 #define STATUS_NO_INHERITANCE 0x8000000B
2226 #define STATUS_GUID_SUBSTITUTION_MADE 0x8000000C
2227 #define STATUS_PARTIAL_COPY 0x8000000D
2228 #define STATUS_DEVICE_PAPER_EMPTY 0x8000000E
2229 #define STATUS_DEVICE_POWERED_OFF 0x8000000F
2230 #define STATUS_DEVICE_OFF_LINE 0x80000010
2231 #define STATUS_DEVICE_BUSY 0x80000011
2232 #define STATUS_NO_MORE_EAS 0x80000012
2233 #define STATUS_INVALID_EA_NAME 0x80000013
2234 #define STATUS_EA_LIST_INCONSISTENT 0x80000014
2235 #define STATUS_INVALID_EA_FLAG 0x80000015
2236 #define STATUS_VERIFY_REQUIRED 0x80000016
2237 #define STATUS_EXTRANEIOUS_INFORMATION 0x80000017
2238 #define STATUS_RXACT_COMMIT_NECESSARY 0x80000018
2239 #define STATUS_NO_MORE_ENTRIES 0x8000001A
2240 #define STATUS_FILEMARK_DETECTED 0x8000001B
2241 #define STATUS_MEDIA_CHANGED 0x8000001C
2242 #define STATUS_BUS_RESET 0x8000001D
2243 #define STATUS_END_OF_MEDIA 0x8000001E
2244 #define STATUS_BEGINNING_OF_MEDIA 0x8000001F
2245 #define STATUS_MEDIA_CHECK 0x80000020
2246 #define STATUS_SETMARK_DETECTED 0x80000021
2247 #define STATUS_NO_DATA_DETECTED 0x80000022
2248 #define STATUS_REDIRECTOR_HAS_OPEN_HANDLES 0x80000023
2249 #define STATUS_SERVER_HAS_OPEN_HANDLES 0x80000024
2250 #define STATUS_ALREADY_DISCONNECTED 0x80000025
2251 #define STATUS_LONGJUMP 0x80000026
2252
2253 #define STATUS_UNSUCCESSFUL 0xC0000001
2254 #define STATUS_NOT_IMPLEMENTED 0xC0000002
2255 #define STATUS_INVALID_INFO_CLASS 0xC0000003
2256 #define STATUS_INFO_LENGTH_MISMATCH 0xC0000004
2257 #define STATUS_ACCESS_VIOLATION 0xC0000005
2258 #define STATUS_IN_PAGE_ERROR 0xC0000006
2259 #define STATUS_PAGEFILE_QUOTA 0xC0000007
2260 #define STATUS_INVALID_HANDLE 0xC0000008
2261 #define STATUS_BAD_INITIAL_STACK 0xC0000009
2262 #define STATUS_BAD_INITIAL_PC 0xC000000A
2263 #define STATUS_INVALID_CID 0xC000000B
2264 #define STATUS_TIMER_NOT_CANCELED 0xC000000C
2265 #define STATUS_INVALID_PARAMETER 0xC000000D
2266 #define STATUS_NO_SUCH_DEVICE 0xC000000E
```



```
2267 #define STATUS_NO_SUCH_FILE 0xC000000F
2268 #define STATUS_INVALID_DEVICE_REQUEST 0xC0000010
2269 #define STATUS_END_OF_FILE 0xC0000011
2270 #define STATUS_WRONG_VOLUME 0xC0000012
2271 #define STATUS_NO_MEDIA_IN_DEVICE 0xC0000013
2272 #define STATUS_UNRECOGNIZED_MEDIA 0xC0000014
2273 #define STATUS_NONEXISTENT_SECTOR 0xC0000015
2274 #define STATUS_MORE_PROCESSING_REQUIRED 0xC0000016
2275 #define STATUS_NO_MEMORY 0xC0000017
2276 #define STATUS_CONFLICTING_ADDRESSES 0xC0000018
2277 #define STATUS_NOT_MAPPED_VIEW 0xC0000019
2278 #define STATUS_UNABLE_TO_FREE_VM 0xC000001A
2279 #define STATUS_UNABLE_TO_DELETE_SECTION 0xC000001B
2280 #define STATUS_INVALID_SYSTEM_SERVICE 0xC000001C
2281 #define STATUS_ILLEGAL_INSTRUCTION 0xC000001D
2282 #define STATUS_INVALID_LOCK_SEQUENCE 0xC000001E
2283 #define STATUS_INVALID_VIEW_SIZE 0xC000001F
2284 #define STATUS_INVALID_FILE_FOR_SECTION 0xC0000020
2285 #define STATUS_ALREADY_COMMITTED 0xC0000021
2286 #define STATUS_ACCESS_DENIED 0xC0000022
2287 #define STATUS_BUFFER_TOO_SMALL 0xC0000023
2288 #define STATUS_OBJECT_TYPE_MISMATCH 0xC0000024
2289 #define STATUS_NONCONTINUABLE_EXCEPTION 0xC0000025
2290 #define STATUS_INVALID_DISPOSITION 0xC0000026
2291 #define STATUS_UNWIND 0xC0000027
2292 #define STATUS_BAD_STACK 0xC0000028
2293 #define STATUS_INVALID_UNWIND_TARGET 0xC0000029
2294 #define STATUS_NOT_LOCKED 0xC000002A
2295 #define STATUS_PARITY_ERROR 0xC000002B
2296 #define STATUS_UNABLE_TO_DECOMMIT_VM 0xC000002C
2297 #define STATUS_NOT_COMMITTED 0xC000002D
2298 #define STATUS_INVALID_PORT_ATTRIBUTES 0xC000002E
2299 #define STATUS_PORT_MESSAGE_TOO_LONG 0xC000002F
2300 #define STATUS_INVALID_PARAMETER_MIX 0xC0000030
2301 #define STATUS_INVALID_QUOTA_LOWER 0xC0000031
2302 #define STATUS_DISK_CORRUPT_ERROR 0xC0000032
2303 #define STATUS_OBJECT_NAME_INVALID 0xC0000033
2304 #define STATUS_OBJECT_NAME_NOT_FOUND 0xC0000034
2305 #define STATUS_OBJECT_NAME_COLLISION 0xC0000035
2306 #define STATUS_PORT_DISCONNECTED 0xC0000037
2307 #define STATUS_DEVICE_ALREADY_ATTACHED 0xC0000038
2308 #define STATUS_OBJECT_PATH_INVALID 0xC0000039
2309 #define STATUS_OBJECT_PATH_NOT_FOUND 0xC000003A
2310 #define STATUS_PATH_SYNTAX_BAD 0xC000003B
2311 #define STATUS_DATA_OVERRUN 0xC000003C
2312 #define STATUS_DATA_LATE_ERROR 0xC000003D
2313 #define STATUS_DATA_ERROR 0xC000003E
2314 #define STATUS_CRC_ERROR 0xC000003F
2315 #define STATUS_SECTION_TOO_BIG 0xC0000040
2316 #define STATUS_PORT_CONNECTION_REFUSED 0xC0000041
2317 #define STATUS_INVALID_PORT_HANDLE 0xC0000042
2318 #define STATUS_SHARING_VIOLATION 0xC0000043
2319 #define STATUS_QUOTA_EXCEEDED 0xC0000044
2320 #define STATUS_INVALID_PAGE_PROTECTION 0xC0000045
2321 #define STATUS_MUTANT_NOT_OWNED 0xC0000046
2322 #define STATUS_SEMAPHORE_LIMIT_EXCEEDED 0xC0000047
2323 #define STATUS_PORT_ALREADY_SET 0xC0000048
2324 #define STATUS_SECTION_NOT_IMAGE 0xC0000049
2325 #define STATUS_SUSPEND_COUNT_EXCEEDED 0xC000004A
2326 #define STATUS_THREAD_IS_TERMINATING 0xC000004B
2327 #define STATUS_BAD_WORKING_SET_LIMIT 0xC000004C
2328 #define STATUS_INCOMPATIBLE_FILE_MAP 0xC000004D
2329 #define STATUS_SECTION_PROTECTION 0xC000004E
2330 #define STATUS_EAS_NOT_SUPPORTED 0xC000004F
2331 #define STATUS_EA_TOO_LARGE 0xC0000050
2332 #define STATUS_NONEXISTENT_EA_ENTRY 0xC0000051
2333 #define STATUS_NO_EAS_ON_FILE 0xC0000052
2334 #define STATUS_EA_CORRUPT_ERROR 0xC0000053
2335 #define STATUS_LOCK_NOT_GRANTED 0xC0000054 /* FIXME: not sure */
2336 #define STATUS_FILE_LOCK_CONFLICT 0xC0000055 /* FIXME: not sure */
2337 #define STATUS_DELETE_PENDING 0xC0000056
2338 #define STATUS_CTL_FILE_NOT_SUPPORTED 0xC0000057
2339 #define STATUS_UNKNOWN_REVISION 0xC0000058
2340 #define STATUS_REVISION_MISMATCH 0xC0000059
2341 #define STATUS_INVALID_OWNER 0xC000005A
2342 #define STATUS_INVALID_PRIMARY_GROUP 0xC000005B
2343 #define STATUS_NO_IMPERSONATION_TOKEN 0xC000005C
2344 #define STATUS_CANT_DISABLE_MANDATORY 0xC000005D
2345 #define STATUS_NO_LOGON_SERVERS 0xC000005E
2346 #define STATUS_NO_SUCH_LOGON_SESSION 0xC000005F
2347 #define STATUS_NO_SUCH_PRIVILEGE 0xC0000060
2348 #define STATUS_PRIVILEGE_NOT_HELD 0xC0000061
2349 #define STATUS_INVALID_ACCOUNT_NAME 0xC0000062
2350 #define STATUS_USER_EXISTS 0xC0000063
2351 #define STATUS_NO_SUCH_USER 0xC0000064
2352 #define STATUS_GROUP_EXISTS 0xC0000065
2353 #define STATUS_NO_SUCH_GROUP 0xC0000066
```

```
2354 #define STATUS_MEMBER_IN_GROUP 0xC0000067
2355 #define STATUS_MEMBER_NOT_IN_GROUP 0xC0000068
2356 #define STATUS_LAST_ADMIN 0xC0000069
2357 #define STATUS_WRONG_PASSWORD 0xC000006A
2358 #define STATUS_ILL_FORMED_PASSWORD 0xC000006B
2359 #define STATUS_PASSWORD_RESTRICTION 0xC000006C
2360 #define STATUS_LOGON_FAILURE 0xC000006D
2361 #define STATUS_ACCOUNT_RESTRICTION 0xC000006E
2362 #define STATUS_INVALID_LOGON_HOURS 0xC000006F
2363 #define STATUS_INVALID_WORKSTATION 0xC0000070
2364 #define STATUS_PASSWORD_EXPIRED 0xC0000071
2365 #define STATUS_ACCOUNT_DISABLED 0xC0000072
2366 #define STATUS_NONE_MAPPED 0xC0000073
2367 #define STATUS_TOO_MANY_LUIDS_REQUESTED 0xC0000074
2368 #define STATUS_LUIDS_EXHAUSTED 0xC0000075
2369 #define STATUS_INVALID_SUB_AUTHORITY 0xC0000076
2370 #define STATUS_INVALID_ACL 0xC0000077
2371 #define STATUS_INVALID_SID 0xC0000078
2372 #define STATUS_INVALID_SECURITY_DESCR 0xC0000079
2373 #define STATUS_PROCEDURE_NOT_FOUND 0xC000007A
2374 #define STATUS_INVALID_IMAGE_FORMAT 0xC000007B
2375 #define STATUS_NO_TOKEN 0xC000007C
2376 #define STATUS_BAD_INHERITANCE_ACL 0xC000007D
2377 #define STATUS_RANGE_NOT_LOCKED 0xC000007E
2378 #define STATUS_DISK_FULL 0xC000007F
2379 #define STATUS_SERVER_DISABLED 0xC0000080
2380 #define STATUS_SERVER_NOT_DISABLED 0xC0000081
2381 #define STATUS_TOO_MANY_GUIDS_REQUESTED 0xC0000082
2382 #define STATUS_GUIDS_EXHAUSTED 0xC0000083
2383 #define STATUS_INVALID_ID_AUTHORITY 0xC0000084
2384 #define STATUS_AGENTS_EXHAUSTED 0xC0000085
2385 #define STATUS_INVALID_VOLUME_LABEL 0xC0000086
2386 #define STATUS_SECTION_NOT_EXTENDED 0xC0000087
2387 #define STATUS_NOT_MAPPED_DATA 0xC0000088
2388 #define STATUS_RESOURCE_DATA_NOT_FOUND 0xC0000089
2389 #define STATUS_RESOURCE_TYPE_NOT_FOUND 0xC000008A
2390 #define STATUS_RESOURCE_NAME_NOT_FOUND 0xC000008B
2391 #define STATUS_ARRAY_BOUNDS_EXCEEDED 0xC000008C
2392 #define STATUS_FLOAT_DENORMAL_OPERAND 0xC000008D
2393 #define STATUS_FLOAT_DIVIDE_BY_ZERO 0xC000008E
2394 #define STATUS_FLOAT_INEXACT_RESULT 0xC000008F
2395 #define STATUS_FLOAT_INVALID_OPERATION 0xC0000090
2396 #define STATUS_FLOAT_OVERFLOW 0xC0000091
2397 #define STATUS_FLOAT_STACK_CHECK 0xC0000092
2398 #define STATUS_FLOAT_UNDERFLOW 0xC0000093
2399 #define STATUS_INTEGER_DIVIDE_BY_ZERO 0xC0000094
2400 #define STATUS_INTEGER_OVERFLOW 0xC0000095
2401 #define STATUS_PRIVILEGED_INSTRUCTION 0xC0000096
2402 #define STATUS_TOO_MANY_PAGING_FILES 0xC0000097
2403 #define STATUS_FILE_INVALID 0xC0000098
2404 #define STATUS_ALLOTTED_SPACE_EXCEEDED 0xC0000099
2405 #define STATUS_INSUFFICIENT_RESOURCES 0xC000009A
2406 #define STATUS_DFS_EXIT_PATH_FOUND 0xC000009B
2407 #define STATUS_DEVICE_DATA_ERROR 0xC000009C
2408 #define STATUS_DEVICE_NOT_CONNECTED 0xC000009D
2409 #define STATUS_DEVICE_POWER_FAILURE 0xC000009E
2410 #define STATUS_FREE_VM_NOT_AT_BASE 0xC000009F
2411 #define STATUS_MEMORY_NOT_ALLOCATED 0xC00000A0
2412 #define STATUS_WORKING_SET_QUOTA 0xC00000A1
2413 #define STATUS_MEDIA_WRITE_PROTECTED 0xC00000A2
2414 #define STATUS_DEVICE_NOT_READY 0xC00000A3
2415 #define STATUS_INVALID_GROUP_ATTRIBUTES 0xC00000A4
2416 #define STATUS_BAD_IMPERSONATION_LEVEL 0xC00000A5
2417 #define STATUS_CANT_OPEN_ANONYMOUS 0xC00000A6
2418 #define STATUS_BAD_VALIDATION_CLASS 0xC00000A7
2419 #define STATUS_BAD_TOKEN_TYPE 0xC00000A8
2420 #define STATUS_BAD_MASTER_BOOT_RECORD 0xC00000A9
2421 #define STATUS_INSTRUCTION_MISALIGNMENT 0xC00000AA
2422 #define STATUS_INSTANCE_NOT_AVAILABLE 0xC00000AB
2423 #define STATUS_PIPE_NOT_AVAILABLE 0xC00000AC
2424 #define STATUS_INVALID_PIPE_STATE 0xC00000AD
2425 #define STATUS_PIPE_BUSY 0xC00000AE
2426 #define STATUS_ILLEGAL_FUNCTION 0xC00000AF
2427 #define STATUS_PIPE_DISCONNECTED 0xC00000B0
2428 #define STATUS_PIPE_CLOSING 0xC00000B1
2429 #define STATUS_PIPE_CONNECTED 0xC00000B2
2430 #define STATUS_PIPE_LISTENING 0xC00000B3
2431 #define STATUS_INVALID_READ_MODE 0xC00000B4
2432 #define STATUS_IO_TIMEOUT 0xC00000B5
2433 #define STATUS_FILE_FORCED_CLOSED 0xC00000B6
2434 #define STATUS_PROFILING_NOT_STARTED 0xC00000B7
2435 #define STATUS_PROFILING_NOT_STOPPED 0xC00000B8
2436 #define STATUS_COULD_NOT_INTERPRET 0xC00000B9
2437 #define STATUS_FILE_IS_A_DIRECTORY 0xC00000BA
2438 #define STATUS_NOT_SUPPORTED 0xC00000BB
2439 #define STATUS_REMOTE_NOT_LISTENING 0xC00000BC
2440 #define STATUS_DUPLICATE_NAME 0xC00000BD
```

```
2441 #define STATUS_BAD_NETWORK_PATH 0xC00000BE
2442 #define STATUS_NETWORK_BUSY 0xC00000BF
2443 #define STATUS_DEVICE_DOES_NOT_EXIST 0xC00000C0
2444 #define STATUS_TOO_MANY_COMMANDS 0xC00000C1
2445 #define STATUS_ADAPTER_HARDWARE_ERROR 0xC00000C2
2446 #define STATUS_INVALID_NETWORK_RESPONSE 0xC00000C3
2447 #define STATUS_UNEXPECTED_NETWORK_ERROR 0xC00000C4
2448 #define STATUS_BAD_REMOTE_ADAPTER 0xC00000C5
2449 #define STATUS_PRINT_QUEUE_FULL 0xC00000C6
2450 #define STATUS_NO_SPOOL_SPACE 0xC00000C7
2451 #define STATUS_PRINT_CANCELLED 0xC00000C8
2452 #define STATUS_NETWORK_NAME_DELETED 0xC00000C9
2453 #define STATUS_NETWORK_ACCESS_DENIED 0xC00000CA
2454 #define STATUS_BAD_DEVICE_TYPE 0xC00000CB
2455 #define STATUS_BAD_NETWORK_NAME 0xC00000CC
2456 #define STATUS_TOO_MANY_NAMES 0xC00000CD
2457 #define STATUS_TOO_MANY_SESSIONS 0xC00000CE
2458 #define STATUS_SHARING_PAUSED 0xC00000CF
2459 #define STATUS_REQUEST_NOT_ACCEPTED 0xC00000D0
2460 #define STATUS_REDIRECTOR_PAUSED 0xC00000D1
2461 #define STATUS_NET_WRITE_FAULT 0xC00000D2
2462 #define STATUS_PROFILING_AT_LIMIT 0xC00000D3
2463 #define STATUS_NOT_SAME_DEVICE 0xC00000D4
2464 #define STATUS_FILE_RENAMED 0xC00000D5
2465 #define STATUS_VIRTUAL_CIRCUIT_CLOSED 0xC00000D6
2466 #define STATUS_NO_SECURITY_ON_OBJECT 0xC00000D7
2467 #define STATUS_CANT_WAIT 0xC00000D8
2468 #define STATUS_PIPE_EMPTY 0xC00000D9
2469 #define STATUS_CANT_ACCESS_DOMAIN_INFO 0xC00000DA
2470 #define STATUS_CANT_TERMINATE_SELF 0xC00000DB
2471 #define STATUS_INVALID_SERVER_STATE 0xC00000DC
2472 #define STATUS_INVALID_DOMAIN_STATE 0xC00000DD
2473 #define STATUS_INVALID_DOMAIN_ROLE 0xC00000DE
2474 #define STATUS_NO_SUCH_DOMAIN 0xC00000DF
2475 #define STATUS_DOMAIN_EXISTS 0xC00000E0
2476 #define STATUS_DOMAIN_LIMIT_EXCEEDED 0xC00000E1
2477 #define STATUS_OPLOCK_NOT_GRANTED 0xC00000E2
2478 #define STATUS_INVALID_OPLOCK_PROTOCOL 0xC00000E3
2479 #define STATUS_INTERNAL_DB_CORRUPTION 0xC00000E4
2480 #define STATUS_INTERNAL_ERROR 0xC00000E5
2481 #define STATUS_GENERIC_NOT_MAPPED 0xC00000E6
2482 #define STATUS_BAD_DESCRIPTOR_FORMAT 0xC00000E7
2483 #define STATUS_INVALID_USER_BUFFER 0xC00000E8
2484 #define STATUS_UNEXPECTED_IO_ERROR 0xC00000E9
2485 #define STATUS_UNEXPECTED_MM_CREATE_ERR 0xC00000EA
2486 #define STATUS_UNEXPECTED_MM_MAP_ERROR 0xC00000EB
2487 #define STATUS_UNEXPECTED_MM_EXTEND_ERR 0xC00000EC
2488 #define STATUS_NOT_LOGON_PROCESS 0xC00000ED
2489 #define STATUS_LOGON_SESSION_EXISTS 0xC00000EE
2490 #define STATUS_INVALID_PARAMETER_1 0xC00000EF
2491 #define STATUS_INVALID_PARAMETER_2 0xC00000F0
2492 #define STATUS_INVALID_PARAMETER_3 0xC00000F1
2493 #define STATUS_INVALID_PARAMETER_4 0xC00000F2
2494 #define STATUS_INVALID_PARAMETER_5 0xC00000F3
2495 #define STATUS_INVALID_PARAMETER_6 0xC00000F4
2496 #define STATUS_INVALID_PARAMETER_7 0xC00000F5
2497 #define STATUS_INVALID_PARAMETER_8 0xC00000F6
2498 #define STATUS_INVALID_PARAMETER_9 0xC00000F7
2499 #define STATUS_INVALID_PARAMETER_10 0xC00000F8
2500 #define STATUS_INVALID_PARAMETER_11 0xC00000F9
2501 #define STATUS_INVALID_PARAMETER_12 0xC00000FA
2502 #define STATUS_REDIRECTOR_NOT_STARTED 0xC00000FB
2503 #define STATUS_REDIRECTOR_STARTED 0xC00000FC
2504 #define STATUS_STACK_OVERFLOW 0xC00000FD
2505 #define STATUS_BAD_FUNCTION_TABLE 0xC00000FF
2506 #define STATUS_VARIABLE_NOT_FOUND 0xC0000100
2507 #define STATUS_DIRECTORY_NOT_EMPTY 0xC0000101
2508 #define STATUS_FILE_CORRUPT_ERROR 0xC0000102
2509 #define STATUS_NOT_A_DIRECTORY 0xC0000103
2510 #define STATUS_BAD_LOGON_SESSION_STATE 0xC0000104
2511 #define STATUS_LOGON_SESSION_COLLISION 0xC0000105
2512 #define STATUS_NAME_TOO_LONG 0xC0000106
2513 #define STATUS_FILES_OPEN 0xC0000107
2514 #define STATUS_CONNECTION_IN_USE 0xC0000108
2515 #define STATUS_MESSAGE_NOT_FOUND 0xC0000109
2516 #define STATUS_PROCESS_IS_TERMINATING 0xC000010A
2517 #define STATUS_INVALID_LOGON_TYPE 0xC000010B
2518 #define STATUS_NO_GUID_TRANSLATION 0xC000010C
2519 #define STATUS_CANNOT_IMPERSONATE 0xC000010D
2520 #define STATUS_IMAGE_ALREADY_LOADED 0xC000010E
2521 #define STATUS_ABIOS_NOT_PRESENT 0xC000010F
2522 #define STATUS_ABIOS_LID_NOT_EXIST 0xC0000110
2523 #define STATUS_ABIOS_LID_ALREADY_OWNED 0xC0000111
2524 #define STATUS_ABIOS_NOT_LID_OWNER 0xC0000112
2525 #define STATUS_ABIOS_INVALID_COMMAND 0xC0000113
2526 #define STATUS_ABIOS_INVALID_LID 0xC0000114
2527 #define STATUS_ABIOS_SELECTOR_NOT_AVAILABLE 0xC0000115
```

```
2528 #define STATUS_ABIOS_INVALID_SELECTOR 0xC0000116
2529 #define STATUS_NO_LDT 0xC0000117
2530 #define STATUS_INVALID_LDT_SIZE 0xC0000118
2531 #define STATUS_INVALID_LDT_OFFSET 0xC0000119
2532 #define STATUS_INVALID_LDT_DESCRIPTOR 0xC000011A
2533 #define STATUS_INVALID_IMAGE_NE_FORMAT 0xC000011B
2534 #define STATUS_RXACT_INVALID_STATE 0xC000011C
2535 #define STATUS_RXACT_COMMIT_FAILURE 0xC000011D
2536 #define STATUS_MAPPED_FILE_SIZE_ZERO 0xC000011E
2537 #define STATUS_TOO_MANY_OPENED_FILES 0xC000011F
2538 #define STATUS_CANCELLED 0xC0000120
2539 #define STATUS_CANNOT_DELETE 0xC0000121
2540 #define STATUS_INVALID_COMPUTER_NAME 0xC0000122
2541 #define STATUS_FILE_DELETED 0xC0000123
2542 #define STATUS_SPECIAL_ACCOUNT 0xC0000124
2543 #define STATUS_SPECIAL_GROUP 0xC0000125
2544 #define STATUS_SPECIAL_USER 0xC0000126
2545 #define STATUS_MEMBERS_PRIMARY_GROUP 0xC0000127
2546 #define STATUS_FILE_CLOSED 0xC0000128
2547 #define STATUS_TOO_MANY_THREADS 0xC0000129
2548 #define STATUS_THREAD_NOT_IN_PROCESS 0xC000012A
2549 #define STATUS_TOKEN_ALREADY_IN_USE 0xC000012B
2550 #define STATUS_PAGEFILE_QUOTA_EXCEEDED 0xC000012C
2551 #define STATUS_COMMITMENT_LIMIT 0xC000012D
2552 #define STATUS_INVALID_IMAGE_LE_FORMAT 0xC000012E
2553 #define STATUS_INVALID_IMAGE_NOT_MZ 0xC000012F
2554 #define STATUS_INVALID_IMAGE_PROTECT 0xC0000130
2555 #define STATUS_INVALID_IMAGE_WIN_16 0xC0000131
2556 #define STATUS_LOGON_SERVER_CONFLICT 0xC0000132
2557 #define STATUS_TIME_DIFFERENCE_AT_DC 0xC0000133
2558 #define STATUS_SYNCHRONIZATION_REQUIRED 0xC0000134
2559 #define STATUS_DLL_NOT_FOUND 0xC0000135
2560 #define STATUS_OPEN_FAILED 0xC0000136
2561 #define STATUS_IO_PRIVILEGE_FAILED 0xC0000137
2562 #define STATUS_ORDINAL_NOT_FOUND 0xC0000138
2563 #define STATUS_ENTRYPOINT_NOT_FOUND 0xC0000139
2564 #define STATUS_CONTROL_C_EXIT 0xC000013A
2565 #define STATUS_LOCAL_DISCONNECT 0xC000013B
2566 #define STATUS_REMOTE_DISCONNECT 0xC000013C
2567 #define STATUS_REMOTE_RESOURCES 0xC000013D
2568 #define STATUS_LINK_FAILED 0xC000013E
2569 #define STATUS_LINK_TIMEOUT 0xC000013F
2570 #define STATUS_INVALID_CONNECTION 0xC0000140
2571 #define STATUS_INVALID_ADDRESS 0xC0000141
2572 #define STATUS_DLL_INIT_FAILED 0xC0000142
2573 #define STATUS_MISSING_SYSTEMFILE 0xC0000143
2574 #define STATUS_UNHANDLED_EXCEPTION 0xC0000144
2575 #define STATUS_APP_INIT_FAILURE 0xC0000145
2576 #define STATUS_PAGEFILE_CREATE_FAILED 0xC0000146
2577 #define STATUS_NO_PAGEFILE 0xC0000147
2578 #define STATUS_INVALID_LEVEL 0xC0000148
2579 #define STATUS_WRONG_PASSWORD_CORE 0xC0000149
2580 #define STATUS_ILLEGAL_FLOAT_CONTEXT 0xC000014A
2581 #define STATUS_PIPE_BROKEN 0xC000014B
2582 #define STATUS_REGISTRY_CORRUPT 0xC000014C
2583 #define STATUS_REGISTRY_IO_FAILED 0xC000014D
2584 #define STATUS_NO_EVENT_PAIR 0xC000014E
2585 #define STATUS_UNRECOGNIZED_VOLUME 0xC000014F
2586 #define STATUS_SERIAL_NO_DEVICE_INITED 0xC0000150
2587 #define STATUS_NO_SUCH_ALIAS 0xC0000151
2588 #define STATUS_MEMBER_NOT_IN_ALIAS 0xC0000152
2589 #define STATUS_MEMBER_IN_ALIAS 0xC0000153
2590 #define STATUS_ALIAS_EXISTS 0xC0000154
2591 #define STATUS_LOGON_NOT_GRANTED 0xC0000155
2592 #define STATUS_TOO_MANY_SECRETS 0xC0000156
2593 #define STATUS_SECRET_TOO_LONG 0xC0000157
2594 #define STATUS_INTERNAL_DB_ERROR 0xC0000158
2595 #define STATUS_FULLSCREEN_MODE 0xC0000159
2596 #define STATUS_TOO_MANY_CONTEXT_IDS 0xC000015A
2597 #define STATUS_LOGON_TYPE_NOT_GRANTED 0xC000015B
2598 #define STATUS_NOT_REGISTRY_FILE 0xC000015C
2599 #define STATUS_NT_CROSS_ENCRYPTION_REQUIRED 0xC000015D
2600 #define STATUS_DOMAIN_CTRLR_CONFIG_ERROR 0xC000015E
2601 #define STATUS_FT_MISSING_MEMBER 0xC000015F
2602 #define STATUS_ILL_FORMED_SERVICE_ENTRY 0xC0000160
2603 #define STATUS_ILLEGAL_CHARACTER 0xC0000161
2604 #define STATUS_UNMAPPABLE_CHARACTER 0xC0000162
2605 #define STATUS_UNDEFINED_CHARACTER 0xC0000163
2606 #define STATUS_FLOPPY_VOLUME 0xC0000164
2607 #define STATUS_FLOPPY_ID_MARK_NOT_FOUND 0xC0000165
2608 #define STATUS_FLOPPY_WRONG_CYLINDER 0xC0000166
2609 #define STATUS_FLOPPY_UNKNOWN_ERROR 0xC0000167
2610 #define STATUS_FLOPPY_BAD_REGISTERS 0xC0000168
2611 #define STATUS_DISK_RECALIBRATE_FAILED 0xC0000169
2612 #define STATUS_DISK_OPERATION_FAILED 0xC000016A
2613 #define STATUS_DISK_RESET_FAILED 0xC000016B
2614 #define STATUS_SHARED_IRQ_BUSY 0xC000016C
```

```
2615 #define STATUS_FT_ORPHANING 0xC000016D
2616 #define STATUS_BIOS_FAILED_TO_CONNECT_INTERRUPT 0xC000016E
2617
2618 #define STATUS_PARTITION_FAILURE 0xC0000172
2619 #define STATUS_INVALID_BLOCK_LENGTH 0xC0000173
2620 #define STATUS_DEVICE_NOT_PARTITIONED 0xC0000174
2621 #define STATUS_UNABLE_TO_LOCK_MEDIA 0xC0000175
2622 #define STATUS_UNABLE_TO_UNLOAD_MEDIA 0xC0000176
2623 #define STATUS_EOM_OVERFLOW 0xC0000177
2624 #define STATUS_NO_MEDIA 0xC0000178
2625 #define STATUS_NO_SUCH_MEMBER 0xC000017A
2626 #define STATUS_INVALID_MEMBER 0xC000017B
2627 #define STATUS_KEY_DELETED 0xC000017C
2628 #define STATUS_NO_LOG_SPACE 0xC000017D
2629 #define STATUS_TOO_MANY_SIDS 0xC000017E
2630 #define STATUS_LM_CROSS_ENCRYPTION_REQUIRED 0xC000017F
2631 #define STATUS_KEY_HAS_CHILDREN 0xC0000180
2632 #define STATUS_CHILD_MUST_BE_VOLATILE 0xC0000181
2633 #define STATUS_DEVICE_CONFIGURATION_ERROR 0xC0000182
2634 #define STATUS_DRIVER_INTERNAL_ERROR 0xC0000183
2635 #define STATUS_INVALID_DEVICE_STATE 0xC0000184
2636 #define STATUS_IO_DEVICE_ERROR 0xC0000185
2637 #define STATUS_DEVICE_PROTOCOL_ERROR 0xC0000186
2638 #define STATUS_BACKUP_CONTROLLER 0xC0000187
2639 #define STATUS_LOG_FILE_FULL 0xC0000188
2640 #define STATUS_TOO_LATE 0xC0000189
2641 #define STATUS_NO_TRUST_LSA_SECRET 0xC000018A
2642 #define STATUS_NO_TRUST_SAM_ACCOUNT 0xC000018B
2643 #define STATUS_TRUSTED_DOMAIN_FAILURE 0xC000018C
2644 #define STATUS_TRUSTED_RELATIONSHIP_FAILURE 0xC000018D
2645 #define STATUS_EVENTLOG_FILE_CORRUPT 0xC000018E
2646 #define STATUS_EVENTLOG_CANT_START 0xC000018F
2647 #define STATUS_TRUST_FAILURE 0xC0000190
2648 #define STATUS_MUTANT_LIMIT_EXCEEDED 0xC0000191
2649 #define STATUS_NETLOGON_NOT_STARTED 0xC0000192
2650 #define STATUS_ACCOUNT_EXPIRED 0xC0000193
2651 #define STATUS_POSSIBLE_DEADLOCK 0xC0000194
2652 #define STATUS_NETWORK_CREDENTIAL_CONFLICT 0xC0000195
2653 #define STATUS_REMOTE_SESSION_LIMIT 0xC0000196
2654 #define STATUS_EVENTLOG_FILE_CHANGED 0xC0000197
2655 #define STATUS_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT 0xC0000198
2656 #define STATUS_NOLOGON_WORKSTATION_TRUST_ACCOUNT 0xC0000199
2657 #define STATUS_NOLOGON_SERVER_TRUST_ACCOUNT 0xC000019A
2658 #define STATUS_DOMAIN_TRUST_INCONSISTENT 0xC000019B
2659 #define STATUS_FS_DRIVER_REQUIRED 0xC000019C
2660
2661 #define STATUS_NO_USER_SESSION_KEY 0xC0000202
2662 #define STATUS_USER_SESSION_DELETED 0xC0000203
2663 #define STATUS_RESOURCE_LANG_NOT_FOUND 0xC0000204
2664 #define STATUS_INSUFF_SERVER_RESOURCES 0xC0000205
2665 #define STATUS_INVALID_BUFFER_SIZE 0xC0000206
2666 #define STATUS_INVALID_ADDRESS_COMPONENT 0xC0000207
2667 #define STATUS_INVALID_ADDRESS_WILDCARD 0xC0000208
2668 #define STATUS_TOO_MANY_ADDRESSES 0xC0000209
2669 #define STATUS_ADDRESS_ALREADY_EXISTS 0xC000020A
2670 #define STATUS_ADDRESS_CLOSED 0xC000020B
2671 #define STATUS_CONNECTION_DISCONNECTED 0xC000020C
2672 #define STATUS_CONNECTION_RESET 0xC000020D
2673 #define STATUS_TOO_MANY_NODES 0xC000020E
2674 #define STATUS_TRANSACTION_ABORTED 0xC000020F
2675 #define STATUS_TRANSACTION_TIMED_OUT 0xC0000210
2676 #define STATUS_TRANSACTION_NO_RELEASE 0xC0000211
2677 #define STATUS_TRANSACTION_NO_MATCH 0xC0000212
2678 #define STATUS_TRANSACTION_RESPONDED 0xC0000213
2679 #define STATUS_TRANSACTION_INVALID_ID 0xC0000214
2680 #define STATUS_TRANSACTION_INVALID_TYPE 0xC0000215
2681 #define STATUS_NOT_SERVER_SESSION 0xC0000216
2682 #define STATUS_NOT_CLIENT_SESSION 0xC0000217
2683 #define STATUS_CANNOT_LOAD_REGISTRY_FILE 0xC0000218
2684 #define STATUS_DEBUG_ATTACH_FAILED 0xC0000219
2685 #define STATUS_SYSTEM_PROCESS_TERMINATED 0xC000021A
2686 #define STATUS_DATA_NOT_ACCEPTED 0xC000021B
2687 #define STATUS_NO_BROWSER_SERVERS_FOUND 0xC000021C
2688 #define STATUS_VDM_HARD_ERROR 0xC000021D
2689 #define STATUS_DRIVER_CANCEL_TIMEOUT 0xC000021E
2690 #define STATUS_REPLY_MESSAGE_MISMATCH 0xC000021F
2691 #define STATUS_MAPPED_ALIGNMENT 0xC0000220
2692 #define STATUS_IMAGE_CHECKSUM_MISMATCH 0xC0000221
2693 #define STATUS_LOST_WRITEBEHIND_DATA 0xC0000222
2694 #define STATUS_CLIENT_SERVER_PARAMETERS_INVALID 0xC0000223
2695 #define STATUS_PASSWORD_MUST_CHANGE 0xC0000224
2696 #define STATUS_NOT_FOUND 0xC0000225
2697 #define STATUS_NOT_TINY_STREAM 0xC0000226
2698 #define STATUS_RECOVERY_FAILURE 0xC0000227
2699 #define STATUS_STACK_OVERFLOW_READ 0xC0000228
2700 #define STATUS_FAIL_CHECK 0xC0000229
2701 #define STATUS_DUPLICATE_OBJECTID 0xC000022A
```



```
2702 #define STATUS_OBJECTID_EXISTS 0xC000022B
2703 #define STATUS_CONVERT_TO_LARGE 0xC000022C
2704 #define STATUS_RETRY 0xC000022D
2705 #define STATUS_FOUND_OUT_OF_SCOPE 0xC000022E
2706 #define STATUS_ALLOCATE_BUCKET 0xC000022F
2707 #define STATUS_PROPSET_NOT_FOUND 0xC0000230
2708 #define STATUS_MARSHALL_OVERFLOW 0xC0000231
2709 #define STATUS_INVALID_VARIANT 0xC0000232
2710 #define STATUS_DOMAIN_CONTROLLER_NOT_FOUND 0xC0000233
2711 #define STATUS_ACCOUNT_LOCKED_OUT 0xC0000234
2712 #define STATUS_HANDLE_NOT_CLOSABLE 0xC0000235
2713 #define STATUS_CONNECTION_REFUSED 0xC0000236
2714 #define STATUS_GRACEFUL_DISCONNECT 0xC0000237
2715 #define STATUS_ADDRESS_ALREADY_ASSOCIATED 0xC0000238
2716 #define STATUS_ADDRESS_NOT_ASSOCIATED 0xC0000239
2717 #define STATUS_CONNECTION_INVALID 0xC000023A
2718 #define STATUS_CONNECTION_ACTIVE 0xC000023B
2719 #define STATUS_NETWORK_UNREACHABLE 0xC000023C
2720 #define STATUS_HOST_UNREACHABLE 0xC000023D
2721 #define STATUS_PROTOCOL_UNREACHABLE 0xC000023E
2722 #define STATUS_PORT_UNREACHABLE 0xC000023F
2723 #define STATUS_REQUEST_ABORTED 0xC0000240
2724 #define STATUS_CONNECTION_ABORTED 0xC0000241
2725 #define STATUS_BAD_COMPRESSION_BUFFER 0xC0000242
2726 #define STATUS_USER_MAPPED_FILE 0xC0000243
2727 #define STATUS_AUDIT_FAILED 0xC0000244
2728 #define STATUS_TIMER_RESOLUTION_NOT_SET 0xC0000245
2729 #define STATUS_CONNECTION_COUNT_LIMIT 0xC0000246
2730 #define STATUS_LOGIN_TIME_RESTRICTION 0xC0000247
2731 #define STATUS_LOGIN_WKSTA_RESTRICTION 0xC0000248
2732 #define STATUS_IMAGE_MP_UP_MISMATCH 0xC0000249
2733 #define STATUS_INSUFFICIENT_LOGON_INFO 0xC0000250
2734 #define STATUS_BAD_DLL_ENTRYPOINT 0xC0000251
2735 #define STATUS_BAD_SERVICE_ENTRYPOINT 0xC0000252
2736 #define STATUS_LPC_REPLY_LOST 0xC0000253
2737 #define STATUS_IP_ADDRESS_CONFLICT1 0xC0000254
2738 #define STATUS_IP_ADDRESS_CONFLICT2 0xC0000255
2739 #define STATUS_REGISTRY_QUOTA_LIMIT 0xC0000256
2740 #define STATUS_PATH_NOT_COVERED 0xC0000257
2741 #define STATUS_NO_CALLBACK_ACTIVE 0xC0000258
2742 #define STATUS_LICENSE_QUOTA_EXCEEDED 0xC0000259
2743 #define STATUS_PWD_TOO_SHORT 0xC000025A
2744 #define STATUS_PWD_TOO_RECENT 0xC000025B
2745 #define STATUS_PWD_HISTORY_CONFLICT 0xC000025C
2746 #define STATUS_PLUGPLAY_NO_DEVICE 0xC000025E
2747 #define STATUS_UNSUPPORTED_COMPRESSION 0xC000025F
2748 #define STATUS_INVALID_HW_PROFILE 0xC0000260
2749 #define STATUS_INVALID_PLUGPLAY_DEVICE_PATH 0xC0000261
2750 #define STATUS_DRIVER_ORDINAL_NOT_FOUND 0xC0000262
2751 #define STATUS_DRIVER_ENTRYPOINT_NOT_FOUND 0xC0000263
2752 #define STATUS_RESOURCE_NOT_OWNED 0xC0000264
2753 #define STATUS_TOO_MANY_LINKS 0xC0000265
2754 #define STATUS_QUOTA_LIST_INCONSISTENT 0xC0000266
2755 #define STATUS_FILE_IS_OFFLINE 0xC0000267
2756 #define STATUS_EVALUATION_EXPIRATION 0xC0000268
2757 #define STATUS_ILLEGAL_DLL_RELOCATION 0xC0000269
2758 #define STATUS_LICENSE_VIOLATION 0xC000026A
2759 #define STATUS_DLL_INIT_FAILED_LOGOFF 0xC000026B
2760 #define STATUS_DRIVER_UNABLE_TO_LOAD 0xC000026C
2761 #define STATUS_DFS_UNAVAILABLE 0xC000026D
2762 #define STATUS_VOLUME_DISMOUNTED 0xC000026E
2763 #define STATUS_WX86_INTERNAL_ERROR 0xC000026F
2764 #define STATUS_WX86_FLOAT_STACK_CHECK 0xC0000270
2765 #define STATUS_WOW_ASSERTION 0xC0009898
2766 #define RPC_NT_INVALID_STRING_BINDING 0xC0020001
2767 #define RPC_NT_WRONG_KIND_OF_BINDING 0xC0020002
2768 #define RPC_NT_INVALID_BINDING 0xC0020003
2769 #define RPC_NT_PROTSEQ_NOT_SUPPORTED 0xC0020004
2770 #define RPC_NT_INVALID_RPC_PROTSEQ 0xC0020005
2771 #define RPC_NT_INVALID_STRING_UUID 0xC0020006
2772 #define RPC_NT_INVALID_ENDPOINT_FORMAT 0xC0020007
2773 #define RPC_NT_INVALID_NET_ADDR 0xC0020008
2774 #define RPC_NT_NO_ENDPOINT_FOUND 0xC0020009
2775 #define RPC_NT_INVALID_TIMEOUT 0xC002000A
2776 #define RPC_NT_OBJECT_NOT_FOUND 0xC002000B
2777 #define RPC_NT_ALREADY_REGISTERED 0xC002000C
2778 #define RPC_NT_TYPE_ALREADY_REGISTERED 0xC002000D
2779 #define RPC_NT_ALREADY_LISTENING 0xC002000E
2780 #define RPC_NT_NO_PROTSEQS_REGISTERED 0xC002000F
2781 #define RPC_NT_NOT_LISTENING 0xC0020010
2782 #define RPC_NT_UNKNOWN_MGR_TYPE 0xC0020011
2783 #define RPC_NT_UNKNOWN_IF 0xC0020012
2784 #define RPC_NT_NO_BINDINGS 0xC0020013
2785 #define RPC_NT_NO_PROTSEQS 0xC0020014
2786 #define RPC_NT_CANT_CREATE_ENDPOINT 0xC0020015
2787 #define RPC_NT_OUT_OF_RESOURCES 0xC0020016
2788 #define RPC_NT_SERVER_UNAVAILABLE 0xC0020017
```

```
2789 #define RPC_NT_SERVER_TOO_BUSY 0xC0020018
2790 #define RPC_NT_INVALID_NETWORK_OPTIONS 0xC0020019
2791 #define RPC_NT_NO_CALL_ACTIVE 0xC002001A
2792 #define RPC_NT_CALL_FAILED 0xC002001B
2793 #define RPC_NT_CALL_FAILED_DNE 0xC002001C
2794 #define RPC_NT_PROTOCOL_ERROR 0xC002001D
2795 #define RPC_NT_UNSUPPORTED_TRANS_SYN 0xC002001F
2796 #define RPC_NT_UNSUPPORTED_TYPE 0xC0020021
2797 #define RPC_NT_INVALID_TAG 0xC0020022
2798 #define RPC_NT_INVALID_BOUND 0xC0020023
2799 #define RPC_NT_NO_ENTRY_NAME 0xC0020024
2800 #define RPC_NT_INVALID_NAME_SYNTAX 0xC0020025
2801 #define RPC_NT_UNSUPPORTED_NAME_SYNTAX 0xC0020026
2802 #define RPC_NT_UUID_NO_ADDRESS 0xC0020028
2803 #define RPC_NT_DUPLICATE_ENDPOINT 0xC0020029
2804 #define RPC_NT_UNKNOWN_AUTHN_TYPE 0xC002002A
2805 #define RPC_NT_MAX_CALLS_TOO_SMALL 0xC002002B
2806 #define RPC_NT_STRING_TOO_LONG 0xC002002C
2807 #define RPC_NT_PROTSEQ_NOT_FOUND 0xC002002D
2808 #define RPC_NT_PROCNUM_OUT_OF_RANGE 0xC002002E
2809 #define RPC_NT_BINDING_HAS_NO_AUTH 0xC002002F
2810 #define RPC_NT_UNKNOWN_AUTHN_SERVICE 0xC0020030
2811 #define RPC_NT_UNKNOWN_AUTHN_LEVEL 0xC0020031
2812 #define RPC_NT_INVALID_AUTH_IDENTITY 0xC0020032
2813 #define RPC_NT_UNKNOWN_AUTHZ_SERVICE 0xC0020033
2814 #define EPT_NT_INVALID_ENTRY 0xC0020034
2815 #define EPT_NT_CANT_PERFORM_OP 0xC0020035
2816 #define EPT_NT_NOT_REGISTERED 0xC0020036
2817 #define RPC_NT_NOTHING_TO_EXPORT 0xC0020037
2818 #define RPC_NT_INCOMPLETE_NAME 0xC0020038
2819 #define RPC_NT_INVALID_VERS_OPTION 0xC0020039
2820 #define RPC_NT_NO_MORE_MEMBERS 0xC002003A
2821 #define RPC_NT_NOT_ALL_OBJS_UNEXPORTED 0xC002003B
2822 #define RPC_NT_INTERFACE_NOT_FOUND 0xC002003C
2823 #define RPC_NT_ENTRY_ALREADY_EXISTS 0xC002003D
2824 #define RPC_NT_ENTRY_NOT_FOUND 0xC002003E
2825 #define RPC_NT_NAME_SERVICE_UNAVAILABLE 0xC002003F
2826 #define RPC_NT_INVALID_NAF_ID 0xC0020040
2827 #define RPC_NT_CANNOT_SUPPORT 0xC0020041
2828 #define RPC_NT_NO_CONTEXT_AVAILABLE 0xC0020042
2829 #define RPC_NT_INTERNAL_ERROR 0xC0020043
2830 #define RPC_NT_ZERO_DIVIDE 0xC0020044
2831 #define RPC_NT_ADDRESS_ERROR 0xC0020045
2832 #define RPC_NT_FP_DIV_ZERO 0xC0020046
2833 #define RPC_NT_FP_UNDERFLOW 0xC0020047
2834 #define RPC_NT_FP_OVERFLOW 0xC0020048
2835 #define RPC_NT_NO_MORE_ENTRIES 0xC0030001
2836 #define RPC_NT_SS_CHAR_TRANS_OPEN_FAIL 0xC0030002
2837 #define RPC_NT_SS_CHAR_TRANS_SHORT_FILE 0xC0030003
2838 #define RPC_NT_SS_IN_NULL_CONTEXT 0xC0030004
2839 #define RPC_NT_SS_CONTEXT_MISMATCH 0xC0030005
2840 #define RPC_NT_SS_CONTEXT_DAMAGED 0xC0030006
2841 #define RPC_NT_SS_HANDLES_MISMATCH 0xC0030007
2842 #define RPC_NT_SS_CANNOT_GET_CALL_HANDLE 0xC0030008
2843 #define RPC_NT_NULL_REF_POINTER 0xC0030009
2844 #define RPC_NT_ENUM_VALUE_OUT_OF_RANGE 0xC003000A
2845 #define RPC_NT_BYTE_COUNT_TOO_SMALL 0xC003000B
2846 #define RPC_NT_BAD_STUB_DATA 0xC003000C
2847 #define RPC_NT_CALL_IN_PROGRESS 0xC0020049
2848 #define RPC_NT_NO_MORE_BINDINGS 0xC002004A
2849 #define RPC_NT_GROUP_MEMBER_NOT_FOUND 0xC002004B
2850 #define EPT_NT_CANT_CREATE 0xC002004C
2851 #define RPC_NT_INVALID_OBJECT 0xC002004D
2852 #define RPC_NT_NO_INTERFACES 0xC002004F
2853 #define RPC_NT_CALL_CANCELLED 0xC0020050
2854 #define RPC_NT_BINDING_INCOMPLETE 0xC0020051
2855 #define RPC_NT_COMM_FAILURE 0xC0020052
2856 #define RPC_NT_UNSUPPORTED_AUTHN_LEVEL 0xC0020053
2857 #define RPC_NT_NO_PRINC_NAME 0xC0020054
2858 #define RPC_NT_NOT_RPC_ERROR 0xC0020055
2859 #define RPC_NT_UUID_LOCAL_ONLY 0x40020056
2860 #define RPC_NT_SEC_PKG_ERROR 0xC0020057
2861 #define RPC_NT_NOT_CANCELLED 0xC0020058
2862 #define RPC_NT_INVALID_ES_ACTION 0xC0030059
2863 #define RPC_NT_WRONG_ES_VERSION 0xC003005A
2864 #define RPC_NT_WRONG_STUB_VERSION 0xC003005B
2865 #define RPC_NT_INVALID_PIPE_OBJECT 0xC003005C
2866 #define RPC_NT_INVALID_PIPE_OPERATION 0xC003005D
2867 #define RPC_NT_WRONG_PIPE_VERSION 0xC003005E
2868 #define RPC_NT_SEND_INCOMPLETE 0x400200AF
2869
2870 #define MAXIMUM_WAIT_OBJECTS 64
2871 #define MAXIMUM_SUSPEND_COUNT 127
2872
2873
2874 /*
2875 * Return values from the actual exception handlers
```

```

2876 */
2877
2878 #define ExceptionContinueExecution 0
2879 #define ExceptionContinueSearch 1
2880 #define ExceptionNestedException 2
2881 #define ExceptionCollidedUnwind 3
2882
2883 /*
2884 * Return values from filters in except() and from UnhandledExceptionFilter
2885 */
2886
2887 #define EXCEPTION_EXECUTE_HANDLER 1
2888 #define EXCEPTION_CONTINUE_SEARCH 0
2889 #define EXCEPTION_CONTINUE_EXECUTION -1
2890
2891 /*
2892 * From OS/2 2.0 exception handling
2893 * Win32 seems to use the same flags as ExceptionFlags in an EXCEPTION_RECORD
2894 */
2895
2896 #define EH_NONCONTINUABLE 0x01
2897 #define EH_UNWINDING 0x02
2898 #define EH_EXIT_UNWIND 0x04
2899 #define EH_STACK_INVALID 0x08
2900 #define EH_NESTED_CALL 0x10
2901
2902 #define EXCEPTION_CONTINUABLE 0
2903 #define EXCEPTION_NONCONTINUABLE EH_NONCONTINUABLE
2904
2905 /*
2906 * The exception record used by Win32 to give additional information
2907 * about exception to exception handlers.
2908 */
2909
2910 #define EXCEPTION_MAXIMUM_PARAMETERS 15
2911
2912 typedef struct __EXCEPTION_RECORD
2913 {
2914     DWORD ExceptionCode;
2915     DWORD ExceptionFlags;
2916     struct __EXCEPTION_RECORD *ExceptionRecord;
2917
2918     LPVOID ExceptionAddress;
2919     DWORD NumberParameters;
2920     DWORD ExceptionInformation[EXCEPTION_MAXIMUM_PARAMETERS];
2921 } EXCEPTION_RECORD, *PEXCEPTION_RECORD;
2922
2923 /*
2924 * The exception pointers structure passed to exception filters
2925 * in except() and the UnhandledExceptionFilter().
2926 */
2927
2928 typedef struct _EXCEPTION_POINTERS
2929 {
2930     PEXCEPTION_RECORD ExceptionRecord;
2931     PCONTEXT ContextRecord;
2932 } EXCEPTION_POINTERS, *PEXCEPTION_POINTERS;
2933
2934
2935 /*
2936 * The exception frame, used for registering exception handlers
2937 * Win32 cares only about this, but compilers generally emit
2938 * larger exception frames for their own use.
2939 */
2940
2941 struct __EXCEPTION_FRAME;
2942
2943 typedef DWORD (*PEXCEPTION_HANDLER) (PEXCEPTION_RECORD, struct __EXCEPTION_FRAME*,
2944                                     PCONTEXT, struct __EXCEPTION_FRAME **);
2945
2946 typedef struct __EXCEPTION_FRAME
2947 {
2948     struct __EXCEPTION_FRAME *Prev;
2949     PEXCEPTION_HANDLER Handler;
2950 } EXCEPTION_FRAME, *PEXCEPTION_FRAME;
2951
2952 /*
2953 * function pointer to a exception filter
2954 */
2955
2956 typedef LONG CALLBACK (*PTOP_LEVEL_EXCEPTION_FILTER) (PEXCEPTION_POINTERS ExceptionInfo);
2957 typedef PTOP_LEVEL_EXCEPTION_FILTER LPTOP_LEVEL_EXCEPTION_FILTER;
2958
2959 DWORD WINAPI UnhandledExceptionFilter( PEXCEPTION_POINTERS epointers );
2960 LPTOP_LEVEL_EXCEPTION_FILTER
2961 WINAPI SetUnhandledExceptionFilter( LPTOP_LEVEL_EXCEPTION_FILTER filter );
2962

```



```

2963 /* status values for ContinueDebugEvent */
2964 #define DBG_CONTINUE 0x00010002
2965 #define DBG_TERMINATE_THREAD 0x40010003
2966 #define DBG_TERMINATE_PROCESS 0x40010004
2967 #define DBG_CONTROL_C 0x40010005
2968 #define DBG_CONTROL_BREAK 0x40010008
2969 #define DBG_EXCEPTION_NOT_HANDLED 0x80010001
2970
2971 typedef struct _NT_TIB
2972 {
2973     struct _EXCEPTION_REGISTRATION_RECORD *ExceptionList;
2974     PVOID StackBase;
2975     PVOID StackLimit;
2976     PVOID SubSystemTib;
2977     union {
2978         PVOID FiberData;
2979         DWORD Version;
2980     } DUMMYUNIONNAME;
2981     PVOID ArbitraryUserPointer;
2982     struct _NT_TIB *Self;
2983 } NT_TIB, *PNT_TIB;
2984
2985 struct _TEB;
2986
2987 #if defined(__i386__) && defined(__GNUC__) && !defined(__CHECKER__)
2988 extern inline struct _TEB WINAPI *NtCurrentTeb(void);
2989 extern inline struct _TEB WINAPI *NtCurrentTeb(void)
2990 {
2991     struct _TEB *teb;
2992     __asm__(".byte 0x64\n\tmovl (0x18),%0" : "=r" (teb));
2993     return teb;
2994 }
2995 #else
2996 extern struct _TEB WINAPI *NtCurrentTeb(void);
2997 #endif
2998
2999
3000 /*
3001 * File formats definitions
3002 */
3003
3004 typedef struct _IMAGE_DOS_HEADER {
3005     WORD e_magic; /* 00: MZ Header signature */
3006     WORD e_cblp; /* 02: Bytes on last page of file */
3007     WORD e_cp; /* 04: Pages in file */
3008     WORD e_crlc; /* 06: Relocations */
3009     WORD e_cparhdr; /* 08: Size of header in paragraphs */
3010     WORD e_minalloc; /* 0a: Minimum extra paragraphs needed */
3011     WORD e_maxalloc; /* 0c: Maximum extra paragraphs needed */
3012     WORD e_ss; /* 0e: Initial (relative) SS value */
3013     WORD e_sp; /* 10: Initial SP value */
3014     WORD e_csum; /* 12: Checksum */
3015     WORD e_ip; /* 14: Initial IP value */
3016     WORD e_cs; /* 16: Initial (relative) CS value */
3017     WORD e_lfarlc; /* 18: File address of relocation table */
3018     WORD e_ovno; /* 1a: Overlay number */
3019     WORD e_res[4]; /* 1c: Reserved words */
3020     WORD e_oemid; /* 24: OEM identifier (for e_oeminfo) */
3021     WORD e_oeminfo; /* 26: OEM information; e_oemid specific */
3022     WORD e_res2[10]; /* 28: Reserved words */
3023     DWORD e_lfanew; /* 3c: Offset to extended header */
3024 } IMAGE_DOS_HEADER, *PIMAGE_DOS_HEADER;
3025
3026 #define IMAGE_DOS_SIGNATURE 0x5A4D /* MZ */
3027 #define IMAGE_OS2_SIGNATURE 0x454E /* NE */
3028 #define IMAGE_OS2_SIGNATURE_LE 0x454C /* LE */
3029 #define IMAGE_OS2_SIGNATURE_LX 0x584C /* LX */
3030 #define IMAGE_VXD_SIGNATURE 0x454C /* LE */
3031 #define IMAGE_NT_SIGNATURE 0x00004550 /* PE00 */
3032
3033 /*
3034 * This is the Windows executable (NE) header.
3035 * the name IMAGE_OS2_HEADER is misleading, but in the SDK this way.
3036 */
3037 typedef struct
3038 {
3039     WORD ne_magic; /* 00 NE signature 'NE' */
3040     BYTE ne_ver; /* 02 Linker version number */
3041     BYTE ne_rev; /* 03 Linker revision number */
3042     WORD ne_enttab; /* 04 Offset to entry table relative to NE */
3043     WORD ne_cbenttab; /* 06 Length of entry table in bytes */
3044     LONG ne_crc; /* 08 Checksum */
3045     WORD ne_flags; /* 0c Flags about segments in this file */
3046     WORD ne_autodata; /* 0e Automatic data segment number */
3047     WORD ne_heap; /* 10 Initial size of local heap */
3048     WORD ne_stack; /* 12 Initial size of stack */
3049     DWORD ne_csip; /* 14 Initial CS:IP */

```

```

3050     DWORD ne_sssp;           /* 18 Initial SS:SP */
3051     WORD ne_cseg;            /* 1c # of entries in segment table */
3052     WORD ne_cmod;            /* 1e # of entries in module reference tab. */
3053     WORD ne_cbnrestab;       /* 20 Length of nonresident-name table */
3054     WORD ne_segtab;          /* 22 Offset to segment table */
3055     WORD ne_rsrctab;         /* 24 Offset to resource table */
3056     WORD ne_restab;          /* 26 Offset to resident-name table */
3057     WORD ne_modtab;          /* 28 Offset to module reference table */
3058     WORD ne_imptab;          /* 2a Offset to imported name table */
3059     DWORD ne_nrestab;        /* 2c Offset to nonresident-name table */
3060     WORD ne_cmovent;         /* 30 # of movable entry points */
3061     WORD ne_align;           /* 32 Logical sector alignment shift count */
3062     WORD ne_cres;            /* 34 # of resource segments */
3063     BYTE ne_exetyp;          /* 36 Flags indicating target OS */
3064     BYTE ne_flagsothers;     /* 37 Additional information flags */
3065     WORD fastload_offset;     /* 38 Offset to fast load area (should be ne_pretthunks) */
3066     WORD fastload_length;     /* 3a Length of fast load area (should be ne_psegrefbytes) */
3067     WORD ne_swaparea;        /* 3c Reserved by Microsoft */
3068     WORD ne_expver;          /* 3e Expected Windows version number */
3069 } IMAGE_OS2_HEADER, *PIMAGE_OS2_HEADER;
3070
3071 typedef struct _IMAGE_VXD_HEADER {
3072     WORD e32_magic;
3073     BYTE e32_border;
3074     BYTE e32_worder;
3075     DWORD e32_level;
3076     WORD e32_cpu;
3077     WORD e32_os;
3078     DWORD e32_ver;
3079     DWORD e32_mflags;
3080     DWORD e32_mpages;
3081     DWORD e32_startobj;
3082     DWORD e32_eip;
3083     DWORD e32_stackobj;
3084     DWORD e32_esp;
3085     DWORD e32_pagesize;
3086     DWORD e32_lastpagesize;
3087     DWORD e32_fixupsize;
3088     DWORD e32_fixupsum;
3089     DWORD e32_ldrsize;
3090     DWORD e32_ldrsum;
3091     DWORD e32_objtab;
3092     DWORD e32_objcnt;
3093     DWORD e32_objmap;
3094     DWORD e32_itermap;
3095     DWORD e32_rsrctab;
3096     DWORD e32_rsrcnt;
3097     WORD e32_restab;
3098     WORD e32_enttab;
3099     WORD e32_dirtab;
3100     WORD e32_dircnt;
3101     WORD e32_fpagetab;
3102     WORD e32_frextab;
3103     WORD e32_impmod;
3104     WORD e32_impmodcnt;
3105     WORD e32_impproc;
3106     WORD e32_pagesum;
3107     WORD e32_datapage;
3108     WORD e32_preload;
3109     WORD e32_nrestab;
3110     WORD e32_cbnrestab;
3111     WORD e32_nressum;
3112     WORD e32_autodata;
3113     WORD e32_debuginfo;
3114     WORD e32_debuglen;
3115     WORD e32_instpreload;
3116     WORD e32_instdemand;
3117     WORD e32_heapsize;
3118     BYTE e32_res3[12];
3119     WORD e32_winresoff;
3120     WORD e32_winreslen;
3121     WORD e32_devid;
3122     WORD e32_ddkver;
3123 } IMAGE_VXD_HEADER, *PIMAGE_VXD_HEADER;
3124
3125
3126 /* These defines describe the meanings of the bits in the Characteristics
3127 field */
3128
3129 #define IMAGE_FILE_RELOCS_STRIPPED 0x0001 /* No relocation info */
3130 #define IMAGE_FILE_EXECUTABLE_IMAGE 0x0002
3131 #define IMAGE_FILE_LINE_NUMS_STRIPPED 0x0004
3132 #define IMAGE_FILE_LOCAL_SYMS_STRIPPED 0x0008
3133 #define IMAGE_FILE_16BIT_MACHINE 0x0040
3134 #define IMAGE_FILE_BYTES_REVERSED_LO 0x0080
3135 #define IMAGE_FILE_32BIT_MACHINE 0x0100
3136 #define IMAGE_FILE_DEBUG_STRIPPED 0x0200

```

```

3137 #define IMAGE_FILE_SYSTEM          0x1000
3138 #define IMAGE_FILE_DLL              0x2000
3139 #define IMAGE_FILE_BYTES_REVERSED_HI 0x8000
3140
3141 /* These are the settings of the Machine field. */
3142 #define IMAGE_FILE_MACHINE_UNKNOWN  0
3143 #define IMAGE_FILE_MACHINE_I860     0x14d
3144 #define IMAGE_FILE_MACHINE_I386     0x14c
3145 #define IMAGE_FILE_MACHINE_R3000    0x162
3146 #define IMAGE_FILE_MACHINE_R4000    0x166
3147 #define IMAGE_FILE_MACHINE_R10000   0x168
3148 #define IMAGE_FILE_MACHINE_ALPHA    0x184
3149 #define IMAGE_FILE_MACHINE_POWERPC  0x1f0
3150
3151 #define IMAGE_SIZEOF_FILE_HEADER    20
3152
3153 /* Possible Magic values */
3154 #define IMAGE_NT_OPTIONAL_HDR_MAGIC  0x10b
3155 #define IMAGE_ROM_OPTIONAL_HDR_MAGIC 0x107
3156
3157 /* These are indexes into the DataDirectory array */
3158 #define IMAGE_FILE_EXPORT_DIRECTORY  0
3159 #define IMAGE_FILE_IMPORT_DIRECTORY  1
3160 #define IMAGE_FILE_RESOURCE_DIRECTORY 2
3161 #define IMAGE_FILE_EXCEPTION_DIRECTORY 3
3162 #define IMAGE_FILE_SECURITY_DIRECTORY 4
3163 #define IMAGE_FILE_BASE_RELOCATION_TABLE 5
3164 #define IMAGE_FILE_DEBUG_DIRECTORY 6
3165 #define IMAGE_FILE_DESCRIPTION_STRING 7
3166 #define IMAGE_FILE_MACHINE_VALUE      8 /* Mips */
3167 #define IMAGE_FILE_THREAD_LOCAL_STORAGE 9
3168 #define IMAGE_FILE_CALLBACK_DIRECTORY 10
3169
3170 /* Directory Entries, indices into the DataDirectory array */
3171
3172 #define IMAGE_DIRECTORY_ENTRY_EXPORT 0
3173 #define IMAGE_DIRECTORY_ENTRY_IMPORT 1
3174 #define IMAGE_DIRECTORY_ENTRY_RESOURCE 2
3175 #define IMAGE_DIRECTORY_ENTRY_EXCEPTION 3
3176 #define IMAGE_DIRECTORY_ENTRY_SECURITY 4
3177 #define IMAGE_DIRECTORY_ENTRY_BASERELOC 5
3178 #define IMAGE_DIRECTORY_ENTRY_DEBUG 6
3179 #define IMAGE_DIRECTORY_ENTRY_COPYRIGHT 7
3180 #define IMAGE_DIRECTORY_ENTRY_GLOBALPTR 8 /* (MIPS GP) */
3181 #define IMAGE_DIRECTORY_ENTRY_TLS 9
3182 #define IMAGE_DIRECTORY_ENTRY_LOAD_CONFIG 10
3183 #define IMAGE_DIRECTORY_ENTRY_BOUND_IMPORT 11
3184 #define IMAGE_DIRECTORY_ENTRY_IAT 12 /* Import Address Table */
3185 #define IMAGE_DIRECTORY_ENTRY_DELAY_IMPORT 13
3186 #define IMAGE_DIRECTORY_ENTRY_COM_DESCRIPTOR 14
3187
3188 /* Subsystem Values */
3189
3190 #define IMAGE_SUBSYSTEM_UNKNOWN 0
3191 #define IMAGE_SUBSYSTEM_NATIVE 1
3192 #define IMAGE_SUBSYSTEM_WINDOWS_GUI 2 /* Windows GUI subsystem */
3193 #define IMAGE_SUBSYSTEM_WINDOWS_CUI 3 /* Windows character subsystem*/
3194 #define IMAGE_SUBSYSTEM_OS2_CUI 5
3195 #define IMAGE_SUBSYSTEM_POSIX_CUI 7
3196
3197 typedef struct _IMAGE_FILE_HEADER {
3198     WORD Machine;
3199     WORD NumberOfSections;
3200     DWORD TimeDateStamp;
3201     DWORD PointerToSymbolTable;
3202     DWORD NumberOfSymbols;
3203     WORD SizeOfOptionalHeader;
3204     WORD Characteristics;
3205 } IMAGE_FILE_HEADER, *PIMAGE_FILE_HEADER;
3206
3207 typedef struct _IMAGE_DATA_DIRECTORY {
3208     DWORD VirtualAddress;
3209     DWORD Size;
3210 } IMAGE_DATA_DIRECTORY, *PIMAGE_DATA_DIRECTORY;
3211
3212 #define IMAGE_NUMBEROF_DIRECTORY_ENTRIES 16
3213
3214 typedef struct _IMAGE_OPTIONAL_HEADER {
3215     /* Standard fields */
3216
3217     WORD Magic; /* 0x10b or 0x107 */ /* 0x00 */
3218     BYTE MajorLinkerVersion;
3219     BYTE MinorLinkerVersion;
3220     DWORD SizeOfCode;
3221     DWORD SizeOfInitializedData;
3222     DWORD SizeOfUninitializedData;

```

```

3224     DWORD AddressOfEntryPoint;          /* 0x10 */
3225     DWORD BaseOfCode;
3226     DWORD BaseOfData;
3227
3228     /* NT additional fields */
3229
3230     DWORD ImageBase;
3231     DWORD SectionAlignment;             /* 0x20 */
3232     DWORD FileAlignment;
3233     WORD MajorOperatingSystemVersion;
3234     WORD MinorOperatingSystemVersion;
3235     WORD MajorImageVersion;
3236     WORD MinorImageVersion;
3237     WORD MajorSubsystemVersion;         /* 0x30 */
3238     WORD MinorSubsystemVersion;
3239     DWORD Win32VersionValue;
3240     DWORD SizeOfImage;
3241     DWORD SizeOfHeaders;
3242     DWORD CheckSum;                     /* 0x40 */
3243     WORD Subsystem;
3244     WORD DllCharacteristics;
3245     DWORD SizeOfStackReserve;
3246     DWORD SizeOfStackCommit;
3247     DWORD SizeOfHeapReserve;            /* 0x50 */
3248     DWORD SizeOfHeapCommit;
3249     DWORD LoaderFlags;
3250     DWORD NumberOfRvaAndSizes;
3251     IMAGE_DATA_DIRECTORY DataDirectory[IMAGE_NUMBEROF_DIRECTORY_ENTRIES]; /* 0x60 */
3252 } IMAGE_OPTIONAL_HEADER, *PIMAGE_OPTIONAL_HEADER;
3253
3254 typedef struct _IMAGE_NT_HEADERS {
3255     DWORD Signature; /* "PE" */
3256     IMAGE_FILE_HEADER FileHeader;
3257     IMAGE_OPTIONAL_HEADER OptionalHeader;
3258 } IMAGE_NT_HEADERS, *PIMAGE_NT_HEADERS;
3259
3260 #define IMAGE_SIZEOF_SHORT_NAME 8
3261
3262 typedef struct _IMAGE_SECTION_HEADER {
3263     BYTE Name[IMAGE_SIZEOF_SHORT_NAME];
3264     union {
3265         DWORD PhysicalAddress;
3266         DWORD VirtualSize;
3267     } Misc;
3268     DWORD VirtualAddress;
3269     DWORD SizeOfRawData;
3270     DWORD PointerToRawData;
3271     DWORD PointerToRelocations;
3272     DWORD PointerToLinenumbers;
3273     WORD NumberOfRelocations;
3274     WORD NumberOfLinenumbers;
3275     DWORD Characteristics;
3276 } IMAGE_SECTION_HEADER, *PIMAGE_SECTION_HEADER;
3277
3278 #define IMAGE_SIZEOF_SECTION_HEADER 40
3279
3280 #define IMAGE_FIRST_SECTION(nthead) \
3281 ((PIMAGE_SECTION_HEADER) ((LPBYTE)&((PIMAGE_NT_HEADERS)(nthead))->OptionalHeader + \
3282 (PIMAGE_NT_HEADERS)(nthead)->FileHeader.SizeOfOptionalHeader))
3283
3284 /* These defines are for the Characteristics bitfield. */
3285 /* #define IMAGE_SCN_TYPE_REG 0x00000000 - Reserved */
3286 /* #define IMAGE_SCN_TYPE_DSECT 0x00000001 - Reserved */
3287 /* #define IMAGE_SCN_TYPE_NOLOAD 0x00000002 - Reserved */
3288 /* #define IMAGE_SCN_TYPE_GROUP 0x00000004 - Reserved */
3289 /* #define IMAGE_SCN_TYPE_NO_PAD 0x00000008 - Reserved */
3290 /* #define IMAGE_SCN_TYPE_COPY 0x00000010 - Reserved */
3291
3292 #define IMAGE_SCN_CNT_CODE 0x00000020
3293 #define IMAGE_SCN_CNT_INITIALIZED_DATA 0x00000040
3294 #define IMAGE_SCN_CNT_UNINITIALIZED_DATA 0x00000080
3295
3296 #define IMAGE_SCN_LNK_OTHER 0x00000100
3297 #define IMAGE_SCN_LNK_INFO 0x00000200
3298 /* #define IMAGE_SCN_TYPE_OVER 0x00000400 - Reserved */
3299 #define IMAGE_SCN_LNK_REMOVE 0x00000800
3300 #define IMAGE_SCN_LNK_COMDAT 0x00001000
3301
3302 /* 0x00002000 - Reserved */
3303 /* #define IMAGE_SCN_MEM_PROTECTED 0x00004000 - Obsolete */
3304 #define IMAGE_SCN_MEM_FARDATA 0x00008000
3305
3306 /* #define IMAGE_SCN_MEM_SYSHEAP 0x00010000 - Obsolete */
3307 #define IMAGE_SCN_MEM_PURGEABLE 0x00020000
3308 #define IMAGE_SCN_MEM_16BIT 0x00020000
3309 #define IMAGE_SCN_MEM_LOCKED 0x00040000
3310 #define IMAGE_SCN_MEM_PRELOAD 0x00080000

```

```

3311
3312 #define IMAGE_SCN_ALIGN_1BYTES      0x00100000
3313 #define IMAGE_SCN_ALIGN_2BYTES      0x00200000
3314 #define IMAGE_SCN_ALIGN_4BYTES      0x00300000
3315 #define IMAGE_SCN_ALIGN_8BYTES      0x00400000
3316 #define IMAGE_SCN_ALIGN_16BYTES     0x00500000 /* Default */
3317 #define IMAGE_SCN_ALIGN_32BYTES     0x00600000
3318 #define IMAGE_SCN_ALIGN_64BYTES     0x00700000
3319 /*          0x00800000 - Unused */
3320
3321 #define IMAGE_SCN_LNK_NRELOC_OVFL    0x01000000
3322
3323
3324 #define IMAGE_SCN_MEM_DISCARDABLE    0x02000000
3325 #define IMAGE_SCN_MEM_NOT_CACHED     0x04000000
3326 #define IMAGE_SCN_MEM_NOT_PAGED     0x08000000
3327 #define IMAGE_SCN_MEM_SHARED         0x10000000
3328 #define IMAGE_SCN_MEM_EXECUTE       0x20000000
3329 #define IMAGE_SCN_MEM_READ           0x40000000
3330 #define IMAGE_SCN_MEM_WRITE          0x80000000
3331
3332 #include "pshpack2.h"
3333
3334 typedef struct _IMAGE_SYMBOL {
3335     union {
3336         BYTE    ShortName[8];
3337         struct {
3338             DWORD    Short;
3339             DWORD    Long;
3340         } Name;
3341         DWORD    LongName[2];
3342     } N;
3343     DWORD    Value;
3344     SHORT    SectionNumber;
3345     WORD     Type;
3346     BYTE     StorageClass;
3347     BYTE     NumberOfAuxSymbols;
3348 } IMAGE_SYMBOL;
3349 typedef IMAGE_SYMBOL *PIMAGE_SYMBOL;
3350
3351 #define IMAGE_SIZEOF_SYMBOL 18
3352
3353 typedef struct _IMAGE_LINENUMBER {
3354     union {
3355         DWORD    SymbolTableIndex;
3356         DWORD    VirtualAddress;
3357     } Type;
3358     WORD     Linenumber;
3359 } IMAGE_LINENUMBER;
3360 typedef IMAGE_LINENUMBER *PIMAGE_LINENUMBER;
3361
3362 #define IMAGE_SIZEOF_LINENUMBER 6
3363
3364 typedef union _IMAGE_AUX_SYMBOL {
3365     struct {
3366         DWORD    TagIndex;
3367         union {
3368             struct {
3369                 WORD    Linenumber;
3370                 WORD    Size;
3371             } LnSz;
3372             DWORD    TotalSize;
3373         } Misc;
3374         union {
3375             struct {
3376                 DWORD    PointerToLinenumber;
3377                 DWORD    PointerToNextFunction;
3378             } Function;
3379             struct {
3380                 WORD     Dimension[4];
3381             } Array;
3382             } FcnAry;
3383         WORD     TvIndex;
3384     } Sym;
3385     struct {
3386         BYTE     Name[IMAGE_SIZEOF_SYMBOL];
3387     } File;
3388     struct {
3389         DWORD    Length;
3390         WORD     NumberOfRelocations;
3391         WORD     NumberOfLinenumbers;
3392         DWORD    CheckSum;
3393         SHORT    Number;
3394         BYTE     Selection;
3395     } Section;
3396 } IMAGE_AUX_SYMBOL;
3397 typedef IMAGE_AUX_SYMBOL *PIMAGE_AUX_SYMBOL;

```

```
3398
3399 #define IMAGE_SIZEOF_AUX_SYMBOL 18
3400
3401 #include "poppack.h"
3402
3403 #define IMAGE_SYM_UNDEFINED          (SHORT) 0
3404 #define IMAGE_SYM_ABSOLUTE          (SHORT) -1
3405 #define IMAGE_SYM_DEBUG             (SHORT) -2
3406
3407 #define IMAGE_SYM_TYPE_NULL          0x0000
3408 #define IMAGE_SYM_TYPE_VOID          0x0001
3409 #define IMAGE_SYM_TYPE_CHAR          0x0002
3410 #define IMAGE_SYM_TYPE_SHORT         0x0003
3411 #define IMAGE_SYM_TYPE_INT           0x0004
3412 #define IMAGE_SYM_TYPE_LONG          0x0005
3413 #define IMAGE_SYM_TYPE_FLOAT         0x0006
3414 #define IMAGE_SYM_TYPE_DOUBLE        0x0007
3415 #define IMAGE_SYM_TYPE_STRUCT        0x0008
3416 #define IMAGE_SYM_TYPE_UNION         0x0009
3417 #define IMAGE_SYM_TYPE_ENUM          0x000A
3418 #define IMAGE_SYM_TYPE_MOE          0x000B
3419 #define IMAGE_SYM_TYPE_BYTE          0x000C
3420 #define IMAGE_SYM_TYPE_WORD          0x000D
3421 #define IMAGE_SYM_TYPE_UINT          0x000E
3422 #define IMAGE_SYM_TYPE_DWORD         0x000F
3423 #define IMAGE_SYM_TYPE_PCODE         0x8000
3424
3425 #define IMAGE_SYM_DTYPE_NULL         0
3426 #define IMAGE_SYM_DTYPE_POINTER      1
3427 #define IMAGE_SYM_DTYPE_FUNCTION     2
3428 #define IMAGE_SYM_DTYPE_ARRAY        3
3429
3430 #define IMAGE_SYM_CLASS_END_OF_FUNCTION (BYTE) -1
3431 #define IMAGE_SYM_CLASS_NULL          0x0000
3432 #define IMAGE_SYM_CLASS_AUTOMATIC     0x0001
3433 #define IMAGE_SYM_CLASS_EXTERNAL      0x0002
3434 #define IMAGE_SYM_CLASS_STATIC        0x0003
3435 #define IMAGE_SYM_CLASS_REGISTER      0x0004
3436 #define IMAGE_SYM_CLASS_EXTERNAL_DEF  0x0005
3437 #define IMAGE_SYM_CLASS_LABEL         0x0006
3438 #define IMAGE_SYM_CLASS_UNDEFINED_LABEL 0x0007
3439 #define IMAGE_SYM_CLASS_MEMBER_OF_STRUCT 0x0008
3440 #define IMAGE_SYM_CLASS_ARGUMENT      0x0009
3441 #define IMAGE_SYM_CLASS_STRUCT_TAG    0x000A
3442 #define IMAGE_SYM_CLASS_MEMBER_OF_UNION 0x000B
3443 #define IMAGE_SYM_CLASS_UNION_TAG     0x000C
3444 #define IMAGE_SYM_CLASS_TYPE_DEFINITION 0x000D
3445 #define IMAGE_SYM_CLASS_UNDEFINED_STATIC 0x000E
3446 #define IMAGE_SYM_CLASS_ENUM_TAG      0x000F
3447 #define IMAGE_SYM_CLASS_MEMBER_OF_ENUM 0x0010
3448 #define IMAGE_SYM_CLASS_REGISTER_PARAM 0x0011
3449 #define IMAGE_SYM_CLASS_BIT_FIELD     0x0012
3450
3451 #define IMAGE_SYM_CLASS_FAR_EXTERNAL  0x0044
3452 #define IMAGE_SYM_CLASS_BLOCK         0x0064
3453 #define IMAGE_SYM_CLASS_FUNCTION      0x0065
3454 #define IMAGE_SYM_CLASS_END_OF_STRUCT 0x0066
3455 #define IMAGE_SYM_CLASS_FILE          0x0067
3456 #define IMAGE_SYM_CLASS_SECTION       0x0068
3457 #define IMAGE_SYM_CLASS_WEAK_EXTERNAL 0x0069
3458
3459 #define N_BTMASK                       0x000F
3460 #define N_TMASK                        0x0030
3461 #define N_TMASK1                       0x00C0
3462 #define N_TMASK2                       0x00F0
3463 #define N_BTSHFT                       4
3464 #define N_TSHFT                        2
3465
3466 #define BTYPE(x) ((x) & N_BTMASK)
3467
3468 #ifndef ISPTR
3469 #define ISPTR(x) (((x) & N_TMASK) == (IMAGE_SYM_DTYPE_POINTER << N_BTSHFT))
3470 #endif
3471
3472 #ifndef ISFCN
3473 #define ISFCN(x) (((x) & N_TMASK) == (IMAGE_SYM_DTYPE_FUNCTION << N_BTSHFT))
3474 #endif
3475
3476 #ifndef ISARY
3477 #define ISARY(x) (((x) & N_TMASK) == (IMAGE_SYM_DTYPE_ARRAY << N_BTSHFT))
3478 #endif
3479
3480 #ifndef ISTAG
3481 #define ISTAG(x) ((x) == IMAGE_SYM_CLASS_STRUCT_TAG || (x) == IMAGE_SYM_CLASS_UNION_TAG ||
3482 (x) == IMAGE_SYM_CLASS_ENUM_TAG)
3483 #endif
```

```

3484 #ifndef INCREMENT
3485 #define INCREMENT(x) (((x)&~N_BTMASK)<<N_TSHIFT) | (IMAGE_SYM_DTYPE_POINTER<<N_BTSHIFT) | ((x)&N_BTMASK)
3486 #endif
3487 #ifndef DECREMENT
3488 #define DECREMENT(x) (((x)>>N_TSHIFT)&~N_BTMASK) | ((x)&N_BTMASK)
3489 #endif
3490
3491 #define IMAGE_COMDAT_SELECT_NODUPPLICATES 1
3492 #define IMAGE_COMDAT_SELECT_ANY 2
3493 #define IMAGE_COMDAT_SELECT_SAME_SIZE 3
3494 #define IMAGE_COMDAT_SELECT_EXACT_MATCH 4
3495 #define IMAGE_COMDAT_SELECT_ASSOCIATIVE 5
3496 #define IMAGE_COMDAT_SELECT_LARGEST 6
3497 #define IMAGE_COMDAT_SELECT_NEWEST 7
3498
3499 #define IMAGE_WEAK_EXTERN_SEARCH_NOLIBRARY 1
3500 #define IMAGE_WEAK_EXTERN_SEARCH_LIBRARY 2
3501 #define IMAGE_WEAK_EXTERN_SEARCH_ALIAS 3
3502
3503 /* Export module directory */
3504
3505 typedef struct _IMAGE_EXPORT_DIRECTORY {
3506     DWORD Characteristics;
3507     DWORD TimeDateStamp;
3508     WORD MajorVersion;
3509     WORD MinorVersion;
3510     DWORD Name;
3511     DWORD Base;
3512     DWORD NumberOfFunctions;
3513     DWORD NumberOfNames;
3514     DWORD AddressOfFunctions;
3515     DWORD AddressOfNames;
3516     DWORD AddressOfNameOrdinals;
3517 } IMAGE_EXPORT_DIRECTORY, *PIMAGE_EXPORT_DIRECTORY;
3518
3519 /* Import name entry */
3520 typedef struct _IMAGE_IMPORT_BY_NAME {
3521     WORD Hint;
3522     BYTE Name[1];
3523 } IMAGE_IMPORT_BY_NAME, *PIMAGE_IMPORT_BY_NAME;
3524
3525 /* Import thunk */
3526 typedef struct _IMAGE_THUNK_DATA {
3527     union {
3528         LPBYTE ForwarderString;
3529         PDWORD Function;
3530         DWORD Ordinal;
3531         PIMAGE_IMPORT_BY_NAME AddressOfData;
3532     } u1;
3533 } IMAGE_THUNK_DATA, *PIMAGE_THUNK_DATA;
3534
3535 /* Import module directory */
3536
3537 typedef struct _IMAGE_IMPORT_DESCRIPTOR {
3538     union {
3539         DWORD Characteristics; /* 0 for terminating null import descriptor */
3540         PIMAGE_THUNK_DATA OriginalFirstThunk; /* RVA to original unbound IAT */
3541     } u;
3542     DWORD TimeDateStamp; /* 0 if not bound,
3543 * -1 if bound, and real date\time stamp
3544 * in IMAGE_DIRECTORY_ENTRY_BOUND_IMPORT
3545 * (new BIND)
3546 * otherwise date/time stamp of DLL bound to
3547 * (Old BIND)
3548 */
3549     DWORD ForwarderChain; /* -1 if no forwarders */
3550     DWORD Name;
3551     /* RVA to IAT (if bound this IAT has actual addresses) */
3552     PIMAGE_THUNK_DATA FirstThunk;
3553 } IMAGE_IMPORT_DESCRIPTOR, *PIMAGE_IMPORT_DESCRIPTOR;
3554
3555 #define IMAGE_ORDINAL_FLAG 0x80000000
3556 #define IMAGE_SNAP_BY_ORDINAL(Ordinal) ((Ordinal & IMAGE_ORDINAL_FLAG) != 0)
3557 #define IMAGE_ORDINAL(Ordinal) (Ordinal & 0xffff)
3558
3559 typedef struct _IMAGE_BOUND_IMPORT_DESCRIPTOR {
3560 {
3561     DWORD TimeDateStamp;
3562     WORD OffsetModuleName;
3563     WORD NumberOfModuleForwarderRefs;
3564 /* Array of zero or more IMAGE_BOUND_FORWARDER_REF follows */
3565 } IMAGE_BOUND_IMPORT_DESCRIPTOR, *PIMAGE_BOUND_IMPORT_DESCRIPTOR;
3566
3567 typedef struct _IMAGE_BOUND_FORWARDER_REF {
3568 {
3569     DWORD TimeDateStamp;
3570     WORD OffsetModuleName;

```

```

3571     WORD        Reserved;
3572 } IMAGE_BOUND_FORWARDER_REF, *PIMAGE_BOUND_FORWARDER_REF;
3573
3574 #include "pshpack2.h"
3575
3576 typedef struct _IMAGE_BASE_RELOCATION
3577 {
3578     DWORD        VirtualAddress;
3579     DWORD        SizeOfBlock;
3580     /* WORD TypeOffset[1]; */
3581 } IMAGE_BASE_RELOCATION, *PIMAGE_BASE_RELOCATION;
3582
3583 typedef struct _IMAGE_RELOCATION
3584 {
3585     union {
3586         DWORD        VirtualAddress;
3587         DWORD        RelocCount;
3588     } u;
3589     DWORD        SymbolTableIndex;
3590     WORD        Type;
3591 } IMAGE_RELOCATION;
3592 typedef IMAGE_RELOCATION *PIMAGE_RELOCATION;
3593
3594 #include "poppack.h"
3595
3596 #define IMAGE_SIZEOF_RELOCATION 10
3597
3598 /* generic relocation types */
3599 #define IMAGE_REL_BASED_ABSOLUTE      0
3600 #define IMAGE_REL_BASED_HIGH         1
3601 #define IMAGE_REL_BASED_LOW          2
3602 #define IMAGE_REL_BASED_HIGHLOW      3
3603 #define IMAGE_REL_BASED_HIGHADJ      4
3604 #define IMAGE_REL_BASED_MIPS_JMPADDR 5
3605 #define IMAGE_REL_BASED_SECTION      6
3606 #define IMAGE_REL_BASED_REL          7
3607 #define IMAGE_REL_BASED_MIPS_JMPADDR16 9
3608 #define IMAGE_REL_BASED_IA64_IMM64   9 /* yes, 9 too */
3609 #define IMAGE_REL_BASED_DIR64        10
3610 #define IMAGE_REL_BASED_HIGH3ADJ     11
3611
3612 /* I386 relocation types */
3613 #define IMAGE_REL_I386_ABSOLUTE      0
3614 #define IMAGE_REL_I386_DIR16         1
3615 #define IMAGE_REL_I386_REL16         2
3616 #define IMAGE_REL_I386_DIR32         6
3617 #define IMAGE_REL_I386_DIR32NB       7
3618 #define IMAGE_REL_I386_SEG12         9
3619 #define IMAGE_REL_I386_SECTION      10
3620 #define IMAGE_REL_I386_SECREL       11
3621 #define IMAGE_REL_I386_REL32        20
3622
3623 /* MIPS relocation types */
3624 #define IMAGE_REL_MIPS_ABSOLUTE      0x0000
3625 #define IMAGE_REL_MIPS_REFHALF       0x0001
3626 #define IMAGE_REL_MIPS_REFWORD       0x0002
3627 #define IMAGE_REL_MIPS_JMPADDR       0x0003
3628 #define IMAGE_REL_MIPS_REFHI         0x0004
3629 #define IMAGE_REL_MIPS_REFLO         0x0005
3630 #define IMAGE_REL_MIPS_GPREL         0x0006
3631 #define IMAGE_REL_MIPS_LITERAL       0x0007
3632 #define IMAGE_REL_MIPS_SECTION       0x000A
3633 #define IMAGE_REL_MIPS_SECREL        0x000B
3634 #define IMAGE_REL_MIPS_SECRELLO      0x000C
3635 #define IMAGE_REL_MIPS_SECRELHI      0x000D
3636 #define IMAGE_REL_MIPS_JMPADDR16     0x0010
3637 #define IMAGE_REL_MIPS_REFWORDNB     0x0022
3638 #define IMAGE_REL_MIPS_PAIR          0x0025
3639
3640 /* ALPHA relocation types */
3641 #define IMAGE_REL_ALPHA_ABSOLUTE     0x0000
3642 #define IMAGE_REL_ALPHA_REFLONG      0x0001
3643 #define IMAGE_REL_ALPHA_REFQUAD      0x0002
3644 #define IMAGE_REL_ALPHA_GPREL        0x0003
3645 #define IMAGE_REL_ALPHA_LITERAL      0x0004
3646 #define IMAGE_REL_ALPHA_LITUSE       0x0005
3647 #define IMAGE_REL_ALPHA_GPDISP       0x0006
3648 #define IMAGE_REL_ALPHA_BRADDR       0x0007
3649 #define IMAGE_REL_ALPHA_HINT         0x0008
3650 #define IMAGE_REL_ALPHA_INLINE_REFLONG 0x0009
3651 #define IMAGE_REL_ALPHA_REFHI        0x000A
3652 #define IMAGE_REL_ALPHA_REFLO        0x000B
3653 #define IMAGE_REL_ALPHA_PAIR         0x000C
3654 #define IMAGE_REL_ALPHA_MATCH        0x000D
3655 #define IMAGE_REL_ALPHA_SECTION      0x000E
3656 #define IMAGE_REL_ALPHA_SECREL       0x000F
3657 #define IMAGE_REL_ALPHA_REFLONGNB    0x0010

```



```

3658 #define IMAGE_REL_ALPHA_SECRELLO 0x0011
3659 #define IMAGE_REL_ALPHA_SECRELHI 0x0012
3660 #define IMAGE_REL_ALPHA_REFQ3 0x0013
3661 #define IMAGE_REL_ALPHA_REFQ2 0x0014
3662 #define IMAGE_REL_ALPHA_REFQ1 0x0015
3663 #define IMAGE_REL_ALPHA_GPRELLO 0x0016
3664 #define IMAGE_REL_ALPHA_GPRELHI 0x0017
3665
3666 /* PowerPC relocation types */
3667 #define IMAGE_REL_PPC_ABSOLUTE 0x0000
3668 #define IMAGE_REL_PPC_ADDR64 0x0001
3669 #define IMAGE_REL_PPC_ADDR 0x0002
3670 #define IMAGE_REL_PPC_ADDR24 0x0003
3671 #define IMAGE_REL_PPC_ADDRL6 0x0004
3672 #define IMAGE_REL_PPC_ADDRL4 0x0005
3673 #define IMAGE_REL_PPC_REL24 0x0006
3674 #define IMAGE_REL_PPC_REL14 0x0007
3675 #define IMAGE_REL_PPC_TOCREL16 0x0008
3676 #define IMAGE_REL_PPC_TOCREL14 0x0009
3677 #define IMAGE_REL_PPC_ADDR32NB 0x000A
3678 #define IMAGE_REL_PPC_SECREL 0x000B
3679 #define IMAGE_REL_PPC_SECTION 0x000C
3680 #define IMAGE_REL_PPC_IFGLUE 0x000D
3681 #define IMAGE_REL_PPC_IMGLUE 0x000E
3682 #define IMAGE_REL_PPC_SECREL16 0x000F
3683 #define IMAGE_REL_PPC_REFHI 0x0010
3684 #define IMAGE_REL_PPC_REFLO 0x0011
3685 #define IMAGE_REL_PPC_PAIR 0x0012
3686 #define IMAGE_REL_PPC_SECRELLO 0x0013
3687 #define IMAGE_REL_PPC_SECRELHI 0x0014
3688 #define IMAGE_REL_PPC_GPREL 0x0015
3689 #define IMAGE_REL_PPC_TYPMASK 0x00FF
3690 /* modifier bits */
3691 #define IMAGE_REL_PPC_NEG 0x0100
3692 #define IMAGE_REL_PPC_BRTAKEN 0x0200
3693 #define IMAGE_REL_PPC_BRNTAKEN 0x0400
3694 #define IMAGE_REL_PPC_TOCDEFN 0x0800
3695
3696 /* SH3 ? relocation type */
3697 #define IMAGE_REL_SH3_ABSOLUTE 0x0000
3698 #define IMAGE_REL_SH3_DIRECT16 0x0001
3699 #define IMAGE_REL_SH3_DIRECT 0x0002
3700 #define IMAGE_REL_SH3_DIRECT8 0x0003
3701 #define IMAGE_REL_SH3_DIRECT8_WORD 0x0004
3702 #define IMAGE_REL_SH3_DIRECT8_LONG 0x0005
3703 #define IMAGE_REL_SH3_DIRECT4 0x0006
3704 #define IMAGE_REL_SH3_DIRECT4_WORD 0x0007
3705 #define IMAGE_REL_SH3_DIRECT4_LONG 0x0008
3706 #define IMAGE_REL_SH3_PCREL8_WORD 0x0009
3707 #define IMAGE_REL_SH3_PCREL8_LONG 0x000A
3708 #define IMAGE_REL_SH3_PCREL12_WORD 0x000B
3709 #define IMAGE_REL_SH3_STARTOF_SECTION 0x000C
3710 #define IMAGE_REL_SH3_SIZEOF_SECTION 0x000D
3711 #define IMAGE_REL_SH3_SECTION 0x000E
3712 #define IMAGE_REL_SH3_SECREL 0x000F
3713 #define IMAGE_REL_SH3_DIRECT32_NB 0x0010
3714
3715 /* ARM (Archimedes?) relocation types */
3716 #define IMAGE_REL_ARM_ABSOLUTE 0x0000
3717 #define IMAGE_REL_ARM_ADDR 0x0001
3718 #define IMAGE_REL_ARM_ADDR32NB 0x0002
3719 #define IMAGE_REL_ARM_BRANCH24 0x0003
3720 #define IMAGE_REL_ARM_BRANCH11 0x0004
3721 #define IMAGE_REL_ARM_SECTION 0x000E
3722 #define IMAGE_REL_ARM_SECREL 0x000F
3723
3724 /* IA64 relocation types */
3725 #define IMAGE_REL_IA64_ABSOLUTE 0x0000
3726 #define IMAGE_REL_IA64_IMM14 0x0001
3727 #define IMAGE_REL_IA64_IMM22 0x0002
3728 #define IMAGE_REL_IA64_IMM64 0x0003
3729 #define IMAGE_REL_IA64_DIR 0x0004
3730 #define IMAGE_REL_IA64_DIR64 0x0005
3731 #define IMAGE_REL_IA64_PCREL21B 0x0006
3732 #define IMAGE_REL_IA64_PCREL21M 0x0007
3733 #define IMAGE_REL_IA64_PCREL21F 0x0008
3734 #define IMAGE_REL_IA64_GPREL22 0x0009
3735 #define IMAGE_REL_IA64_LTOFF22 0x000A
3736 #define IMAGE_REL_IA64_SECTION 0x000B
3737 #define IMAGE_REL_IA64_SECREL22 0x000C
3738 #define IMAGE_REL_IA64_SECREL64I 0x000D
3739 #define IMAGE_REL_IA64_SECREL 0x000E
3740 #define IMAGE_REL_IA64_LTOFF64 0x000F
3741 #define IMAGE_REL_IA64_DIR32NB 0x0010
3742 #define IMAGE_REL_IA64_RESERVED_11 0x0011
3743 #define IMAGE_REL_IA64_RESERVED_12 0x0012
3744 #define IMAGE_REL_IA64_RESERVED_13 0x0013

```

```

3745 #define IMAGE_REL_IA64_RESERVED_14 0x0014
3746 #define IMAGE_REL_IA64_RESERVED_15 0x0015
3747 #define IMAGE_REL_IA64_RESERVED_16 0x0016
3748 #define IMAGE_REL_IA64_ADDEND 0x001F
3749
3750 /* archive format */
3751
3752 #define IMAGE_ARCHIVE_START_SIZE 8
3753 #define IMAGE_ARCHIVE_START "!<arch>\n"
3754 #define IMAGE_ARCHIVE_END "\n"
3755 #define IMAGE_ARCHIVE_PAD "\n"
3756 #define IMAGE_ARCHIVE_LINKER_MEMBER "/"
3757 #define IMAGE_ARCHIVE_LONGNAMES_MEMBER "//"
3758
3759 typedef struct _IMAGE_ARCHIVE_MEMBER_HEADER
3760 {
3761     BYTE    Name[16];
3762     BYTE    Date[12];
3763     BYTE    UserID[6];
3764     BYTE    GroupID[6];
3765     BYTE    Mode[8];
3766     BYTE    Size[10];
3767     BYTE    EndHeader[2];
3768 } IMAGE_ARCHIVE_MEMBER_HEADER, *PIMAGE_ARCHIVE_MEMBER_HEADER;
3769
3770 #define IMAGE_SIZEOF_ARCHIVE_MEMBER_HDR 60
3771
3772 /*
3773  * Resource directory stuff
3774  */
3775 typedef struct _IMAGE_RESOURCE_DIRECTORY {
3776     DWORD    Characteristics;
3777     DWORD    TimeDateStamp;
3778     WORD     MajorVersion;
3779     WORD     MinorVersion;
3780     WORD     NumberOfNamedEntries;
3781     WORD     NumberOfIdEntries;
3782     /* IMAGE_RESOURCE_DIRECTORY_ENTRY DirectoryEntries[]; */
3783 } IMAGE_RESOURCE_DIRECTORY, *PIMAGE_RESOURCE_DIRECTORY;
3784
3785 #define IMAGE_RESOURCE_NAME_IS_STRING 0x80000000
3786 #define IMAGE_RESOURCE_DATA_IS_DIRECTORY 0x80000000
3787
3788 typedef struct _IMAGE_RESOURCE_DIRECTORY_ENTRY {
3789     union u1 {
3790         struct fleegle {
3791             #ifdef BITFIELDS_BIGENDIAN
3792                 unsigned NameIsString:1;
3793                 unsigned NameOffset:31;
3794             #else
3795                 unsigned NameOffset:31;
3796                 unsigned NameIsString:1;
3797             #endif
3798         } DUMMYSTRUCTNAME1;
3799         DWORD    Name;
3800         struct sneegle {
3801             #ifdef WORDS_BIGENDIAN
3802                 WORD    __pad;
3803                 WORD    Id;
3804             #else
3805                 WORD    Id;
3806                 WORD    __pad;
3807             #endif
3808         } DUMMYSTRUCTNAME2;
3809     } DUMMYUNIONNAME1;
3810     union u2 {
3811         DWORD    OffsetToData;
3812         struct drooper {
3813             #ifdef BITFIELDS_BIGENDIAN
3814                 unsigned DataIsDirectory:1;
3815                 unsigned OffsetToDirectory:31;
3816             #else
3817                 unsigned OffsetToDirectory:31;
3818                 unsigned DataIsDirectory:1;
3819             #endif
3820         } DUMMYSTRUCTNAME3;
3821     } DUMMYUNIONNAME2;
3822 } IMAGE_RESOURCE_DIRECTORY_ENTRY, *PIMAGE_RESOURCE_DIRECTORY_ENTRY;
3823
3824
3825 typedef struct _IMAGE_RESOURCE_DIRECTORY_STRING {
3826     WORD    Length;
3827     CHAR    NameString[ 1 ];
3828 } IMAGE_RESOURCE_DIRECTORY_STRING, *PIMAGE_RESOURCE_DIRECTORY_STRING;
3829
3830 typedef struct _IMAGE_RESOURCE_DIR_STRING_U {
3831     WORD    Length;

```

```
3832     WCHAR    NameString[ 1 ];
3833 } IMAGE_RESOURCE_DIR_STRING_U, *PIMAGE_RESOURCE_DIR_STRING_U;
3834
3835 typedef struct _IMAGE_RESOURCE_DATA_ENTRY {
3836     DWORD    OffsetToData;
3837     DWORD    Size;
3838     DWORD    CodePage;
3839     DWORD    ResourceHandle;
3840 } IMAGE_RESOURCE_DATA_ENTRY, *PIMAGE_RESOURCE_DATA_ENTRY;
3841
3842
3843 typedef VOID CALLBACK (*PIMAGE_TLS_CALLBACK) (
3844     LPVOID DllHandle, DWORD Reason, LPVOID Reserved
3845 );
3846
3847 typedef struct _IMAGE_TLS_DIRECTORY {
3848     DWORD    StartAddressOfRawData;
3849     DWORD    EndAddressOfRawData;
3850     LPDWORD  AddressOfIndex;
3851     PIMAGE_TLS_CALLBACK *AddressOfCallBacks;
3852     DWORD    SizeOfZeroFill;
3853     DWORD    Characteristics;
3854 } IMAGE_TLS_DIRECTORY, *PIMAGE_TLS_DIRECTORY;
3855
3856 typedef struct _IMAGE_DEBUG_DIRECTORY {
3857     DWORD    Characteristics;
3858     DWORD    TimeDateStamp;
3859     WORD     MajorVersion;
3860     WORD     MinorVersion;
3861     DWORD    Type;
3862     DWORD    SizeOfData;
3863     DWORD    AddressOfRawData;
3864     DWORD    PointerToRawData;
3865 } IMAGE_DEBUG_DIRECTORY, *PIMAGE_DEBUG_DIRECTORY;
3866
3867 #define IMAGE_DEBUG_TYPE_UNKNOWN        0
3868 #define IMAGE_DEBUG_TYPE_COFF           1
3869 #define IMAGE_DEBUG_TYPE_CODEVIEW       2
3870 #define IMAGE_DEBUG_TYPE_FPO            3
3871 #define IMAGE_DEBUG_TYPE_MISC           4
3872 #define IMAGE_DEBUG_TYPE_EXCEPTION      5
3873 #define IMAGE_DEBUG_TYPE_FIXUP          6
3874 #define IMAGE_DEBUG_TYPE_OMAP_TO_SRC    7
3875 #define IMAGE_DEBUG_TYPE_OMAP_FROM_SRC  8
3876 #define IMAGE_DEBUG_TYPE_BORLAND        9
3877 #define IMAGE_DEBUG_TYPE_RESERVED10     10
3878
3879 typedef struct _IMAGE_COFF_SYMBOLS_HEADER {
3880     DWORD    NumberOfSymbols;
3881     DWORD    LvaToFirstSymbol;
3882     DWORD    NumberOfLinenumbers;
3883     DWORD    LvaToFirstLinenumber;
3884     DWORD    RvaToFirstByteOfCode;
3885     DWORD    RvaToLastByteOfCode;
3886     DWORD    RvaToFirstByteOfData;
3887     DWORD    RvaToLastByteOfData;
3888 } IMAGE_COFF_SYMBOLS_HEADER, *PIMAGE_COFF_SYMBOLS_HEADER;
3889
3890 #define FRAME_FPO            0
3891 #define FRAME_TRAP           1
3892 #define FRAME_TSS            2
3893 #define FRAME_NONFPO         3
3894
3895 typedef struct _FPO_DATA {
3896     DWORD    ulOffsetStart;
3897     DWORD    cbProcSize;
3898     DWORD    cdwLocals;
3899     WORD     cdwParams;
3900     unsigned cbProlog : 8;
3901     unsigned cbRegs : 3;
3902     unsigned fHasSEH : 1;
3903     unsigned fUseBP : 1;
3904     unsigned reserved : 1;
3905     unsigned cbFrame : 2;
3906 } FPO_DATA, *PFPO_DATA;
3907
3908 typedef struct _IMAGE_LOAD_CONFIG_DIRECTORY {
3909     DWORD    Characteristics;
3910     DWORD    TimeDateStamp;
3911     WORD     MajorVersion;
3912     WORD     MinorVersion;
3913     DWORD    GlobalFlagsClear;
3914     DWORD    GlobalFlagsSet;
3915     DWORD    CriticalSectionDefaultTimeout;
3916     DWORD    DeCommitFreeBlockThreshold;
3917     DWORD    DeCommitTotalFreeThreshold;
3918     PVOID    LockPrefixTable;
```

```

3919     DWORD MaximumAllocationSize;
3920     DWORD VirtualMemoryThreshold;
3921     DWORD ProcessHeapFlags;
3922     DWORD ProcessAffinityMask;
3923     WORD   CSDVersion;
3924     WORD   Reserved1;
3925     PVOID EditList;
3926     DWORD Reserved[1];
3927 } IMAGE_LOAD_CONFIG_DIRECTORY, *PIMAGE_LOAD_CONFIG_DIRECTORY;
3928
3929 typedef struct _IMAGE_FUNCTION_ENTRY {
3930     DWORD StartingAddress;
3931     DWORD EndingAddress;
3932     DWORD EndOfPrologue;
3933 } IMAGE_FUNCTION_ENTRY, *PIMAGE_FUNCTION_ENTRY;
3934
3935 #define IMAGE_DEBUG_MISC_EXENAME    1
3936
3937 typedef struct _IMAGE_DEBUG_MISC {
3938     DWORD    DataType;
3939     DWORD    Length;
3940     BYTE     Unicode;
3941     BYTE     Reserved[ 3 ];
3942     BYTE     Data[ 1 ];
3943 } IMAGE_DEBUG_MISC, *PIMAGE_DEBUG_MISC;
3944
3945 /* This is the structure that appears at the very start of a .DBG file. */
3946
3947 typedef struct _IMAGE_SEPARATE_DEBUG_HEADER {
3948     WORD    Signature;
3949     WORD    Flags;
3950     WORD    Machine;
3951     WORD    Characteristics;
3952     DWORD   TimeDateStamp;
3953     DWORD   CheckSum;
3954     DWORD   ImageBase;
3955     DWORD   SizeOfImage;
3956     DWORD   NumberOfSections;
3957     DWORD   ExportedNamesSize;
3958     DWORD   DebugDirectorySize;
3959     DWORD   SectionAlignment;
3960     DWORD   Reserved[ 2 ];
3961 } IMAGE_SEPARATE_DEBUG_HEADER, *PIMAGE_SEPARATE_DEBUG_HEADER;
3962
3963 #define IMAGE_SEPARATE_DEBUG_SIGNATURE 0x4944
3964
3965
3966 typedef struct tagMESSAGE_RESOURCE_ENTRY {
3967     WORD    Length;
3968     WORD    Flags;
3969     BYTE     Text[1];
3970 } MESSAGE_RESOURCE_ENTRY, *PMESSAGE_RESOURCE_ENTRY;
3971 #define MESSAGE_RESOURCE_UNICODE    0x0001
3972
3973 typedef struct tagMESSAGE_RESOURCE_BLOCK {
3974     DWORD    LowId;
3975     DWORD    HighId;
3976     DWORD    OffsetToEntries;
3977 } MESSAGE_RESOURCE_BLOCK, *PMESSAGE_RESOURCE_BLOCK;
3978
3979 typedef struct tagMESSAGE_RESOURCE_DATA {
3980     DWORD    NumberOfBlocks;
3981     MESSAGE_RESOURCE_BLOCK  Blocks[ 1 ];
3982 } MESSAGE_RESOURCE_DATA, *PMESSAGE_RESOURCE_DATA;
3983
3984 /*
3985  * Here follows typedefs for security and tokens.
3986  */
3987
3988 /*
3989  * First a constant for the following typedefs.
3990  */
3991
3992 #define ANYSIZE_ARRAY    1
3993
3994 /* FIXME: Orphan.    What does it point to? */
3995 typedef PVOID PACCESS_TOKEN;
3996
3997 /*
3998  * TOKEN_INFORMATION_CLASS
3999  */
4000
4001 typedef enum _TOKEN_INFORMATION_CLASS {
4002     TokenUser = 1,
4003     TokenGroups,
4004     TokenPrivileges,
4005     TokenOwner,

```

```
4006 TokenPrimaryGroup,
4007 TokenDefaultDacl,
4008 TokenSource,
4009 TokenType,
4010 TokenImpersonationLevel,
4011 TokenStatistics
4012 } TOKEN_INFORMATION_CLASS;
4013
4014 #define TOKEN_TOKEN_ADJUST_DEFAULT 0x0080
4015 #define TOKEN_ADJUST_GROUPS 0x0040
4016 #define TOKEN_ADJUST_PRIVILEGES 0x0020
4017 #define TOKEN_ADJUST_SESSIONID 0x0100
4018 #define TOKEN_ASSIGN_PRIMARY 0x0001
4019 #define TOKEN_DUPLICATE 0x0002
4020 #define TOKEN_EXECUTE STANDARD_RIGHTS_EXECUTE
4021 #define TOKEN_IMPERSONATE 0x0004
4022 #define TOKEN_QUERY 0x0008
4023 #define TOKEN_QUERY_SOURCE 0x0010
4024 #define TOKEN_ADJUST_DEFAULT 0x0080
4025 #define TOKEN_READ (STANDARD_RIGHTS_READ|TOKEN_QUERY)
4026 #define TOKEN_WRITE (STANDARD_RIGHTS_WRITE | \
4027 TOKEN_ADJUST_PRIVILEGES | \
4028 TOKEN_ADJUST_GROUPS | \
4029 TOKEN_ADJUST_DEFAULT )
4030 #define TOKEN_ALL_ACCESS (STANDARD_RIGHTS_REQUIRED | \
4031 TOKEN_ASSIGN_PRIMARY | \
4032 TOKEN_DUPLICATE | \
4033 TOKEN_IMPERSONATE | \
4034 TOKEN_QUERY | \
4035 TOKEN_QUERY_SOURCE | \
4036 TOKEN_ADJUST_PRIVILEGES | \
4037 TOKEN_ADJUST_GROUPS | \
4038 TOKEN_ADJUST_SESSIONID | \
4039 TOKEN_ADJUST_DEFAULT )
4040
4041 #ifndef _SECURITY_DEFINED
4042 #define _SECURITY_DEFINED
4043
4044
4045 typedef DWORD ACCESS_MASK, *PACCESS_MASK;
4046
4047 typedef struct _GENERIC_MAPPING {
4048     ACCESS_MASK GenericRead;
4049     ACCESS_MASK GenericWrite;
4050     ACCESS_MASK GenericExecute;
4051     ACCESS_MASK GenericAll;
4052 } GENERIC_MAPPING, *PGENERIC_MAPPING;
4053
4054 #ifndef SID_IDENTIFIER_AUTHORITY_DEFINED
4055 #define SID_IDENTIFIER_AUTHORITY_DEFINED
4056 typedef struct {
4057     BYTE Value[6];
4058 } SID_IDENTIFIER_AUTHORITY, *PSID_IDENTIFIER_AUTHORITY, *LPSID_IDENTIFIER_AUTHORITY;
4059 #endif /* !defined(SID_IDENTIFIER_AUTHORITY_DEFINED) */
4060
4061 #ifndef SID_DEFINED
4062 #define SID_DEFINED
4063 typedef struct _SID {
4064     BYTE Revision;
4065     BYTE SubAuthorityCount;
4066     SID_IDENTIFIER_AUTHORITY IdentifierAuthority;
4067     DWORD SubAuthority[1];
4068 } SID, *PSID;
4069 #endif /* !defined(SID_DEFINED) */
4070
4071 #define SID_REVISION (1) /* Current revision */
4072 #define SID_MAX_SUB_AUTHORITIES (15) /* current max subauths */
4073 #define SID_RECOMMENDED_SUB_AUTHORITIES (1) /* recommended subauths */
4074
4075
4076 /*
4077 * ACL
4078 */
4079
4080 #define ACL_REVISION1 1
4081 #define ACL_REVISION2 2
4082 #define ACL_REVISION3 3
4083 #define ACL_REVISION4 4
4084
4085 #define MIN_ACL_REVISION ACL_REVISION2
4086 #define MAX_ACL_REVISION ACL_REVISION4
4087
4088 typedef struct _ACL {
4089     BYTE AclRevision;
4090     BYTE Sbz1;
4091     WORD AclSize;
4092     WORD AceCount;
```

```

4093     WORD Sbz2;
4094 } ACL, *PACL;
4095
4096 /* SECURITY_DESCRIPTOR */
4097 #define SECURITY_DESCRIPTOR_REVISION    1
4098 #define SECURITY_DESCRIPTOR_REVISION1   1
4099
4100
4101 #define SE_OWNER_DEFAULTED    0x0001
4102 #define SE_GROUP_DEFAULTED    0x0002
4103 #define SE_DACL_PRESENT       0x0004
4104 #define SE_DACL_DEFAULTED     0x0008
4105 #define SE_SACL_PRESENT       0x0010
4106 #define SE_SACL_DEFAULTED     0x0020
4107 #define SE_SELF_RELATIVE      0x8000
4108
4109 typedef DWORD SECURITY_INFORMATION, *PSECURITY_INFORMATION;
4110 typedef WORD SECURITY_DESCRIPTOR_CONTROL, *PSECURITY_DESCRIPTOR_CONTROL;
4111
4112 /* The security descriptor structure */
4113 typedef struct {
4114     BYTE Revision;
4115     BYTE Sbz1;
4116     SECURITY_DESCRIPTOR_CONTROL Control;
4117     DWORD Owner;
4118     DWORD Group;
4119     DWORD Sacl;
4120     DWORD Dacl;
4121 } SECURITY_DESCRIPTOR_RELATIVE, *PISECURITY_DESCRIPTOR_RELATIVE;
4122
4123 typedef struct {
4124     BYTE Revision;
4125     BYTE Sbz1;
4126     SECURITY_DESCRIPTOR_CONTROL Control;
4127     PSID Owner;
4128     PSID Group;
4129     PACL Sacl;
4130     PACL Dacl;
4131 } SECURITY_DESCRIPTOR, *PSECURITY_DESCRIPTOR;
4132
4133 #define SECURITY_DESCRIPTOR_MIN_LENGTH    (sizeof(SECURITY_DESCRIPTOR))
4134
4135 #endif /* _SECURITY_DEFINED */
4136
4137 /*
4138 * SID_AND_ATTRIBUTES
4139 */
4140
4141 typedef struct _SID_AND_ATTRIBUTES {
4142     PSID Sid;
4143     DWORD Attributes;
4144 } SID_AND_ATTRIBUTES ;
4145
4146 /* security entities */
4147 #define SECURITY_NULL_RID                (0x00000000L)
4148 #define SECURITY_WORLD_RID                (0x00000000L)
4149 #define SECURITY_LOCAL_RID                (0x00000000L)
4150
4151 #define SECURITY_NULL_SID_AUTHORITY      {0,0,0,0,0,0}
4152
4153 /* S-1-1 */
4154 #define SECURITY_WORLD_SID_AUTHORITY      {0,0,0,0,0,1}
4155
4156 /* S-1-2 */
4157 #define SECURITY_LOCAL_SID_AUTHORITY      {0,0,0,0,0,2}
4158
4159 /* S-1-3 */
4160 #define SECURITY_CREATOR_SID_AUTHORITY    {0,0,0,0,0,3}
4161 #define SECURITY_CREATOR_OWNER_RID        (0x00000000L)
4162 #define SECURITY_CREATOR_GROUP_RID        (0x00000001L)
4163 #define SECURITY_CREATOR_OWNER_SERVER_RID (0x00000002L)
4164 #define SECURITY_CREATOR_GROUP_SERVER_RID (0x00000003L)
4165
4166 /* S-1-4 */
4167 #define SECURITY_NON_UNIQUE_AUTHORITY     {0,0,0,0,0,4}
4168
4169 /* S-1-5 */
4170 #define SECURITY_NT_AUTHORITY             {0,0,0,0,0,5}
4171 #define SECURITY_DIALUP_RID               0x00000001L
4172 #define SECURITY_NETWORK_RID              0x00000002L
4173 #define SECURITY_BATCH_RID                0x00000003L
4174 #define SECURITY_INTERACTIVE_RID          0x00000004L
4175 #define SECURITY_LOGON_IDS_RID            0x00000005L
4176 #define SECURITY_SERVICE_RID              0x00000006L
4177 #define SECURITY_ANONYMOUS_LOGON_RID      0x00000007L
4178 #define SECURITY_PROXY_RID                0x00000008L
4179 #define SECURITY_ENTERPRISE_CONTROLLERS_RID 0x00000009L

```

```

4180 #define SECURITY_PRINCIPAL_SELF_RID          0x0000000AL
4181 #define SECURITY_AUTHENTICATED_USER_RID      0x0000000BL
4182 #define SECURITY_RESTRICTED_CODE_RID        0x0000000CL
4183 #define SECURITY_TERMINAL_SERVER_RID        0x0000000DL
4184 #define SECURITY_LOCAL_SYSTEM_RID          0x00000012L
4185 #define SECURITY_NT_NON_UNIQUE              0x00000015L
4186 #define SECURITY_BUILTIN_DOMAIN_RID        0x00000020L
4187
4188 #define DOMAIN_GROUP_RID_ADMINS              0x00000200L
4189 #define DOMAIN_GROUP_RID_USERS              0x00000201L
4190 #define DOMAIN_GROUP_RID_GUESTS             0x00000202L
4191
4192 #define DOMAIN_ALIAS_RID_ADMINS              0x00000220L
4193 #define DOMAIN_ALIAS_RID_USERS              0x00000221L
4194 #define DOMAIN_ALIAS_RID_GUESTS             0x00000222L
4195
4196 #define SECURITY_SERVER_LOGON_RID            SECURITY_ENTERPRISE_CONTROLLERS_RID
4197
4198 #define SECURITY_LOGON_IDS_RID_COUNT         (3L)
4199
4200 /*
4201 * TOKEN_USER
4202 */
4203
4204 typedef struct _TOKEN_USER {
4205     SID_AND_ATTRIBUTES User;
4206 } TOKEN_USER;
4207
4208 /*
4209 * TOKEN_GROUPS
4210 */
4211
4212 typedef struct _TOKEN_GROUPS {
4213     DWORD GroupCount;
4214     SID_AND_ATTRIBUTES Groups[ANYSIZE_ARRAY];
4215 } TOKEN_GROUPS;
4216
4217 /*
4218 * LUID_AND_ATTRIBUTES
4219 */
4220
4221 typedef union _LARGE_INTEGER {
4222     struct dorp {
4223         DWORD LowPart;
4224         LONG HighPart;
4225     } DUMMYSTRUCTNAME;
4226     LONGLONG QuadPart;
4227 } LARGE_INTEGER, *PLARGE_INTEGER, *PLARGE_INTEGER;
4228
4229 typedef union _ULARGE_INTEGER {
4230     struct banana {
4231         DWORD LowPart;
4232         DWORD HighPart;
4233     } DUMMYSTRUCTNAME;
4234     ULONGLONG QuadPart;
4235 } ULARGE_INTEGER, *LPULARGE_INTEGER, *PULARGE_INTEGER;
4236
4237 /*
4238 * Locally Unique Identifier
4239 */
4240
4241 typedef struct _LUID {
4242     DWORD LowPart;
4243     LONG HighPart;
4244 } LUID, *PLUID;
4245
4246 #include "pshpack4.h"
4247 typedef struct _LUID_AND_ATTRIBUTES {
4248     LUID Luid;
4249     DWORD Attributes;
4250 } LUID_AND_ATTRIBUTES;
4251 #include "poppack.h"
4252
4253 /*
4254 * PRIVILEGE_SET
4255 */
4256
4257 typedef struct _PRIVILEGE_SET {
4258     DWORD PrivilegeCount;
4259     DWORD Control;
4260     LUID_AND_ATTRIBUTES Privilege[ANYSIZE_ARRAY];
4261 } PRIVILEGE_SET, *PPRIVILEGE_SET;
4262
4263 /*
4264 * TOKEN_PRIVILEGES
4265 */
4266

```

```

4267 typedef struct _TOKEN_PRIVILEGES {
4268     DWORD PrivilegeCount;
4269     LUID_AND_ATTRIBUTES Privileges[ANYSIZE_ARRAY];
4270 } TOKEN_PRIVILEGES, *PTOKEN_PRIVILEGES;
4271
4272 /*
4273 * TOKEN_OWNER
4274 */
4275
4276 typedef struct _TOKEN_OWNER {
4277     PSID Owner;
4278 } TOKEN_OWNER;
4279
4280 /*
4281 * TOKEN_PRIMARY_GROUP
4282 */
4283
4284 typedef struct _TOKEN_PRIMARY_GROUP {
4285     PSID PrimaryGroup;
4286 } TOKEN_PRIMARY_GROUP;
4287
4288
4289 /*
4290 * TOKEN_DEFAULT_DACL
4291 */
4292
4293 typedef struct _TOKEN_DEFAULT_DACL {
4294     PACL DefaultDacl;
4295 } TOKEN_DEFAULT_DACL;
4296
4297 /*
4298 * TOKEN_SOURCE
4299 */
4300
4301 typedef struct _TOKEN_SOURCE {
4302     char Sourcename[8];
4303     LUID SourceIdentifier;
4304 } TOKEN_SOURCE;
4305
4306 /*
4307 * TOKEN_TYPE
4308 */
4309
4310 typedef enum tagTOKEN_TYPE {
4311     TokenPrimary = 1,
4312     TokenImpersonation
4313 } TOKEN_TYPE;
4314
4315 /*
4316 * SECURITY_IMPERSONATION_LEVEL
4317 */
4318
4319 typedef enum _SECURITY_IMPERSONATION_LEVEL {
4320     SecurityAnonymous,
4321     SecurityIdentification,
4322     SecurityImpersonation,
4323     SecurityDelegation
4324 } SECURITY_IMPERSONATION_LEVEL, *PSECURITY_IMPERSONATION_LEVEL;
4325
4326
4327 typedef BOOLEAN SECURITY_CONTEXT_TRACKING_MODE,
4328     * PSECURITY_CONTEXT_TRACKING_MODE;
4329 /*
4330 * Quality of Service
4331 */
4332
4333 typedef struct _SECURITY_QUALITY_OF_SERVICE {
4334     DWORD Length;
4335     SECURITY_IMPERSONATION_LEVEL ImpersonationLevel;
4336     SECURITY_CONTEXT_TRACKING_MODE ContextTrackingMode;
4337     BOOLEAN EffectiveOnly;
4338 } SECURITY_QUALITY_OF_SERVICE, *PSECURITY_QUALITY_OF_SERVICE;
4339
4340 /*
4341 * TOKEN_STATISTICS
4342 */
4343
4344 typedef struct _TOKEN_STATISTICS {
4345     LUID TokenId;
4346     LUID AuthenticationId;
4347     LARGE_INTEGER ExpirationTime;
4348     TOKEN_TYPE TokenType;
4349     SECURITY_IMPERSONATION_LEVEL ImpersonationLevel;
4350     DWORD DynamicCharged;
4351     DWORD DynamicAvailable;
4352     DWORD GroupCount;
4353     DWORD PrivilegeCount;

```



```
4354     LUID ModifiedId;
4355 } TOKEN_STATISTICS;
4356
4357 /*
4358 * ACLs of NT
4359 */
4360
4361 #define ACL_REVISION    2
4362
4363 #define ACL_REVISION1   1
4364 #define ACL_REVISION2   2
4365
4366 /* ACEs, directly starting after an ACL */
4367 typedef struct _ACE_HEADER {
4368     BYTE    AceType;
4369     BYTE    AceFlags;
4370     WORD    AceSize;
4371 } ACE_HEADER, *PACE_HEADER;
4372
4373 /* AceType */
4374 #define ACCESS_ALLOWED_ACE_TYPE    0
4375 #define ACCESS_DENIED_ACE_TYPE    1
4376 #define SYSTEM_AUDIT_ACE_TYPE    2
4377 #define SYSTEM_ALARM_ACE_TYPE    3
4378
4379 /* inherit AceFlags */
4380 #define OBJECT_INHERIT_ACE    0x01
4381 #define CONTAINER_INHERIT_ACE    0x02
4382 #define NO_PROPAGATE_INHERIT_ACE    0x04
4383 #define INHERIT_ONLY_ACE    0x08
4384 #define VALID_INHERIT_FLAGS    0x0F
4385
4386 /* AceFlags mask for what events we (should) audit */
4387 #define SUCCESSFUL_ACCESS_ACE_FLAG    0x40
4388 #define FAILED_ACCESS_ACE_FLAG    0x80
4389
4390 /* different ACEs depending on AceType
4391 * SidStart marks the begin of a SID
4392 * so the thing finally looks like this:
4393 * 0: ACE_HEADER
4394 * 4: ACCESS_MASK
4395 * 8... : SID
4396 */
4397 typedef struct _ACCESS_ALLOWED_ACE {
4398     ACE_HEADER Header;
4399     DWORD    Mask;
4400     DWORD    SidStart;
4401 } ACCESS_ALLOWED_ACE, *PACCESS_ALLOWED_ACE;
4402
4403 typedef struct _ACCESS_DENIED_ACE {
4404     ACE_HEADER Header;
4405     DWORD    Mask;
4406     DWORD    SidStart;
4407 } ACCESS_DENIED_ACE, *PACCESS_DENIED_ACE;
4408
4409 typedef struct _SYSTEM_AUDIT_ACE {
4410     ACE_HEADER Header;
4411     DWORD    Mask;
4412     DWORD    SidStart;
4413 } SYSTEM_AUDIT_ACE, *PSYSTEM_AUDIT_ACE;
4414
4415 typedef struct _SYSTEM_ALARM_ACE {
4416     ACE_HEADER Header;
4417     DWORD    Mask;
4418     DWORD    SidStart;
4419 } SYSTEM_ALARM_ACE, *PSYSTEM_ALARM_ACE;
4420
4421 typedef enum tagSID_NAME_USE {
4422     SidTypeUser = 1,
4423     SidTypeGroup,
4424     SidTypeDomain,
4425     SidTypeAlias,
4426     SidTypeWellKnownGroup,
4427     SidTypeDeletedAccount,
4428     SidTypeInvalid,
4429     SidTypeUnknown
4430 } SID_NAME_USE, *PSID_NAME_USE;
4431
4432 /* Access rights */
4433
4434 /* DELETE may be already defined via /usr/include/arpa/nameser_compat.h */
4435 #undef DELETE
4436 #define DELETE    0x00010000
4437 #define READ_CONTROL    0x00020000
4438 #define WRITE_DAC    0x00040000
4439 #define WRITE_OWNER    0x00080000
4440 #define SYNCHRONIZE    0x00100000
```

```

4441 #define STANDARD_RIGHTS_REQUIRED    0x000f0000
4442
4443 #define STANDARD_RIGHTS_READ          READ_CONTROL
4444 #define STANDARD_RIGHTS_WRITE        READ_CONTROL
4445 #define STANDARD_RIGHTS_EXECUTE      READ_CONTROL
4446
4447 #define STANDARD_RIGHTS_ALL           0x001f0000
4448
4449 #define SPECIFIC_RIGHTS_ALL           0x0000ffff
4450
4451 #define GENERIC_READ                   0x80000000
4452 #define GENERIC_WRITE                  0x40000000
4453 #define GENERIC_EXECUTE                 0x20000000
4454 #define GENERIC_ALL                     0x10000000
4455
4456 #define MAXIMUM_ALLOWED                0x02000000
4457 #define ACCESS_SYSTEM_SECURITY         0x01000000
4458
4459 #define EVENT_MODIFY_STATE              0x0002
4460 #define EVENT_ALL_ACCESS                (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x3)
4461
4462 #define SEMAPHORE_MODIFY_STATE          0x0002
4463 #define SEMAPHORE_ALL_ACCESS            (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x3)
4464
4465 #define MUTEX_MODIFY_STATE              0x0001
4466 #define MUTEX_ALL_ACCESS                (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x1)
4467
4468 #define TIMER_QUERY_STATE               0x0001
4469 #define TIMER_MODIFY_STATE              0x0002
4470 #define TIMER_ALL_ACCESS                (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x3)
4471
4472 #define PROCESS_TERMINATE               0x0001
4473 #define PROCESS_CREATE_THREAD           0x0002
4474 #define PROCESS_VM_OPERATION            0x0008
4475 #define PROCESS_VM_READ                 0x0010
4476 #define PROCESS_VM_WRITE                0x0020
4477 #define PROCESS_DUP_HANDLE              0x0040
4478 #define PROCESS_CREATE_PROCESS          0x0080
4479 #define PROCESS_SET_QUOTA               0x0100
4480 #define PROCESS_SET_INFORMATION         0x0200
4481 #define PROCESS_QUERY_INFORMATION       0x0400
4482 #define PROCESS_ALL_ACCESS              (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0xffff)
4483
4484 #define THREAD_TERMINATE                0x0001
4485 #define THREAD_SUSPEND_RESUME           0x0002
4486 #define THREAD_GET_CONTEXT              0x0008
4487 #define THREAD_SET_CONTEXT              0x0010
4488 #define THREAD_SET_INFORMATION           0x0020
4489 #define THREAD_QUERY_INFORMATION        0x0040
4490 #define THREAD_SET_THREAD_TOKEN         0x0080
4491 #define THREAD_IMPERSONATE              0x0100
4492 #define THREAD_DIRECT_IMPERSONATION     0x0200
4493 #define THREAD_ALL_ACCESS                (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x3ff)
4494
4495 #define THREAD_BASE_PRIORITY_LOWRT      15
4496 #define THREAD_BASE_PRIORITY_MAX        2
4497 #define THREAD_BASE_PRIORITY_MIN        -2
4498 #define THREAD_BASE_PRIORITY_IDLE       -15
4499
4500 #define FILE_READ_DATA                   0x0001    /* file & pipe */
4501 #define FILE_LIST_DIRECTORY              0x0001    /* directory */
4502 #define FILE_WRITE_DATA                  0x0002    /* file & pipe */
4503 #define FILE_ADD_FILE                    0x0002    /* directory */
4504 #define FILE_APPEND_DATA                 0x0004    /* file */
4505 #define FILE_ADD_SUBDIRECTORY            0x0004    /* directory */
4506 #define FILE_CREATE_PIPE_INSTANCE        0x0004    /* named pipe */
4507 #define FILE_READ_EA                     0x0008    /* file & directory */
4508 #define FILE_READ_PROPERTIES              FILE_READ_EA
4509 #define FILE_WRITE_EA                     0x0010    /* file & directory */
4510 #define FILE_WRITE_PROPERTIES            FILE_WRITE_EA
4511 #define FILE_EXECUTE                     0x0020    /* file */
4512 #define FILE_TRAVERSE                    0x0020    /* directory */
4513 #define FILE_DELETE_CHILD                0x0040    /* directory */
4514 #define FILE_READ_ATTRIBUTES             0x0080    /* all */
4515 #define FILE_WRITE_ATTRIBUTES            0x0100    /* all */
4516 #define FILE_ALL_ACCESS                   (STANDARD_RIGHTS_REQUIRED| SYNCHRONIZE| 0x1ff)
4517
4518 #define FILE_GENERIC_READ                 (STANDARD_RIGHTS_READ | FILE_READ_DATA | \
4519 FILE_READ_ATTRIBUTES | FILE_READ_EA | \
4520 SYNCHRONIZE)
4521 #define FILE_GENERIC_WRITE                (STANDARD_RIGHTS_WRITE | FILE_WRITE_DATA | \
4522 FILE_WRITE_ATTRIBUTES | FILE_WRITE_EA | \
4523 FILE_APPEND_DATA | SYNCHRONIZE)
4524 #define FILE_GENERIC_EXECUTE              (STANDARD_RIGHTS_EXECUTE | FILE_EXECUTE | \
4525 FILE_READ_ATTRIBUTES | SYNCHRONIZE)
4526
4527

```

```
4528 /* File attribute flags */
4529 #define FILE_SHARE_READ 0x00000001L
4530 #define FILE_SHARE_WRITE 0x00000002L
4531 #define FILE_SHARE_DELETE 0x00000004L
4532 #define FILE_ATTRIBUTE_READONLY 0x00000001L
4533 #define FILE_ATTRIBUTE_HIDDEN 0x00000002L
4534 #define FILE_ATTRIBUTE_SYSTEM 0x00000004L
4535 #define FILE_ATTRIBUTE_LABEL 0x00000008L /* Not in Windows API */
4536 #define FILE_ATTRIBUTE_DIRECTORY 0x00000010L
4537 #define FILE_ATTRIBUTE_ARCHIVE 0x00000020L
4538 #define FILE_ATTRIBUTE_NORMAL 0x00000080L
4539 #define FILE_ATTRIBUTE_TEMPORARY 0x00000100L
4540 #define FILE_ATTRIBUTE_ATOMIC_WRITE 0x00000200L
4541 #define FILE_ATTRIBUTE_XACTION_WRITE 0x00000400L
4542 #define FILE_ATTRIBUTE_COMPRESSED 0x00000800L
4543 #define FILE_ATTRIBUTE_OFFLINE 0x00001000L
4544 #define FILE_ATTRIBUTE_SYMLINK 0x80000000L /* Not in Windows API */
4545
4546 /* File notification flags */
4547 #define FILE_NOTIFY_CHANGE_FILE_NAME 0x00000001
4548 #define FILE_NOTIFY_CHANGE_DIR_NAME 0x00000002
4549 #define FILE_NOTIFY_CHANGE_ATTRIBUTES 0x00000004
4550 #define FILE_NOTIFY_CHANGE_SIZE 0x00000008
4551 #define FILE_NOTIFY_CHANGE_LAST_WRITE 0x00000010
4552 #define FILE_NOTIFY_CHANGE_LAST_ACCESS 0x00000020
4553 #define FILE_NOTIFY_CHANGE_CREATION 0x00000040
4554 #define FILE_NOTIFY_CHANGE_SECURITY 0x00000100
4555
4556 #define FILE_ACTION_ADDED 0x00000001
4557 #define FILE_ACTION_REMOVED 0x00000002
4558 #define FILE_ACTION_MODIFIED 0x00000003
4559 #define FILE_ACTION_RENAMED_OLD_NAME 0x00000004
4560 #define FILE_ACTION_RENAMED_NEW_NAME 0x00000005
4561
4562
4563 #define FILE_CASE_SENSITIVE_SEARCH 0x00000001
4564 #define FILE_CASE_PRESERVED_NAMES 0x00000002
4565 #define FILE_UNICODE_ON_DISK 0x00000004
4566 #define FILE_PERSISTENT_ACLS 0x00000008
4567 #define FILE_FILE_COMPRESSION 0x00000010
4568 #define FILE_VOLUME_IS_COMPRESSED 0x00008000
4569
4570 /* File alignments (NT) */
4571 #define FILE_BYTE_ALIGNMENT 0x00000000
4572 #define FILE_WORD_ALIGNMENT 0x00000001
4573 #define FILE_LONG_ALIGNMENT 0x00000003
4574 #define FILE_QUAD_ALIGNMENT 0x00000007
4575 #define FILE_OCTA_ALIGNMENT 0x0000000f
4576 #define FILE_32_BYTE_ALIGNMENT 0x0000001f
4577 #define FILE_64_BYTE_ALIGNMENT 0x0000003f
4578 #define FILE_128_BYTE_ALIGNMENT 0x0000007f
4579 #define FILE_256_BYTE_ALIGNMENT 0x000000ff
4580 #define FILE_512_BYTE_ALIGNMENT 0x000001ff
4581
4582 #define REG_NONE 0 /* no type */
4583 #define REG_SZ 1 /* string type (ASCII) */
4584 #define REG_EXPAND_SZ 2 /* string, includes %ENVVAR% (expanded by caller) (ASCII) */
4585 #define REG_BINARY 3 /* binary format, callerspecific */
4586 /* YES, REG_DWORD == REG_DWORD_LITTLE_ENDIAN */
4587 #define REG_DWORD 4 /* DWORD in little endian format */
4588 #define REG_DWORD_LITTLE_ENDIAN 4 /* DWORD in little endian format */
4589 #define REG_DWORD_BIG_ENDIAN 5 /* DWORD in big endian format */
4590 #define REG_LINK 6 /* symbolic link (UNICODE) */
4591 #define REG_MULTI_SZ 7 /* multiple strings, delimited by \0, terminated by \0\0 (ASCII) */
4592 #define REG_RESOURCE_LIST 8 /* resource list? huh? */
4593 #define REG_FULL_RESOURCE_DESCRIPTOR 9 /* full resource descriptor? huh? */
4594 #define REG_RESOURCE_REQUIREMENTS_LIST 10
4595
4596 /* ----- begin registry ----- */
4597
4598 /* Registry security values */
4599 #define OWNER_SECURITY_INFORMATION 0x00000001
4600 #define GROUP_SECURITY_INFORMATION 0x00000002
4601 #define DACL_SECURITY_INFORMATION 0x00000004
4602 #define SACL_SECURITY_INFORMATION 0x00000008
4603
4604 #define REG_OPTION_RESERVED 0x00000000
4605 #define REG_OPTION_NON_VOLATILE 0x00000000
4606 #define REG_OPTION_VOLATILE 0x00000001
4607 #define REG_OPTION_CREATE_LINK 0x00000002
4608 #define REG_OPTION_BACKUP_RESTORE 0x00000004 /* FIXME */
4609 #define REG_OPTION_OPEN_LINK 0x00000008
4610 #define REG_LEGAL_OPTION (REG_OPTION_RESERVED| \
4611 REG_OPTION_NON_VOLATILE| \
4612 REG_OPTION_VOLATILE| \
4613 REG_OPTION_CREATE_LINK| \
4614 REG_OPTION_BACKUP_RESTORE| \
```

```

4615 REG_OPTION_OPEN_LINK)
4616
4617
4618 #define REG_CREATED_NEW_KEY 0x00000001
4619 #define REG_OPENED_EXISTING_KEY 0x00000002
4620
4621 /* For RegNotifyChangeKeyValue */
4622 #define REG_NOTIFY_CHANGE_NAME 0x1
4623
4624 #define KEY_QUERY_VALUE 0x00000001
4625 #define KEY_SET_VALUE 0x00000002
4626 #define KEY_CREATE_SUB_KEY 0x00000004
4627 #define KEY_ENUMERATE_SUB_KEYS 0x00000008
4628 #define KEY_NOTIFY 0x00000010
4629 #define KEY_CREATE_LINK 0x00000020
4630
4631 #define KEY_READ ((STANDARD_RIGHTS_READ| \
4632 KEY_QUERY_VALUE| \
4633 KEY_ENUMERATE_SUB_KEYS| \
4634 KEY_NOTIFY) \
4635 & (~SYNCHRONIZE) \
4636 )
4637 #define KEY_WRITE ((STANDARD_RIGHTS_WRITE| \
4638 KEY_SET_VALUE| \
4639 KEY_CREATE_SUB_KEY) \
4640 & (~SYNCHRONIZE) \
4641 )
4642 #define KEY_EXECUTE ((KEY_READ) \
4643 & (~SYNCHRONIZE)) \
4644 )
4645 #define KEY_ALL_ACCESS ((STANDARD_RIGHTS_ALL| \
4646 KEY_QUERY_VALUE| \
4647 KEY_SET_VALUE| \
4648 KEY_CREATE_SUB_KEY| \
4649 KEY_ENUMERATE_SUB_KEYS| \
4650 KEY_NOTIFY| \
4651 KEY_CREATE_LINK) \
4652 & (~SYNCHRONIZE) \
4653 )
4654 /* ----- end registry ----- */
4655
4656
4657 #define EVENTLOG_SUCCESS 0x0000
4658 #define EVENTLOG_ERROR_TYPE 0x0001
4659 #define EVENTLOG_WARNING_TYPE 0x0002
4660 #define EVENTLOG_INFORMATION_TYPE 0x0004
4661 #define EVENTLOG_AUDIT_SUCCESS 0x0008
4662 #define EVENTLOG_AUDIT_FAILURE 0x0010
4663
4664 #define SERVICE_BOOT_START 0x00000000
4665 #define SERVICE_SYSTEM_START 0x00000001
4666 #define SERVICE_AUTO_START 0x00000002
4667 #define SERVICE_DEMAND_START 0x00000003
4668 #define SERVICE_DISABLED 0x00000004
4669
4670 #define SERVICE_ERROR_IGNORE 0x00000000
4671 #define SERVICE_ERROR_NORMAL 0x00000001
4672 #define SERVICE_ERROR_SEVERE 0x00000002
4673 #define SERVICE_ERROR_CRITICAL 0x00000003
4674
4675 /* Service types */
4676 #define SERVICE_KERNEL_DRIVER 0x00000001
4677 #define SERVICE_FILE_SYSTEM_DRIVER 0x00000002
4678 #define SERVICE_ADAPTER 0x00000004
4679 #define SERVICE_RECOGNIZER_DRIVER 0x00000008
4680
4681 #define SERVICE_DRIVER ( SERVICE_KERNEL_DRIVER | SERVICE_FILE_SYSTEM_DRIVER | \
4682 SERVICE_RECOGNIZER_DRIVER )
4683
4684 #define SERVICE_WIN32_OWN_PROCESS 0x00000010
4685 #define SERVICE_WIN32_SHARE_PROCESS 0x00000020
4686 #define SERVICE_WIN32 (SERVICE_WIN32_OWN_PROCESS | SERVICE_WIN32_SHARE_PROCESS)
4687
4688 #define SERVICE_INTERACTIVE_PROCESS 0x00000100
4689
4690 #define SERVICE_TYPE_ALL ( SERVICE_WIN32 | SERVICE_ADAPTER | \
4691 SERVICE_DRIVER | SERVICE_INTERACTIVE_PROCESS )
4692
4693
4694 typedef enum _CM_SERVICE_NODE_TYPE
4695 {
4696     DriverType = SERVICE_KERNEL_DRIVER,
4697     FileSystemType = SERVICE_FILE_SYSTEM_DRIVER,
4698     Win32ServiceOwnProcess = SERVICE_WIN32_OWN_PROCESS,
4699     Win32ServiceShareProcess = SERVICE_WIN32_SHARE_PROCESS,
4700     AdapterType = SERVICE_ADAPTER,
4701     RecognizerType = SERVICE_RECOGNIZER_DRIVER

```

```

4702 } SERVICE_NODE_TYPE;
4703
4704 typedef enum _CM_SERVICE_LOAD_TYPE
4705 {
4706     BootLoad      = SERVICE_BOOT_START,
4707     SystemLoad    = SERVICE_SYSTEM_START,
4708     AutoLoad      = SERVICE_AUTO_START,
4709     DemandLoad    = SERVICE_DEMAND_START,
4710     DisableLoad   = SERVICE_DISABLED
4711 } SERVICE_LOAD_TYPE;
4712
4713 typedef enum _CM_ERROR_CONTROL_TYPE
4714 {
4715     IgnoreError    = SERVICE_ERROR_IGNORE,
4716     NormalError    = SERVICE_ERROR_NORMAL,
4717     SevereError    = SERVICE_ERROR_SEVERE,
4718     CriticalError  = SERVICE_ERROR_CRITICAL
4719 } SERVICE_ERROR_TYPE;
4720
4721
4722
4723 #define RtlEqualMemory(Destination, Source, Length) (!memcmp((Destination), (Source), (Length)))
4724 #define RtlMoveMemory(Destination, Source, Length) memmove((Destination), (Source), (Length))
4725 #define RtlCopyMemory(Destination, Source, Length) memcpy((Destination), (Source), (Length))
4726 #define RtlFillMemory(Destination, Length, Fill) memset((Destination), (Fill), (Length))
4727 #define RtlZeroMemory(Destination, Length) memset((Destination), 0, (Length))
4728
4729 #include "guiddef.h"
4730
4731 typedef struct _RTL_CRITICAL_SECTION_DEBUG
4732 {
4733     WORD    Type;
4734     WORD    CreatorBackTraceIndex;
4735     struct _RTL_CRITICAL_SECTION *CriticalSection;
4736     LIST_ENTRY ProcessLocksList;
4737     DWORD    EntryCount;
4738     DWORD    ContentionCount;
4739     DWORD    Spare[ 2 ];
4740 } RTL_CRITICAL_SECTION_DEBUG, *PRTL_CRITICAL_SECTION_DEBUG, RTL_RESOURCE_DEBUG, *PRTL_RESOURCE_DEBUG;
4741
4742 typedef struct _RTL_CRITICAL_SECTION {
4743     PRTL_CRITICAL_SECTION_DEBUG DebugInfo;
4744     LONG    LockCount;
4745     LONG    RecursionCount;
4746     HANDLE  OwningThread;
4747     HANDLE  LockSemaphore;
4748     ULONG_PTR SpinCount;
4749 } RTL_CRITICAL_SECTION, *PRTL_CRITICAL_SECTION;
4750
4751 #endif /* __WINE_WINNT_H */

```

5.13 winuser.h

```

1  #ifndef _WINUSER_
2  #define _WINUSER_
3
4  #ifndef RC_INVOKED
5  #include <stdarg.h>
6  #endif
7
8  #ifdef __cplusplus
9  extern "C" {
10 #endif
11
12 /* Define a bunch of callback types */
13
14 #if defined(STRICT) || defined(__WINE__)
15 typedef BOOL    CALLBACK (*DLGPROC) (HWND, UINT, WPARAM, LPARAM);
16 typedef BOOL    CALLBACK (*DRAWSTATEPROC) (HDC, LPARAM, WPARAM, int, int);
17 typedef INT     CALLBACK (*EDITWORDBREAKPROCA) (LPSTR, INT, INT, INT);
18 typedef INT     CALLBACK (*EDITWORDBREAKPROCW) (LPWSTR, INT, INT, INT);
19 typedef BOOL    CALLBACK (*GRAYSTRINGPROC) (HDC, LPARAM, INT);
20 typedef LRESULT CALLBACK (*HOOKPROC) (INT, WPARAM, LPARAM);
21 typedef BOOL    CALLBACK (*NAMEENUMPROCA) (LPSTR, LPARAM);
22 typedef BOOL    CALLBACK (*NAMEENUMPROCW) (LPWSTR, LPARAM);
23 typedef BOOL    CALLBACK (*PROPENUMPROCA) (HWND, LPCSTR, HANDLE);
24 typedef BOOL    CALLBACK (*PROPENUMPROCW) (HWND, LPCWSTR, HANDLE);
25 typedef BOOL    CALLBACK (*PROPENUMPROCEXA) (HWND, LPCSTR, HANDLE, ULONG_PTR);
26 typedef BOOL    CALLBACK (*PROPENUMPROCEXW) (HWND, LPCWSTR, HANDLE, ULONG_PTR);
27 typedef VOID    CALLBACK (*SENDASYNCPROC) (HWND, UINT, ULONG_PTR, LRESULT);
28 typedef VOID    CALLBACK (*TIMERPROC) (HWND, UINT, UINT, DWORD);
29 typedef BOOL    CALLBACK (*WNDENUMPROC) (HWND, LPARAM);
30 #else
31 typedef FARPROC DLGPROC;

```

```

32 typedef FARPROC DRAWSTATEPROC;
33 typedef FARPROC EDITWORDBREAKPROCA;
34 typedef FARPROC EDITWORDBREAKPROCW;
35 typedef FARPROC GRAYSTRINGPROC;
36 typedef FARPROC HOOKPROC;
37 typedef FARPROC NAMEENUMPROCA;
38 typedef FARPROC NAMEENUMPROCW;
39 typedef FARPROC PROPENUMPROCA;
40 typedef FARPROC PROPENUMPROCW;
41 typedef FARPROC PROPENUMPROCEXA;
42 typedef FARPROC PROPENUMPROCEXW;
43 typedef FARPROC SENDASYNCPROC;
44 typedef FARPROC TIMERPROC;
45 typedef FARPROC WNDENUMPROC;
46 #endif /* STRICT || __WINE__ */
47
48 typedef NAMEENUMPROCA WINSTAENUMPROCA;
49 typedef NAMEENUMPROCA DESKTOPENUMPROCA;
50 typedef NAMEENUMPROCW WINSTAENUMPROCW;
51 typedef NAMEENUMPROCW DESKTOPENUMPROCW;
52
53 typedef LRESULT CALLBACK (*WNDPROC) (HWND, UINT, WPARAM, LPARAM);
54
55 DECL_WINELIB_TYPE_AW(DESKTOPENUMPROC)
56 DECL_WINELIB_TYPE_AW(EDITWORDBREAKPROC)
57 DECL_WINELIB_TYPE_AW(NAMEENUMPROC)
58 DECL_WINELIB_TYPE_AW(PROPENUMPROC)
59 DECL_WINELIB_TYPE_AW(PROPENUMPROCEX)
60 DECL_WINELIB_TYPE_AW(WINSTAENUMPROC)
61
62
63 typedef HANDLE HDWP;
64
65 /* flags for FILTERKEYS dwFlags field */
66 #define FKF_AVAILABLE 0x00000002
67 #define FKF_CLICKON 0x00000040
68 #define FKF_FILTERKEYSON 0x00000001
69 #define FKF_HOTKEYACTIVE 0x00000004
70 #define FKF_HOTKEYSOUND 0x00000010
71 #define FKF_CONFIRMHOTKEY 0x00000008
72 #define FKF_INDICATOR 0x00000020
73
74 typedef struct tagFILTERKEYS
75 {
76     UINT    cbSize;
77     DWORD   dwFlags;
78     DWORD   iWaitMSec;
79     DWORD   iDelayMSec;
80     DWORD   iRepeatMSec;
81     DWORD   iBounceMSec;
82 } FILTERKEYS, *PFILTERKEYS, *LPFILTERKEYS;
83
84 /* flags for TOGGLEKEYS dwFlags field */
85 #define TKF_AVAILABLE 0x00000002
86 #define TKF_CONFIRMHOTKEY 0x00000008
87 #define TKF_HOTKEYACTIVE 0x00000004
88 #define TKF_HOTKEYSOUND 0x00000010
89 #define TKF_TOGGLEKEYSON 0x00000001
90
91 typedef struct tagTOGGLEKEYS
92 {
93     DWORD   cbSize;
94     DWORD   dwFlags;
95 } TOGGLEKEYS, *PTOGGLEKEYS, *LPTOGGLEKEYS;
96
97 /* flags for MOUSEKEYS dwFlags field */
98 #define MKF_AVAILABLE 0x00000002
99 #define MKF_CONFIRMHOTKEY 0x00000008
100 #define MKF_HOTKEYACTIVE 0x00000004
101 #define MKF_HOTKEYSOUND 0x00000010
102 #define MKF_INDICATOR 0x00000020
103 #define MKF_MOUSEKEYSON 0x00000001
104 #define MKF_MODIFIERS 0x00000040
105 #define MKF_REPLACENUMBERS 0x00000080
106
107 typedef struct tagMOUSEKEYS
108 {
109     UINT    cbSize;
110     DWORD   dwFlags;
111     DWORD   iMaxSpeed;
112     DWORD   iTimeToMaxSpeed;
113     DWORD   iCtrlSpeed;
114     DWORD   dwReserved1;
115     DWORD   dwReserved2;
116 } MOUSEKEYS, *PMOUSEKEYS, *LPMOUSEKEYS;
117
118 /* flags for STICKYKEYS dwFlags field */

```

```

119 #define SKF_AUDIBLEFEEDBACK 0x00000040
120 #define SKF_AVAILABLE 0x00000002
121 #define SKF_CONFIRMHOTKEY 0x00000008
122 #define SKF_HOTKEYACTIVE 0x00000004
123 #define SKF_HOTKEYSOUND 0x00000010
124 #define SKF_INDICATOR 0x00000020
125 #define SKF_STICKYKEYSON 0x00000001
126 #define SKF_TRISTATE 0x00000080
127 #define SKF_TWOKEYSOFF 0x00000100
128
129 typedef struct tagSTICKYKEYS
130 {
131     DWORD cbSize;
132     DWORD dwFlags;
133 } STICKYKEYS, *PSTICKYKEYS, *LPSTICKYKEYS;
134
135 /* flags for ACCESTIMEOUT dwFlags field */
136 #define ATF_ONOFFFEEDBACK 0x00000002
137 #define ATF_AVAILABLE 0x00000004
138 #define ATF_TIMEOUTTON 0x00000001
139
140 typedef struct tagACCESTIMEOUT
141 {
142     UINT cbSize;
143     DWORD dwFlags;
144     DWORD iTimeOutMSec;
145 } ACCESTIMEOUT, *PACCESTIMEOUT, *LPACCESTIMEOUT;
146
147 /* flags for SERIALKEYS dwFlags field */
148 #define SERKF_ACTIVE 0x00000008
149 #define SERKF_AVAILABLE 0x00000002
150 #define SERKF_INDICATOR 0x00000004
151 #define SERKF_SERIALKEYSON 0x00000001
152
153 typedef struct tagSERIALKEYSA
154 {
155     UINT cbSize;
156     DWORD dwFlags;
157     LPSTR lpszActivePort;
158     LPSTR lpszPort;
159     UINT iBaudRate;
160     UINT iPortState;
161     UINT iActive;
162 } SERIALKEYSA, *LPSERIALKEYSA;
163
164 typedef struct tagSERIALKEYSW {
165     UINT cbSize;
166     DWORD dwFlags;
167     LPWSTR lpszActivePort;
168     LPWSTR lpszPort;
169     UINT iBaudRate;
170     UINT iPortState;
171     UINT iActive;
172 } SERIALKEYSW, *LPSERIALKEYSW;
173
174 DECL_WINELIB_TYPE_AW(SERIALKEYS)
175 DECL_WINELIB_TYPE_AW(LPSERIALKEYS)
176
177 /* flags for SOUNDSENTRY dwFlags field */
178 #define SSF_AVAILABLE 0x00000002
179 #define SSF_SOUNDSENTRYON 0x00000001
180
181 #define SSTF_BORDER 0x00000002
182 #define SSTF_CHARS 0x00000001
183 #define SSTF_DISPLAY 0x00000003
184 #define SSTF_NONE 0x00000000
185
186 #define SSGF_DISPLAY 0x00000003
187 #define SSGF_NONE 0x00000000
188
189 #define SSWF_DISPLAY 0x00000003
190 #define SSWF_NONE 0x00000000
191 #define SSWF_TITLE 0x00000001
192 #define SSWF_WINDOW 0x00000002
193
194 typedef struct tagSOUNDSENTRYA
195 {
196     UINT cbSize;
197     DWORD dwFlags;
198     DWORD iFSTextEffect;
199     DWORD iFSTextEffectMSec;
200     DWORD iFSTextEffectColorBits;
201     DWORD iFSGrafEffect;
202     DWORD iFSGrafEffectMSec;
203     DWORD iFSGrafEffectColor;
204     DWORD iWindowsEffect;
205     DWORD iWindowsEffectMSec;

```

```

206     LPSTR    lpszWindowsEffectDLL;
207     DWORD    iWindowsEffectOrdinal;
208 } SOUNDSENTRYA, *LPSOUNDSENTRYA;
209
210 typedef struct tagSOUNDSENTRYW
211 {
212     UINT      cbSize;
213     DWORD     dwFlags;
214     DWORD     iFSTextEffect;
215     DWORD     iFSTextEffectMSec;
216     DWORD     iFSTextEffectColorBits;
217     DWORD     iFSGrafEffect;
218     DWORD     iFSGrafEffectMSec;
219     DWORD     iFSGrafEffectColor;
220     DWORD     iWindowsEffect;
221     DWORD     iWindowsEffectMSec;
222     LPWSTR    lpszWindowsEffectDLL;
223     DWORD     iWindowsEffectOrdinal;
224 } SOUNDSENTRYW, *LPSOUNDSENTRYW;
225
226 DECL_WINELIB_TYPE_AW(SOUNDSENTRY)
227 DECL_WINELIB_TYPE_AW(LPSOUNDSENTRY)
228
229 /* flags for HIGHCONTRAST dwFlags field */
230 #define HCF_HIGHCONTRASTON 0x00000001
231 #define HCF_AVAILABLE     0x00000002
232 #define HCF_HOTKEYACTIVE  0x00000004
233 #define HCF_CONFIRMHOTKEY 0x00000008
234 #define HCF_HOTKEYSOUND   0x00000010
235 #define HCF_INDICATOR     0x00000020
236 #define HCF_HOTKEYAVAILABLE 0x00000040
237
238 typedef struct tagHIGHCONTRASTA
239 {
240     UINT      cbSize;
241     DWORD     dwFlags;
242     LPSTR     lpszDefaultScheme;
243 } HIGHCONTRASTA, *LPHIGHCONTRASTA;
244
245 typedef struct tagHIGHCONTRASTW
246 {
247     UINT      cbSize;
248     DWORD     dwFlags;
249     LPWSTR    lpszDefaultScheme;
250 } HIGHCONTRASTW, *LPHIGHCONTRASTW;
251
252 DECL_WINELIB_TYPE_AW(HIGHCONTRAST)
253 DECL_WINELIB_TYPE_AW(LPHIGHCONTRAST)
254
255 typedef struct
256 {
257     UINT      message;
258     UINT      paramL;
259     UINT      paramH;
260     DWORD     time;
261     HWND      hwnd;
262 } EVENTMSG, *PEVENTMSG, *LPEVENTMSG;
263
264 /* WH_KEYBOARD_LL structure */
265 typedef struct tagKBDLLHOOKSTRUCT
266 {
267     DWORD     vkCode;
268     DWORD     scanCode;
269     DWORD     flags;
270     DWORD     time;
271     ULONG_PTR dwExtraInfo;
272 } KBDLLHOOKSTRUCT, *LPKBDLLHOOKSTRUCT, *PKBDLLHOOKSTRUCT;
273
274 #define LLKHF_EXTENDED    (KF_EXTENDED » 8)
275 #define LLKHF_INJECTED    0x00000010
276 #define LLKHF_ALTDOWN    (KF_ALTDOWN » 8)
277 #define LLKHF_UP          (KF_UP » 8)
278
279 /* WH_MOUSE_LL structure */
280 typedef struct tagMSLLHOOKSTRUCT
281 {
282     POINT      pt;
283     DWORD     mouseData;
284     DWORD     flags;
285     DWORD     time;
286     ULONG_PTR dwExtraInfo;
287 } MSLLHOOKSTRUCT, *LPMSLLHOOKSTRUCT, *PMSLLHOOKSTRUCT;
288
289 #define LLMHF_INJECTED    0x00000001
290
291 /* Mouse hook structure */
292

```



```

293 typedef struct
294 {
295     POINT pt;
296     HWND hwnd;
297     UINT wHitTestCode;
298     DWORD dwExtraInfo;
299 } MOUSEHOOKSTRUCT, *PMOUSEHOOKSTRUCT, *LPMOUSEHOOKSTRUCT;
300
301
302 /* Hardware hook structure */
303
304 typedef struct
305 {
306     HWND hwnd;
307     UINT wMessage;
308     WPARAM wParam;
309     LPARAM lParam;
310 } HARDWAREHOOKSTRUCT, *PHARDWAREHOOKSTRUCT, *LPHARDWAREHOOKSTRUCT;
311
312
313 /* Debug hook structure */
314
315 typedef struct
316 {
317     DWORD idThread;
318     DWORD idThreadInstaller;
319     LPARAM lParam;
320     WPARAM wParam;
321     INT code;
322 } DEBUGHOOKINFO, *PDEBUGHOOKINFO, *LPDEBUGHOOKINFO;
323
324 #define HKL_PREV 0
325 #define HKL_NEXT 1
326
327 #define KLF_ACTIVATE 0x00000001
328 #define KLF_SUBSTITUTE_OK 0x00000002
329 #define KLF_UNLOADPREVIOUS 0x00000004
330 #define KLF_REORDER 0x00000008
331 #define KLF_REPLACELANG 0x00000010
332 #define KLF_NOTELLSHELL 0x00000080
333
334 #define KL_NAMELENGTH 9
335
336 typedef struct tagMOUSEINPUT
337 {
338     LONG dx;
339     LONG dy;
340     DWORD mouseData;
341     DWORD dwFlags;
342     DWORD time;
343     ULONG_PTR dwExtraInfo;
344 } MOUSEINPUT, *PMOUSEINPUT, *LPMOUSEINPUT;
345
346 typedef struct tagKEYBDINPUT
347 {
348     WORD wVk;
349     WORD wScan;
350     DWORD dwFlags;
351     DWORD time;
352     ULONG_PTR dwExtraInfo;
353 } KEYBDINPUT, *PKEYBDINPUT, *LPKEYBDINPUT;
354
355 typedef struct tagHARDWAREINPUT
356 {
357     DWORD uMsg;
358     WORD wParamL;
359     WORD wParamH;
360 } HARDWAREINPUT, *PHARDWAREINPUT, *LPHARDWAREINPUT;
361
362 #define INPUT_MOUSE 0
363 #define INPUT_KEYBOARD 1
364 #define INPUT_HARDWARE 2
365
366 typedef struct tagINPUT
367 {
368     DWORD type;
369     union
370     {
371         MOUSEINPUT mi;
372         KEYBDINPUT ki;
373         HARDWAREINPUT hi;
374     } DUMMYUNIONNAME;
375 } INPUT, *PINPUT, *LPINPUT;
376
377
378 /***** Dialogs *****/
379

```

```

380 /* Gcc on Solaris has a version of this that we don't care about */
381 #undef FSHIFT
382
383 #define FVIRTKEY    TRUE           /* Assumed to be == TRUE */
384 #define FNOINVERT  0x02
385 #define FSHIFT     0x04
386 #define FCONTROL   0x08
387 #define FALT       0x10
388
389
390 typedef struct tagANIMATIONINFO
391 {
392     UINT        cbSize;
393     INT         iMinAnimate;
394 } ANIMATIONINFO, *LPANIMATIONINFO;
395
396 typedef struct tagNMHDR
397 {
398     HWND        hwndFrom;
399     UINT        idFrom;
400     UINT        code;
401 } NMHDR, *LPNMHDR;
402
403 typedef struct
404 {
405     UINT        cbSize;
406     INT         iTabLength;
407     INT         iLeftMargin;
408     INT         iRightMargin;
409     UINT        uiLengthDrawn;
410 } DRAWTEXT_PARAMS, *LPDRAWTEXT_PARAMS;
411
412 #define WM_USER      0x0400
413
414 #define DT_EDITCONTROL      0x00002000
415 #define DT_PATH_ELLIPSIS   0x00004000
416 #define DT_END_ELLIPSIS    0x00008000
417 #define DT_MODIFYSTRING    0x00010000
418 #define DT_RTLREADING      0x00020000
419 #define DT_WORD_ELLIPSIS   0x00040000
420
421 typedef struct
422 {
423     LPARAM        lParam;
424     WPARAM        wParam;
425     UINT          message;
426     HWND          hwnd;
427 } CWPSTRUCT, *PCWPSTRUCT, *LPCWPSTRUCT;
428
429 typedef struct
430 {
431     LRESULT        lResult;
432     LPARAM        lParam;
433     WPARAM        wParam;
434     DWORD          message;
435     HWND          hwnd;
436 } CWPRETSTRUCT, *PCWPRETSTRUCT, *LPCWPRETSTRUCT;
437
438 typedef struct
439 {
440     UINT          length;
441     UINT          flags;
442     UINT          showCmd;
443     POINT         ptMinPosition WINE_PACKED;
444     POINT         ptMaxPosition WINE_PACKED;
445     RECT          rcNormalPosition WINE_PACKED;
446 } WINDOWPLACEMENT, *PWINDOWPLACEMENT, *LPWINDOWPLACEMENT;
447
448
449 /* WINDOWPLACEMENT flags */
450 #define WPF_SETMINPOSITION      0x0001
451 #define WPF_RESTORETOMAXIMIZED 0x0002
452
453 /***** Dialogs *****/
454
455 #define MAKEINTRESOURCEA(i) (LPSTR)((DWORD)((WORD)(i)))
456 #define MAKEINTRESOURCEW(i) (LPWSTR)((DWORD)((WORD)(i)))
457 #define MAKEINTRESOURCE WINELIB_NAME_AW(MAKEINTRESOURCE)
458
459 /* Predefined resource types */
460 #define RT_CURSORA      MAKEINTRESOURCEA(1)
461 #define RT_CURSORW      MAKEINTRESOURCEW(1)
462 #define RT_CURSOR       WINELIB_NAME_AW(RT_CURSOR)
463 #define RT_BITMAPA      MAKEINTRESOURCEA(2)
464 #define RT_BITMAPW      MAKEINTRESOURCEW(2)
465 #define RT_BITMAP       WINELIB_NAME_AW(RT_BITMAP)
466 #define RT_ICONA        MAKEINTRESOURCEA(3)

```

```

467 #define RT_ICONW MAKEINTRESOURCEW(3)
468 #define RT_ICON WINELIB_NAME_AW(RT_ICON)
469 #define RT_MENUA MAKEINTRESOURCEA(4)
470 #define RT_MENUW MAKEINTRESOURCEW(4)
471 #define RT_MENU WINELIB_NAME_AW(RT_MENU)
472 #define RT_DIALOGA MAKEINTRESOURCEA(5)
473 #define RT_DIALOGW MAKEINTRESOURCEW(5)
474 #define RT_DIALOG WINELIB_NAME_AW(RT_DIALOG)
475 #define RT_STRINGA MAKEINTRESOURCEA(6)
476 #define RT_STRINGW MAKEINTRESOURCEW(6)
477 #define RT_STRING WINELIB_NAME_AW(RT_STRING)
478 #define RT_FONTDIRA MAKEINTRESOURCEA(7)
479 #define RT_FONTDIRW MAKEINTRESOURCEW(7)
480 #define RT_FONTDIR WINELIB_NAME_AW(RT_FONTDIR)
481 #define RT_FONTA MAKEINTRESOURCEA(8)
482 #define RT_FONTW MAKEINTRESOURCEW(8)
483 #define RT_FONT WINELIB_NAME_AW(RT_FONT)
484 #define RT_ACCELERATORA MAKEINTRESOURCEA(9)
485 #define RT_ACCELERATORW MAKEINTRESOURCEW(9)
486 #define RT_ACCELERATOR WINELIB_NAME_AW(RT_ACCELERATOR)
487 #define RT_RCDATAA MAKEINTRESOURCEA(10)
488 #define RT_RCDATAW MAKEINTRESOURCEW(10)
489 #define RT_RCDATA WINELIB_NAME_AW(RT_RCDATA)
490 #define RT_MESSAGEABLEA MAKEINTRESOURCEA(11)
491 #define RT_MESSAGEABLEW MAKEINTRESOURCEW(11)
492 #define RT_MESSAGEABLE WINELIB_NAME_AW(RT_MESSAGEABLE)
493 #define RT_GROUP_CURSORA MAKEINTRESOURCEA(12)
494 #define RT_GROUP_CURSORW MAKEINTRESOURCEW(12)
495 #define RT_GROUP_CURSOR WINELIB_NAME_AW(RT_GROUP_CURSOR)
496 #define RT_GROUP_ICONA MAKEINTRESOURCEA(14)
497 #define RT_GROUP_ICONW MAKEINTRESOURCEW(14)
498 #define RT_GROUP_ICON WINELIB_NAME_AW(RT_GROUP_ICON)
499 #define RT_VERSIONA MAKEINTRESOURCEA(16)
500 #define RT_VERSIONW MAKEINTRESOURCEW(16)
501 #define RT_VERSION WINELIB_NAME_AW(RT_VERSION)
502 #define RT_DLGINCLUDEA MAKEINTRESOURCEA(17)
503 #define RT_DLGINCLUDEW MAKEINTRESOURCEW(17)
504 #define RT_DLGINCLUDE WINELIB_NAME_AW(RT_DLGINCLUDE)
505 #define RT_PLUGPLAYA MAKEINTRESOURCEA(19)
506 #define RT_PLUGPLAYW MAKEINTRESOURCEW(19)
507 #define RT_PLUGPLAY WINELIB_NAME_AW(RT_PLUGPLAY)
508 #define RT_VXDA MAKEINTRESOURCEA(20)
509 #define RT_VXDW MAKEINTRESOURCEW(20)
510 #define RT_VXD WINELIB_NAME_AW(RT_VXD)
511 #define RT_ANICURSORA MAKEINTRESOURCEA(21)
512 #define RT_ANICURSOW MAKEINTRESOURCEW(21)
513 #define RT_ANICURSOR WINELIB_NAME_AW(RT_ANICURSOR)
514 #define RT_ANIICONA MAKEINTRESOURCEA(22)
515 #define RT_ANIICONW MAKEINTRESOURCEW(22)
516 #define RT_ANIICON WINELIB_NAME_AW(RT_ANIICON)
517 #define RT_HTMLA MAKEINTRESOURCEA(23)
518 #define RT_HTMLW MAKEINTRESOURCEW(23)
519 #define RT_HTML WINELIB_NAME_AW(RT_HTML)
520
521
522 /* cbWndExtra bytes for dialog class */
523 #define DLGWINDOWEXTRA 30
524
525 /* Button control styles */
526 #define BS_PUSHBUTTON 0x00000000L
527 #define BS_DEFPUSHBUTTON 0x00000001L
528 #define BS_CHECKBOX 0x00000002L
529 #define BS_AUTOCHECKBOX 0x00000003L
530 #define BS_RADIOBUTTON 0x00000004L
531 #define BS_3STATE 0x00000005L
532 #define BS_AUTO3STATE 0x00000006L
533 #define BS_GROUPBOX 0x00000007L
534 #define BS_USERBUTTON 0x00000008L
535 #define BS_AUTORADIOBUTTON 0x00000009L
536 #define BS_OWNERDRAW 0x0000000BL
537 #define BS_LEFTTEXT 0x00000020L
538 #define BS_RIGHTBUTTON BS_LEFTTEXT
539
540 #define BS_TEXT 0x00000000L
541 #define BS_ICON 0x00000040L
542 #define BS_BITMAP 0x00000080L
543 #define BS_LEFT 0x00000100L
544 #define BS_RIGHT 0x00000200L
545 #define BS_CENTER 0x00000300L
546 #define BS_TOP 0x00000400L
547 #define BS_BOTTOM 0x00000800L
548 #define BS_VCENTER 0x00000C00L
549 #define BS_PUSHLIKE 0x00001000L
550 #define BS_MULTILINE 0x00002000L
551 #define BS_NOTIFY 0x00004000L
552 #define BS_FLAT 0x00008000L
553

```

```
554  /* Dialog styles */
555 #define DS_ABSALIGN      0x0001
556 #define DS_SYSMODAL      0x0002
557 #define DS_3DLOOK        0x0004  /* win95 */
558 #define DS_FIXEDSYS      0x0008  /* win95 */
559 #define DS_NOFAILCREATE   0x0010  /* win95 */
560 #define DS_LOCALEEDIT    0x0020
561 #define DS_SETFONT        0x0040
562 #define DS_MODALFRAME     0x0080
563 #define DS_NOIDLEMSG      0x0100
564 #define DS_SETFOREGROUND  0x0200  /* win95 */
565 #define DS_CONTROL        0x0400  /* win95 */
566 #define DS_CENTER         0x0800  /* win95 */
567 #define DS_CENTERMOUSE    0x1000  /* win95 */
568 #define DS_CONTEXTHELP    0x2000  /* win95 */
569
570
571  /* Dialog messages */
572 #define DM_GETDEFID        (WM_USER+0)
573 #define DM_SETDEFID        (WM_USER+1)
574 #define DM_REPOSITION      (WM_USER+2)
575
576 #define DC_HASDEFID        0x534b
577
578  /* Owner draw control types */
579 #define ODT_MENU           1
580 #define ODT_LISTBOX        2
581 #define ODT_COMBOBOX       3
582 #define ODT_BUTTON         4
583 #define ODT_STATIC         5
584
585  /* Owner draw actions */
586 #define ODA_DRAWENTIRE     0x0001
587 #define ODA_SELECT         0x0002
588 #define ODA_FOCUS          0x0004
589
590  /* Owner draw state */
591 #define ODS_SELECTED       0x0001
592 #define ODS_GRAYED         0x0002
593 #define ODS_DISABLED       0x0004
594 #define ODS_CHECKED        0x0008
595 #define ODS_FOCUS          0x0010
596 #define ODS_COMBOBOXEDIT   0x1000
597 #define ODS_HOTLIGHT       0x0040
598 #define ODS_INACTIVE       0x0080
599
600  /* Edit control styles */
601 #define ES_LEFT            0x00000000
602 #define ES_CENTER          0x00000001
603 #define ES_RIGHT           0x00000002
604 #define ES_MULTILINE       0x00000004
605 #define ES_UPPERCASE       0x00000008
606 #define ES_LOWERCASE       0x00000010
607 #define ES_PASSWORD        0x00000020
608 #define ES_AUTOVSCROLL     0x00000040
609 #define ES_AUTOHSCROLL     0x00000080
610 #define ES_NOHIDESEL       0x00000100
611 #define ES_COMBO           0x00000200  /* Undocumented. Parent is a combobox */
612 #define ES_OEMCONVERT       0x00000400
613 #define ES_READONLY        0x00000800
614 #define ES_WANTRETURN      0x00001000
615 #define ES_NUMBER          0x00002000
616
617  /* OEM Resource Ordinal Numbers */
618 #define OBM_CLOSED          32731
619 #define OBM_TRITYPE         32732
620 #define OBM_LFARROWI        32734
621 #define OBM_RGARROWI        32735
622 #define OBM_DNARROWI        32736
623 #define OBM_UPARROWI        32737
624 #define OBM_COMBO           32738
625 #define OBM_MNARROW         32739
626 #define OBM_LFARROWD        32740
627 #define OBM_RGARROWD        32741
628 #define OBM_DNARROWD        32742
629 #define OBM_UPARROWD        32743
630 #define OBM_RESTORED        32744
631 #define OBM_ZOOMD           32745
632 #define OBM_REDUCED         32746
633 #define OBM_RESTORE         32747
634 #define OBM_ZOOM            32748
635 #define OBM_REDUCE          32749
636 #define OBM_LFARROW         32750
637 #define OBM_RGARROW         32751
638 #define OBM_DNARROW         32752
639 #define OBM_UPARROW         32753
640 #define OBM_CLOSE           32754
```

```

641 #define OBM_OLD_RESTORE      32755
642 #define OBM_OLD_ZOOM         32756
643 #define OBM_OLD_REDUCE       32757
644 #define OBM_BTNCORNERS       32758
645 #define OBM_CHECKBOXES       32759
646 #define OBM_CHECK            32760
647 #define OBM_BTFSIZE          32761
648 #define OBM_OLD_LFARROW      32762
649 #define OBM_OLD_RGARROW      32763
650 #define OBM_OLD_DNARROW      32764
651 #define OBM_OLD_UPARROW      32765
652 #define OBM_SIZE              32766
653 #define OBM_OLD_CLOSE        32767
654
655 #define OCR_NORMAL            32512
656 #define OCR_IBEAM             32513
657 #define OCR_WAIT              32514
658 #define OCR_CROSS             32515
659 #define OCR_UP                32516
660 #define OCR_SIZE              32640
661 #define OCR_ICON              32641
662 #define OCR_SIZENWSE         32642
663 #define OCR_SIZENESW         32643
664 #define OCR_SIZEWE           32644
665 #define OCR_SIZENS           32645
666 #define OCR_SIZEALL          32646
667 #define OCR_ICOCUR           32647
668 #define OCR_NO                32648
669 #define OCR_HAND              32649
670 #define OCR_APPSTARTING       32650
671 #define OCR_HELP              32651
672
673 /* only defined in wine (FIXME) */
674 #define OCR_DRAGOBJECT         32653
675
676 #define OIC_SAMPLE            32512
677 #define OIC_HAND              32513
678 #define OIC_QUES              32514
679 #define OIC_BANG              32515
680 #define OIC_NOTE              32516
681 #define OIC_WINLOGO           32517
682 #define OIC_WARNING           OIC_BANG
683 #define OIC_ERROR             OIC_HAND
684 #define OIC_INFORMATION       OIC_NOTE
685
686 #ifndef NOCOLOR
687
688 #define COLOR_SCROLLBAR        0
689 #define COLOR_BACKGROUND       1
690 #define COLOR_ACTIVECAPTION    2
691 #define COLOR_INACTIVECAPTION  3
692 #define COLOR_MENU             4
693 #define COLOR_WINDOW           5
694 #define COLOR_WINDOWFRAME      6
695 #define COLOR_MENUTEXT         7
696 #define COLOR_WINDOWTEXT       8
697 #define COLOR_CAPTIONTEXT      9
698 #define COLOR_ACTIVEBORDER     10
699 #define COLOR_INACTIVEBORDER   11
700 #define COLOR_APPWORKSPACE     12
701 #define COLOR_HIGHLIGHT        13
702 #define COLOR_HIGHLIGHTTEXT    14
703 #define COLOR_BTNFACE           15
704 #define COLOR_BTNSHADOW         16
705 #define COLOR_GRAYTEXT         17
706 #define COLOR_BTNTEXT           18
707 #define COLOR_INACTIVECAPTIONTEXT 19
708 #define COLOR_BTNHIGHLIGHT     20
709 /* win95 colors */
710 #define COLOR_3DDKSHADOW        21
711 #define COLOR_3DLIGHT           22
712 #define COLOR_INFOTEXT         23
713 #define COLOR_INFOBK           24
714 #define COLOR_DESKTOP           COLOR_BACKGROUND
715 #define COLOR_3DFACE            COLOR_BTNFACE
716 #define COLOR_3DSHADOW         COLOR_BTNSHADOW
717 #define COLOR_3DHIGHLIGHT      COLOR_BTNHIGHLIGHT
718 #define COLOR_3DHILIGHT        COLOR_BTNHIGHLIGHT
719 #define COLOR_BTNHILIGHT       COLOR_BTNHIGHLIGHT
720 /* win98 colors */
721 #define COLOR_ALTERNATEBTNFACE  25 /* undocumented, constant's name unknown */
722 #define COLOR_HOTLIGHT          26
723 #define COLOR_GRADIENTACTIVECAPTION 27
724 #define COLOR_GRADIENTINACTIVECAPTION 28
725
726 /* WM_CTLCOLOR values */
727 #define CTLCOLOR_MSGBOX         0

```

```

728 #define CTLCOLOR_EDIT                1
729 #define CTLCOLOR_LISTBOX              2
730 #define CTLCOLOR_BTN                  3
731 #define CTLCOLOR_DLG                  4
732 #define CTLCOLOR_SCROLLBAR            5
733 #define CTLCOLOR_STATIC                6
734
735 COLORREF WINAPI GetSysColor(INT);
736 BOOL WINAPI SetSysColors(INT, const INT*, const COLORREF*);
737
738 #endif /* NOCOLOR */
739
740 /* Edit control messages */
741 #define EM_GETSEL                      0x00b0
742 #define EM_SETSEL                      0x00b1
743 #define EM_GETRECT                     0x00b2
744 #define EM_SETRECT                     0x00b3
745 #define EM_SETRECTNP                  0x00b4
746 #define EM_SCROLL                     0x00b5
747 #define EM_LINESCROLL                  0x00b6
748 #define EM_SCROLLCARET                 0x00b7
749 #define EM_GETMODIFY                   0x00b8
750 #define EM_SETMODIFY                   0x00b9
751 #define EM_GETLINECOUNT               0x00ba
752 #define EM_LINEINDEX                  0x00bb
753 #define EM_SETHANDLE                   0x00bc
754 #define EM_GETHANDLE                   0x00bd
755 #define EM_GETTHUMB                    0x00be
756 /* FIXME : missing from specs 0x00bf and 0x00c0 */
757 #define EM_LINELENGTH                  0x00c1
758 #define EM_REPLACESEL                  0x00c2
759 /* FIXME : missing from specs 0x00c3 */
760 #define EM_GETLINE                     0x00c4
761 #define EM_LIMITTEXT                   0x00c5
762 #define EM_CANUNDO                     0x00c6
763 #define EM_UNDO                        0x00c7
764 #define EM_FMTLINES                    0x00c8
765 #define EM_LINEFROMCHAR                0x00c9
766 /* FIXME : missing from specs 0x00ca */
767 #define EM_SETTABSTOPS                 0x00cb
768 #define EM_SETPASSWORDCHAR             0x00cc
769 #define EM_EMPTYUNDOBUFFER             0x00cd
770 #define EM_GETFIRSTVISIBLELINE         0x00ce
771 #define EM_SETREADONLY                 0x00cf
772 #define EM_SETWORDBREAKPROC            0x00d0
773 #define EM_GETWORDBREAKPROC            0x00d1
774 #define EM_GETPASSWORDCHAR             0x00d2
775 #define EM_SETMARGINS                  0x00d3
776 #define EM_GETMARGINS                  0x00d4
777 #define EM_GETLIMITTEXT                0x00d5
778 #define EM_POSFROMCHAR                 0x00d6
779 #define EM_CHARFROMPOS                 0x00d7
780 /* a name change since win95 */
781 #define EM_SETLIMITTEXT                EM_LIMITTEXT
782
783 /* EDITWORDBREAKPROC code values */
784 #define WB_LEFT                        0
785 #define WB_RIGHT                       1
786 #define WB_ISDELIMITER                 2
787
788 /* Edit control notification codes */
789 #define EN_SETFOCUS                    0x0100
790 #define EN_KILLFOCUS                   0x0200
791 #define EN_CHANGE                      0x0300
792 #define EN_UPDATE                      0x0400
793 #define EN_ERRSPACE                    0x0500
794 #define EN_MAXTEXT                     0x0501
795 #define EN_HSCROLL                     0x0601
796 #define EN_VSCROLL                     0x0602
797
798 /* New since win95 : EM_SETMARGIN parameters */
799 #define EC_LEFTMARGIN                  0x0001
800 #define EC_RIGHTMARGIN                 0x0002
801 #define EC_USEFONTINFO                 0xffff
802
803
804 /* GetSystemMetrics() codes */
805 #define SM_CXSCREEN                     0
806 #define SM_CYSCREEN                     1
807 #define SM_CXVSCROLL                   2
808 #define SM_CYHSCROLL                   3
809 #define SM_CYCAPTION                   4
810 #define SM_CXBORDER                     5
811 #define SM_CYBORDER                     6
812 #define SM_CXDLGFRAME                  7
813 #define SM_CYDLGFRAME                  8
814 #define SM_CYVTHUMB                     9

```

```

815 #define SM_CXHTHUMB      10
816 #define SM_CXICON        11
817 #define SM_CYICON        12
818 #define SM_CXCURSOR      13
819 #define SM_CYCURSOR      14
820 #define SM_CYMENU        15
821 #define SM_CXFULLSCREEN   16
822 #define SM_CYFULLSCREEN   17
823 #define SM_CYKANJIWINDOW 18
824 #define SM_MOUSEPRESENT   19
825 #define SM_CVSCROLL       20
826 #define SM_CXHSCROLL     21
827 #define SM_DEBUG          22
828 #define SM_SWAPBUTTON     23
829 #define SM_RESERVED1      24
830 #define SM_RESERVED2      25
831 #define SM_RESERVED3      26
832 #define SM_RESERVED4      27
833 #define SM_CXMIN          28
834 #define SM_CYMIN          29
835 #define SM_CXSIZE         30
836 #define SM_CYSIZE         31
837 #define SM_CXFRAME        32
838 #define SM_CYFRAME        33
839 #define SM_CXMINTRACK     34
840 #define SM_CYMINTRACK     35
841 #define SM_CXDOUBLECLK    36
842 #define SM_CYDOUBLECLK    37
843 #define SM_CXICONSPACING  38
844 #define SM_CYICONSPACING  39
845 #define SM_MENUDROPALIGNMENT 40
846 #define SM_PENWINDOWS     41
847 #define SM_DBCSENABLED    42
848 #define SM_CMOUSEBUTTONS  43
849 #define SM_CXFIXEDFRAME    SM_CXDLGFRAME
850 #define SM_CYFIXEDFRAME    SM_CYDLGFRAME
851 #define SM_CXSIZEFRAME     SM_CXFRAME
852 #define SM_CYSIZEFRAME     SM_CYFRAME
853 #define SM_SECURE          44
854 #define SM_CXEDGE         45
855 #define SM_CYEDGE         46
856 #define SM_CXMINSPACING   47
857 #define SM_CYMINSPACING   48
858 #define SM_CXSMICON       49
859 #define SM_CYSMICON       50
860 #define SM_CYSMCAPTION    51
861 #define SM_CXSMSIZE       52
862 #define SM_CYSMSIZE       53
863 #define SM_CXMENUSIZE     54
864 #define SM_CYMENUSIZE     55
865 #define SM_ARRANGE        56
866 #define SM_CXMINIMIZED    57
867 #define SM_CYMINIMIZED    58
868 #define SM_CXMAXTRACK     59
869 #define SM_CYMAXTRACK     60
870 #define SM_CXMAXIMIZED    61
871 #define SM_CYMAXIMIZED    62
872 #define SM_NETWORK        63
873 #define SM_CLEANBOOT       67
874 #define SM_CXDRAG         68
875 #define SM_CYDRAG         69
876 #define SM_SHOWSOUNDS     70
877 #define SM_CXMENUCHECK    71
878 #define SM_CYMENUCHECK    72
879 #define SM_SLOWMACHINE     73
880 #define SM_MIDEASTENABLED  74
881 #define SM_MOUSEWHEELPRESENT 75
882 #define SM_XVIRTUALSCREEN  76
883 #define SM_YVIRTUALSCREEN  77
884 #define SM_CXVIRTUALSCREEN  78
885 #define SM_CYVIRTUALSCREEN  79
886 #define SM_CMONITORS       80
887 #define SM_SAMEDISPLAYFORMAT 81
888 #define SM_CMETRICS        83
889
890
891 /* Messages */
892
893 /* WM_GETDLGCODE values */
894
895
896 #define WM_NULL             0x0000
897 #define WM_CREATE           0x0001
898 #define WM_DESTROY          0x0002
899 #define WM_MOVE             0x0003
900 #define WM_SIZEWAIT         0x0004
901 #define WM_SIZE             0x0005

```

```

902 #define WM_ACTIVATE          0x0006
903 #define WM_SETFOCUS          0x0007
904 #define WM_KILLFOCUS         0x0008
905 #define WM_SETVISIBLE         0x0009
906 #define WM_ENABLE            0x000a
907 #define WM_SETREDRAW          0x000b
908 #define WM_SETTEXT            0x000c
909 #define WM_GETTEXT            0x000d
910 #define WM_GETTEXTLENGTH     0x000e
911 #define WM_PAINT              0x000f
912 #define WM_CLOSE              0x0010
913 #define WM_QUERYENDSESSION    0x0011
914 #define WM_QUIT               0x0012
915 #define WM_QUERYOPEN          0x0013
916 #define WM_ERASEBKGDND        0x0014
917 #define WM_SYSCOLORCHANGE     0x0015
918 #define WM_ENDSESSION         0x0016
919 #define WM_SYSTEMERROR        0x0017
920 #define WM_SHOWWINDOW         0x0018
921 #define WM_CTLCOLOR           0x0019
922 #define WM_WININICHANGE       0x001a
923 #define WM_SETTINGCHANGE      WM_WININICHANGE
924 #define WM_DEVMODECHANGE      0x001b
925 #define WM_ACTIVATEAPP        0x001c
926 #define WM_FONTCHANGE         0x001d
927 #define WM_TIMECHANGE         0x001e
928 #define WM_CANCELMODE        0x001f
929 #define WM_SETCURSOR          0x0020
930 #define WM_MOUSEACTIVATE       0x0021
931 #define WM_CHILDACTIVATE      0x0022
932 #define WM_QUEUESYNC          0x0023
933 #define WM_GETMINMAXINFO      0x0024
934
935 #define WM_PAINTICON           0x0026
936 #define WM_ICONERASEBKGDND    0x0027
937 #define WM_NEXTDLGCTL         0x0028
938 #define WM_ALTTABACTIVE       0x0029
939 #define WM_SPOOLERSTATUS      0x002a
940 #define WM_DRAWITEM           0x002b
941 #define WM_MEASUREITEM        0x002c
942 #define WM_DELETEITEM         0x002d
943 #define WM_VKEYTOITEM         0x002e
944 #define WM_CHARTOITEM         0x002f
945 #define WM_SETFONT            0x0030
946 #define WM_GETFONT            0x0031
947 #define WM_SETHOTKEY          0x0032
948 #define WM_GETHOTKEY          0x0033
949 #define WM_FILESYSCCHANGE     0x0034
950 #define WM_ISACTIVEICON       0x0035
951 #define WM_QUERYPARKICON      0x0036
952 #define WM_QUERYDRAGICON      0x0037
953 #define WM_QUEYSAVESTATE      0x0038
954 #define WM_COMPAREITEM        0x0039
955 #define WM_TESTING            0x003a
956
957 #define WM_OTHERWINDOWCREATED  0x003c
958 #define WM_OTHERWINDOWDESTROYED 0x003d
959 #define WM_ACTIVATESHELLWINDOW 0x003e
960
961 #define WM_COMPACTING          0x0041
962
963 #define WM_COMMNOTIFY          0x0044
964 #define WM_WINDOWPOSCHANGING   0x0046
965 #define WM_WINDOWPOSCHANGED    0x0047
966 #define WM_POWER               0x0048
967
968 /* Win32 4.0 messages */
969 #define WM_COPYDATA            0x004a
970 #define WM_CANCELJOURNAL       0x004b
971 #define WM_NOTIFY              0x004e
972 #define WM_INPUTLANGCHANGEREQUEST 0x0050
973 #define WM_INPUTLANGCHANGE     0x0051
974 #define WM_TCARD               0x0052
975 #define WM_HELP                0x0053
976 #define WM_USERCHANGED         0x0054
977 #define WM_NOTIFYFORMAT        0x0055
978
979 #define WM_CONTEXTMENU          0x007b
980 #define WM_STYLECHANGING        0x007c
981 #define WM_STYLECHANGED         0x007d
982 #define WM_DISPLAYCHANGE        0x007e
983 #define WM_GETICON             0x007f
984 #define WM_SETICON             0x0080
985
986 /* Non-client system messages */
987 #define WM_NCCREATE             0x0081
988 #define WM_NCDESTROY            0x0082

```



```
989 #define WM_NCCALCSIZE      0x0083
990 #define WM_NCHITTEST       0x0084
991 #define WM_NCPAINT         0x0085
992 #define WM_NCACTIVATE      0x0086
993
994 #define WM_GETDLGCODE       0x0087
995 #define WM_SYNCPAINT        0x0088
996 #define WM_SYNCTASK         0x0089
997
998 /* Non-client mouse messages */
999 #define WM_NCMOUSEMOVE      0x00a0
1000 #define WM_NCLBUTTONDOWN    0x00a1
1001 #define WM_NCLBUTTONUP      0x00a2
1002 #define WM_NCLBUTTONDBLCLK  0x00a3
1003 #define WM_NCRBUTTONDOWN    0x00a4
1004 #define WM_NCRBUTTONUP      0x00a5
1005 #define WM_NCRBUTTONDBLCLK  0x00a6
1006 #define WM_NCMBUTTONDOWN    0x00a7
1007 #define WM_NCMBUTTONUP      0x00a8
1008 #define WM_NCMBUTTONDBLCLK  0x00a9
1009
1010 #define WM_NCXBUTTONDOWN    0x00ab
1011 #define WM_NCXBUTTONUP      0x00ac
1012 #define WM_NCXBUTTONDBLCLK  0x00ad
1013
1014 /* Keyboard messages */
1015 #define WM_KEYDOWN          0x0100
1016 #define WM_KEYUP            0x0101
1017 #define WM_CHAR             0x0102
1018 #define WM_DEADCHAR         0x0103
1019 #define WM_SYSKEYDOWN       0x0104
1020 #define WM_SYSKEYUP         0x0105
1021 #define WM_SYSCHAR          0x0106
1022 #define WM_SYSDEADCHAR      0x0107
1023 #define WM_KEYFIRST         WM_KEYDOWN
1024 #define WM_KEYLAST          0x0108
1025
1026 /* Win32 4.0 messages for IME */
1027 #define WM_IME_STARTCOMPOSITION 0x010d
1028 #define WM_IME_ENDCOMPOSITION  0x010e
1029 #define WM_IME_COMPOSITION     0x010f
1030 #define WM_IME_KEYLAST         0x010f
1031
1032 #define WM_INITDIALOG         0x0110
1033 #define WM_COMMAND            0x0111
1034 #define WM_SYSCOMMAND         0x0112
1035 #define WM_TIMER              0x0113
1036 #define WM_SYSTIMER           0x0118
1037
1038 /* scroll messages */
1039 #define WM_HSCROLL            0x0114
1040 #define WM_VSCROLL            0x0115
1041
1042 /* Menu messages */
1043 #define WM_INITMENU           0x0116
1044 #define WM_INITMENUPOPUP     0x0117
1045
1046 #define WM_MENUSELECT         0x011f
1047 #define WM_MENUCHAR           0x0120
1048 #define WM_ENTERIDLE          0x0121
1049
1050 #define WM_LBTRACKPOINT       0x0131
1051
1052 /* Win32 CTLCOLOR messages */
1053 #define WM_CTLCOLORMSGBOX     0x0132
1054 #define WM_CTLCOLOREDIT       0x0133
1055 #define WM_CTLCOLORLISTBOX    0x0134
1056 #define WM_CTLCOLORBTN        0x0135
1057 #define WM_CTLCOLORDLG        0x0136
1058 #define WM_CTLCOLORSCROLLBAR  0x0137
1059 #define WM_CTLCOLORSTATIC     0x0138
1060
1061 /* Mouse messages */
1062 #define WM_MOUSEMOVE          0x0200
1063 #define WM_LBUTTONDOWN        0x0201
1064 #define WM_LBUTTONUP          0x0202
1065 #define WM_LBUTTONDBLCLK      0x0203
1066 #define WM_RBUTTONDOWN        0x0204
1067 #define WM_RBUTTONUP          0x0205
1068 #define WM_RBUTTONDBLCLK      0x0206
1069 #define WM_MBUTTONDOWN        0x0207
1070 #define WM_MBUTTONUP          0x0208
1071 #define WM_MBUTTONDBLCLK      0x0209
1072 #define WM_MOUSEWHEEL         0x020a
1073 #define WM_XBUTTONDOWN        0x020b
1074 #define WM_XBUTTONUP          0x020c
1075 #define WM_XBUTTONDBLCLK      0x020d
```

```
1076
1077 #define WM_MOUSEFIRST      0x0200
1078 #define WM_MOUSELAST      0x020D
1079
1080 #define WHEEL_DELTA        120
1081 #define WHEEL_PAGESCROLL  (UINT_MAX)
1082 #define WM_PARENTNOTIFY    0x0210
1083 #define WM_ENTERMENULOOP   0x0211
1084 #define WM_EXITMENULOOP    0x0212
1085 #define WM_NEXTMENU        0x0213
1086
1087 /* Win32 4.0 messages */
1088 #define WM_SIZING           0x0214
1089 #define WM_CAPTURECHANGED  0x0215
1090 #define WM_MOVING           0x0216
1091 #define WM_POWERBROADCAST   0x0218
1092 #define WM_DEVICECHANGE     0x0219
1093
1094 /* wParam for WM_SIZING message */
1095 #define WMSZ_LEFT           1
1096 #define WMSZ_RIGHT          2
1097 #define WMSZ_TOP            3
1098 #define WMSZ_TOPLEFT        4
1099 #define WMSZ_TOPRIGHT       5
1100 #define WMSZ_BOTTOM         6
1101 #define WMSZ_BOTTOMLEFT     7
1102 #define WMSZ_BOTTOMRIGHT    8
1103
1104 /* MDI messages */
1105 #define WM_MDICREATE        0x0220
1106 #define WM_MDIDESTROY       0x0221
1107 #define WM_MDIACTIVATE      0x0222
1108 #define WM_MDIRESTORE       0x0223
1109 #define WM_MDINEXT          0x0224
1110 #define WM_MDIMAXIMIZE      0x0225
1111 #define WM_MDTITLE          0x0226
1112 #define WM_MDICASCADE       0x0227
1113 #define WM_MDIICONARRANGE   0x0228
1114 #define WM_MDIGETACTIVE     0x0229
1115 #define WM_MDIREFRESHMENU   0x0234
1116
1117 /* D&D messages */
1118 #define WM_DROPBJECT        0x022A
1119 #define WM_QUERYDROBJECT    0x022B
1120 #define WM_BEGINDRAG        0x022C
1121 #define WM_DRAGLOOP         0x022D
1122 #define WM_DRAGSELECT       0x022E
1123 #define WM_DRAGMOVE         0x022F
1124 #define WM_MDISETMENU       0x0230
1125
1126 #define WM_ENTERSIZEMOVE    0x0231
1127 #define WM_EXITSIZEMOVE     0x0232
1128 #define WM_DROPFILES        0x0233
1129
1130
1131 /* Win32 4.0 messages for IME */
1132 #define WM_IME_SETCONTEXT    0x0281
1133 #define WM_IME_NOTIFY       0x0282
1134 #define WM_IME_CONTROL       0x0283
1135 #define WM_IME_COMPOSITIONFULL 0x0284
1136 #define WM_IME_SELECT        0x0285
1137 #define WM_IME_CHAR          0x0286
1138 /* Win32 5.0 messages for IME */
1139 #define WM_IME_REQUEST       0x0288
1140
1141 /* Win32 4.0 messages for IME */
1142 #define WM_IME_KEYDOWN       0x0290
1143 #define WM_IME_KEYUP         0x0291
1144
1145 /* Clipboard command messages */
1146 #define WM_CUT                0x0300
1147 #define WM_COPY               0x0301
1148 #define WM_PASTE              0x0302
1149 #define WM_CLEAR              0x0303
1150 #define WM_UNDO               0x0304
1151
1152 /* Clipboard owner messages */
1153 #define WM_RENDERFORMAT       0x0305
1154 #define WM_RENDERALLFORMATS   0x0306
1155 #define WM_DESTROYCLIPBOARD   0x0307
1156
1157 /* Clipboard viewer messages */
1158 #define WM_DRAWCLIPBOARD      0x0308
1159 #define WM_PAINTCLIPBOARD     0x0309
1160 #define WM_VSCROLLCLIPBOARD   0x030A
1161 #define WM_SIZECLIPBOARD      0x030B
1162 #define WM_ASKCBFORMATNAME    0x030C
```

```

1163 #define WM_CHANGECHAIN 0x030D
1164 #define WM_HSCROLLCLIPBOARD 0x030E
1165
1166 #define WM_QUERYNEWPALETTE 0x030F
1167 #define WM_PALETTEISCHANGING 0x0310
1168 #define WM_PALETTECHANGED 0x0311
1169 #define WM_HOTKEY 0x0312
1170
1171 #define WM_PRINT 0x0317
1172 #define WM_PRINTCLIENT 0x0318
1173
1174 #define WM_PENWINFIRST 0x0380
1175 #define WM_PENWINLAST 0x038F
1176
1177
1178 #define WM_APP 0x8000
1179
1180 /* MsgWaitForMultipleObjectsEx flags */
1181 #define MWMO_WAITALL 0x0001
1182 #define MWMO_ALERTABLE 0x0002
1183 #define MWMO_INPUTAVAILABLE 0x0004
1184
1185 #define DLGC_WANTARROWS 0x0001
1186 #define DLGC_WANTTAB 0x0002
1187 #define DLGC_WANTALLKEYS 0x0004
1188 #define DLGC_WANTMESSAGE 0x0004
1189 #define DLGC_HASSETSEL 0x0008
1190 #define DLGC_DEFPUSHBUTTON 0x0010
1191 #define DLGC_UNDEFPUSHBUTTON 0x0020
1192 #define DLGC_RADIOBUTTON 0x0040
1193 #define DLGC_WANTCHARS 0x0080
1194 #define DLGC_STATIC 0x0100
1195 #define DLGC_BUTTON 0x2000
1196
1197 /* Standard dialog button IDs */
1198 #define IDOK 1
1199 #define IDCANCEL 2
1200 #define IDABORT 3
1201 #define IDRETRY 4
1202 #define IDIGNORE 5
1203 #define IDYES 6
1204 #define IDNO 7
1205 #define IDCLOSE 8
1206 #define IDHELP 9
1207
1208 /***** Window classes *****/
1209
1210 typedef struct tagCREATESTRUCTA
1211 {
1212     LPVOID lpCreateParams;
1213     HINSTANCE hInstance;
1214     HMENU hMenu;
1215     HWND hwndParent;
1216     INT cy;
1217     INT cx;
1218     INT y;
1219     INT x;
1220     LONG style;
1221     LPCSTR lpszName;
1222     LPCSTR lpszClass;
1223     DWORD dwExStyle;
1224 } CREATESTRUCTA, *LPCREATESTRUCTA;
1225
1226 typedef struct
1227 {
1228     LPVOID lpCreateParams;
1229     HINSTANCE hInstance;
1230     HMENU hMenu;
1231     HWND hwndParent;
1232     INT cy;
1233     INT cx;
1234     INT y;
1235     INT x;
1236     LONG style;
1237     LPCWSTR lpszName;
1238     LPCWSTR lpszClass;
1239     DWORD dwExStyle;
1240 } CREATESTRUCTW, *LPCREATESTRUCTW;
1241
1242 DECL_WINELIB_TYPE_AW(CREATESTRUCT)
1243 DECL_WINELIB_TYPE_AW(LPCREATESTRUCT)
1244
1245 typedef struct
1246 {
1247     HDC hdc;
1248     BOOL fErase;
1249     RECT rcPaint;

```

```

1250     BOOL    fRestore;
1251     BOOL    fIncUpdate;
1252     BYTE    rgbReserved[32];
1253 } PAINTSTRUCT, *PPAINTSTRUCT, *LPPAINTSTRUCT;
1254
1255 typedef struct
1256 {
1257     HMENU    hWindowMenu;
1258     UINT     idFirstChild;
1259 } CLIENTCREATESTRUCT, *LPCCLIENTCREATESTRUCT;
1260
1261
1262 typedef struct
1263 {
1264     LPCSTR    szClass;
1265     LPCSTR    szTitle;
1266     HINSTANCE hOwner;
1267     INT       x;
1268     INT       y;
1269     INT       cx;
1270     INT       cy;
1271     DWORD     style;
1272     LPARAM    lParam;
1273 } MDICREATESTRUCTA, *LPMDICREATESTRUCTA;
1274
1275 typedef struct
1276 {
1277     LPCWSTR    szClass;
1278     LPCWSTR    szTitle;
1279     HINSTANCE  hOwner;
1280     INT        x;
1281     INT        y;
1282     INT        cx;
1283     INT        cy;
1284     DWORD     style;
1285     LPARAM    lParam;
1286 } MDICREATESTRUCTW, *LPMDICREATESTRUCTW;
1287
1288 DECL_WINELIB_TYPE_AW(MDICREATESTRUCT)
1289 DECL_WINELIB_TYPE_AW(LPMDICREATESTRUCT)
1290
1291 #define MDITILE_VERTICAL    0x0000
1292 #define MDITILE_HORIZONTAL 0x0001
1293 #define MDITILE_SKIPDISABLED 0x0002
1294
1295 #define MDIS_ALLCHILDSTYLES 0x0001
1296
1297 typedef struct {
1298     DWORD    styleOld;
1299     DWORD    styleNew;
1300 } STYLESTRUCT, *LPSTYLESTRUCT;
1301
1302 #define WC_DIALOGA MAKEINTATOMA(0x8002)
1303 #define WC_DIALOGW MAKEINTATOMW(0x8002)
1304 #define WC_DIALOG    WINELIB_NAME_AW(WC_DIALOG)
1305
1306 /* Offsets for GetWindowLong() and GetWindowWord() */
1307 #define GWL_USERDATA    (-21)
1308 #define GWL_EXSTYLE     (-20)
1309 #define GWL_STYLE       (-16)
1310 #define GWL_ID          (-12)
1311 #define GWL_HWNDPARENT  (-8)
1312 #define GWL_HINSTANCE   (-6)
1313 #define GWL_WNDPROC     (-4)
1314 #define DWL_MSGRESULT    0
1315 #define DWL_DLGPROC      4
1316 #define DWL_USER         8
1317
1318 /* GetWindow() constants */
1319 #define GW_HWNDFIRST    0
1320 #define GW_HWNDLAST    1
1321 #define GW_HWNDNEXT    2
1322 #define GW_HWNDPREV    3
1323 #define GW_OWNER        4
1324 #define GW_CHILD        5
1325
1326 /* GetAncestor() constants */
1327 #define GA_PARENT        1
1328 #define GA_ROOT          2
1329 #define GA_ROOTOWNER    3
1330
1331 /* WM_GETMINMAXINFO struct */
1332 typedef struct
1333 {
1334     POINT    ptReserved;
1335     POINT    ptMaxSize;
1336     POINT    ptMaxPosition;

```

```

1337     POINT    ptMinTrackSize;
1338     POINT    ptMaxTrackSize;
1339 } MINMAXINFO, *PMINMAXINFO, *LPMINMAXINFO;
1340
1341
1342 /* RedrawWindow() flags */
1343 #define RDW_INVALIDATE      0x0001
1344 #define RDW_INTERNALPAINT  0x0002
1345 #define RDW_ERASE           0x0004
1346 #define RDW_VALIDATE       0x0008
1347 #define RDW_NOINTERNALPAINT 0x0010
1348 #define RDW_NOERASE        0x0020
1349 #define RDW_NOCHILDREN     0x0040
1350 #define RDW_ALLCHILDREN    0x0080
1351 #define RDW_UPDATENOW      0x0100
1352 #define RDW_ERASENOW       0x0200
1353 #define RDW_FRAME          0x0400
1354 #define RDW_NOFRAME        0x0800
1355
1356 /* debug flags */
1357 #define DBGFILL_ALLOC      0xfd
1358 #define DBGFILL_FREE       0xfb
1359 #define DBGFILL_BUFFER     0xf9
1360 #define DBGFILL_STACK      0xf7
1361
1362 /* WM_WINDOWPOSCHANGING/CHANGED struct */
1363 typedef struct tagWINDOWPOS
1364 {
1365     HWND    hwnd;
1366     HWND    hwndInsertAfter;
1367     INT     x;
1368     INT     y;
1369     INT     cx;
1370     INT     cy;
1371     UINT    flags;
1372 } WINDOWPOS, *PWINDOWPOS, *LPWINDOWPOS;
1373
1374
1375 /* WM_MOUSEACTIVATE return values */
1376 #define MA_ACTIVATE      1
1377 #define MA_ACTIVATEANDEAT 2
1378 #define MA_NOACTIVATE    3
1379 #define MA_NOACTIVATEANDEAT 4
1380
1381 /* WM_ACTIVATE wParam values */
1382 #define WA_INACTIVE      0
1383 #define WA_ACTIVE        1
1384 #define WA_CLICKACTIVE   2
1385
1386 /* WM_GETICON/WM_SETICON params values */
1387 #define ICON_SMALL       0
1388 #define ICON_BIG         1
1389
1390 /* WM_NCCALCSIZE parameter structure */
1391 typedef struct
1392 {
1393     RECT    rgrc[3];
1394     WINDOWPOS *lppos;
1395 } NCCALCSIZE_PARAMS, *LPNCCALCSIZE_PARAMS;
1396
1397
1398 /* WM_NCCALCSIZE return flags */
1399 #define WVR_ALIGNTOP      0x0010
1400 #define WVR_ALIGNLEFT    0x0020
1401 #define WVR_ALIGNBOTTOM   0x0040
1402 #define WVR_ALIGNRIGHT    0x0080
1403 #define WVR_HREDRAW       0x0100
1404 #define WVR_VREDRAW       0x0200
1405 #define WVR_REDRAW        (WVR_HREDRAW | WVR_VREDRAW)
1406 #define WVR_VALIDRECTS    0x0400
1407
1408 /* WM_NCHITTEST return codes */
1409 #define HTERROR            (-2)
1410 #define HTTRANSPARENT      (-1)
1411 #define HTNOWHERE          0
1412 #define HTCLIENT           1
1413 #define HTCAPTION          2
1414 #define HTSYSTEMMENU       3
1415 #define HTSIZE             4
1416 #define HTMENU             5
1417 #define HTHSCROLL          6
1418 #define HTVSCROLL          7
1419 #define HTMINBUTTON        8
1420 #define HTMAXBUTTON        9
1421 #define HTLEFT            10
1422 #define HTRIGHT           11
1423 #define HTTOP              12

```

```

1424 #define HTTOPLEFT          13
1425 #define HTTOPRIGHT         14
1426 #define HTBOTTOM          15
1427 #define HTBOTTOMLEFT       16
1428 #define HTBOTTOMRIGHT      17
1429 #define HTBORDER           18
1430 #define HTGROWBOX           HTSIZE
1431 #define HTREDUCE            HTMINBUTTON
1432 #define HTZOOM              HTMAXBUTTON
1433 #define HTOBJECT            19
1434 #define HTCLOSE             20
1435 #define HTHELP              21
1436 #define HTSIZEFIRST         HTLEFT
1437 #define HTSIZELAST          HTBOTTOMRIGHT
1438
1439 /* SendMessageTimeout flags */
1440 #define SMTO_NORMAL          0x0000
1441 #define SMTO_BLOCK           0x0001
1442 #define SMTO_ABORTIFHUNG     0x0002
1443 #define SMTO_NOTIMEOUTIFNOTHUNG 0x0008
1444
1445 /* WM_SYSCOMMAND parameters */
1446 #ifdef SC_SIZE /* at least HP-UX: already defined in /usr/include/sys/signal.h */
1447 #undef SC_SIZE
1448 #endif
1449 #define SC_SIZE              0xf000
1450 #define SC_MOVE              0xf010
1451 #define SC_MINIMIZE          0xf020
1452 #define SC_MAXIMIZE          0xf030
1453 #define SC_NEXTWINDOW        0xf040
1454 #define SC_PREVWINDOW        0xf050
1455 #define SC_CLOSE             0xf060
1456 #define SC_VSCROLL           0xf070
1457 #define SC_HSCROLL           0xf080
1458 #define SC_MOUSEMENU         0xf090
1459 #define SC_KEYMENU           0xf100
1460 #define SC_ARRANGE           0xf110
1461 #define SC_RESTORE           0xf120
1462 #define SC_TASKLIST          0xf130
1463 #define SC_SCREENSAVE        0xf140
1464 #define SC_HOTKEY            0xf150
1465 /* Win32 4.0 */
1466 #define SC_DEFAULT           0xf160
1467 #define SC_MONITORPOWER      0xf170
1468 #define SC_CONTEXTHELP       0xf180
1469 #define SC_SEPARATOR         0xf00f
1470
1471 /* obsolete names(SC_ICON and SC_ZOOM) */
1472 #define SC_ICON              SC_MINIMIZE
1473 #define SC_ZOOM              SC_MAXIMIZE
1474
1475
1476 #define CS_VREDRAW            0x0001
1477 #define CS_HREDRAW            0x0002
1478 #define CS_KEYCVTWINDOW      0x0004
1479 #define CS_DBLCLKS           0x0008
1480 #define CS_OWNDC             0x0020
1481 #define CS_CLASSDC           0x0040
1482 #define CS_PARENTDC          0x0080
1483 #define CS_NOKEYCVT          0x0100
1484 #define CS_NOCLOSE           0x0200
1485 #define CS_SAVEBITS          0x0800
1486 #define CS_BYTEALIGNCLIENT   0x1000
1487 #define CS_BYTEALIGNWINDOW   0x2000
1488 #define CS_GLOBALCLASS        0x4000
1489 #define CS_IME                0x00010000
1490
1491 #define PRF_CHECKVISIBLE      0x00000001L
1492 #define PRF_NONCLIENT        0x00000002L
1493 #define PRF_CLIENT            0x00000004L
1494 #define PRF_ERASEBKGDND      0x00000008L
1495 #define PRF_CHILDREN          0x00000010L
1496 #define PRF_OWNED             0x00000020L
1497
1498 /* Offsets for GetClassLong() and GetClassWord() */
1499 #define GCL_MENUNAME          (-8)
1500 #define GCL_HBRBACKGROUND     (-10)
1501 #define GCL_HCURSOR           (-12)
1502 #define GCL_HICON             (-14)
1503 #define GCL_HMODULE           (-16)
1504 #define GCL_CBWNDEXTRA         (-18)
1505 #define GCL_CBCLSEXTRA        (-20)
1506 #define GCL_WNDPROC           (-24)
1507 #define GCL_STYLE              (-26)
1508 #define GCW_ATOM              (-32)
1509 #define GCL_HICONSM            (-34)
1510

```

```

1511
1512 /***** Window hooks *****/
1513
1514 /* Hook values */
1515 #define WH_MIN                (-1)
1516 #define WH_MSGFILTER          (-1)
1517 #define WH_JOURNALRECORD      0
1518 #define WH_JOURNALPLAYBACK    1
1519 #define WH_KEYBOARD           2
1520 #define WH_GETMESSAGE         3
1521 #define WH_CALLWNDPROC        4
1522 #define WH_CBT                5
1523 #define WH_SYSMSGFILTER       6
1524 #define WH_MOUSE              7
1525 #define WH_HARDWARE           8
1526 #define WH_DEBUG              9
1527 #define WH_SHELL              10
1528 #define WH_FOREGROUNDIDLE     11
1529 #define WH_CALLWNDPROCRET     12
1530 #define WH_KEYBOARD_LL        13
1531 #define WH_MOUSE_LL           14
1532 #define WH_MAX                14
1533
1534 #define WH_MINHOOK            WH_MIN
1535 #define WH_MAXHOOK            WH_MAX
1536
1537 /* Hook action codes */
1538 #define HC_ACTION              0
1539 #define HC_GETNEXT             1
1540 #define HC_SKIP                2
1541 #define HC_NOREMOVE            3
1542 #define HC_NOREM               HC_NOREMOVE
1543 #define HC_SYSMODALON          4
1544 #define HC_SYSMODALOFF        5
1545
1546 /* CallMsgFilter() values */
1547 #define MSGF_DIALOGBOX         0
1548 #define MSGF_MESSAGEBOX        1
1549 #define MSGF_MENU              2
1550 #define MSGF_MOVE              3
1551 #define MSGF_SIZE              4
1552 #define MSGF_SCROLLBAR         5
1553 #define MSGF_NEXTWINDOW        6
1554 #define MSGF_MAX               8
1555 #define MSGF_USER              0x1000
1556 #define MSGF_DDEMGRR           0x8001
1557
1558 typedef struct
1559 {
1560     UINT            style;
1561     WNDPROC         lpfnWndProc;
1562     INT             cbClsExtra;
1563     INT             cbWndExtra;
1564     HINSTANCE        hInstance;
1565     HICON            hIcon;
1566     HCURSOR          hCursor;
1567     HBRUSH           hbrBackground;
1568     LPCSTR           lpszMenuName;
1569     LPCSTR           lpszClassName;
1570 } WNDCLASSA, *PWNDCLASSA, *LPWNDCLASSA;
1571
1572 typedef struct
1573 {
1574     UINT            style;
1575     WNDPROC         lpfnWndProc;
1576     INT             cbClsExtra;
1577     INT             cbWndExtra;
1578     HINSTANCE        hInstance;
1579     HICON            hIcon;
1580     HCURSOR          hCursor;
1581     HBRUSH           hbrBackground;
1582     LPCWSTR          lpszMenuName;
1583     LPCWSTR          lpszClassName;
1584 } WNDCLASSW, *PWNDCLASSW, *LPWNDCLASSW;
1585
1586 DECL_WINELIB_TYPE_AW(WNDCLASS)
1587 DECL_WINELIB_TYPE_AW(PWNDCLASS)
1588 DECL_WINELIB_TYPE_AW(LPWNDCLASS)
1589
1590 typedef struct {
1591     DWORD dwData;
1592     DWORD cbData;
1593     LPVOID lpData;
1594 } COPYDATASTRUCT, *PCOPYDATASTRUCT;
1595
1596 typedef struct {
1597     HMENU hmenuIn;

```

```

1598     HMENU hmenuNext;
1599     HWND  hwndNext;
1600 } MDINEXTMENU, *PMDINEXTMENU, *LPMDINEXTMENU;
1601
1602 typedef struct
1603 {
1604     DWORD    mkSize;
1605     CHAR     mkKeyList;
1606     CHAR     szKeyphrase[1];
1607 } MULTIKEYHELPA, *PMULTIKEYHELPA, *LPMULTIKEYHELPA;
1608
1609 typedef struct
1610 {
1611     DWORD    mkSize;
1612     WCHAR    mkKeyList;
1613     WCHAR    szKeyphrase[1];
1614 } MULTIKEYHELPAW, *PMULTIKEYHELPAW, *LPMULTIKEYHELPAW;
1615
1616 DECL_WINELIB_TYPE_AW(MULTIKEYHELP)
1617 DECL_WINELIB_TYPE_AW(PMULTIKEYHELP)
1618 DECL_WINELIB_TYPE_AW(LPMULTIKEYHELP)
1619
1620 typedef struct {
1621     int wStructSize;
1622     int x;
1623     int y;
1624     int dx;
1625     int dy;
1626     int wMax;
1627     CHAR rgchMember[2];
1628 } HELPWININFOA, *PHELPWININFOA, *LPHELPWININFOA;
1629
1630 typedef struct {
1631     int wStructSize;
1632     int x;
1633     int y;
1634     int dx;
1635     int dy;
1636     int wMax;
1637     WCHAR rgchMember[2];
1638 } HELPWININFOW, *PHELPWININFOW, *LPHELPWININFOW;
1639
1640 DECL_WINELIB_TYPE_AW(HELPWININFO)
1641 DECL_WINELIB_TYPE_AW(PHELPWININFO)
1642 DECL_WINELIB_TYPE_AW(LPHELPWININFO)
1643
1644 #define HELP_CONTEXT      0x0001
1645 #define HELP_QUIT         0x0002
1646 #define HELP_INDEX       0x0003
1647 #define HELP_CONTENTS    0x0003
1648 #define HELP_HELPONHELP  0x0004
1649 #define HELP_SETINDEX    0x0005
1650 #define HELP_SETCONTENTS 0x0005
1651 #define HELP_CONTEXTPOPUP 0x0008
1652 #define HELP_FORCEFILE   0x0009
1653 #define HELP_KEY         0x0101
1654 #define HELP_COMMAND     0x0102
1655 #define HELP_PARTIALKEY   0x0105
1656 #define HELP_MULTIKEY     0x0201
1657 #define HELP_SETWINPOS    0x0203
1658 #define HELP_CONTEXTMENU 0x000a
1659 #define HELP_FINDER      0x000b
1660 #define HELP_WM_HELP     0x000c
1661 #define HELP_SETPOPUP_POS 0x000d
1662
1663 #define HELP_TCARD        0x8000
1664 #define HELP_TCARD_DATA   0x0010
1665 #define HELP_TCARD_OTHER_CALLER 0x0011
1666
1667
1668     /* ChangeDisplaySettings return codes */
1669
1670 #define DISP_CHANGE_SUCCESSFUL 0
1671 #define DISP_CHANGE_RESTART   1
1672 #define DISP_CHANGE_FAILED    (-1)
1673 #define DISP_CHANGE_BADMODE   (-2)
1674 #define DISP_CHANGE_NOTUPDATED (-3)
1675 #define DISP_CHANGE_BADFLAGS  (-4)
1676 #define DISP_CHANGE_BADPARAM  (-5)
1677
1678 /* ChangeDisplaySettings.dwFlags */
1679 #define CDS_UPDATEREGISTRY 0x00000001
1680 #define CDS_TEST           0x00000002
1681 #define CDS_FULLSCREEN     0x00000004
1682 #define CDS_GLOBAL         0x00000008
1683 #define CDS_SET_PRIMARY    0x00000010
1684 #define CDS_RESET          0x40000000

```



```

1685 #define CDS_SETRECT      0x20000000
1686 #define CDS_NORESET      0x10000000
1687
1688 typedef struct
1689 {
1690     UINT        cbSize;
1691     UINT        style;
1692     WNDPROC      lpfnWndProc;
1693     INT          cbClsExtra;
1694     INT          cbWndExtra;
1695     HINSTANCE     hInstance;
1696     HICON         hIcon;
1697     HCURSOR       hCursor;
1698     HBRUSH        hbrBackground;
1699     LPCSTR        lpszMenuName;
1700     LPCSTR        lpszClassName;
1701     HICON         hIconSm;
1702 } WNDCLASSEX, *PWNDCLASSEX, *LPWNDCLASSEX;
1703
1704 typedef struct
1705 {
1706     UINT        cbSize;
1707     UINT        style;
1708     WNDPROC      lpfnWndProc;
1709     INT          cbClsExtra;
1710     INT          cbWndExtra;
1711     HINSTANCE     hInstance;
1712     HICON         hIcon;
1713     HCURSOR       hCursor;
1714     HBRUSH        hbrBackground;
1715     LPCWSTR       lpszMenuName;
1716     LPCWSTR       lpszClassName;
1717     HICON         hIconSm;
1718 } WNDCLASSEXW, *PWNDCLASSEXW, *LPWNDCLASSEXW;
1719
1720 DECL_WINELIB_TYPE_AW(WNDCLASSEX)
1721 DECL_WINELIB_TYPE_AW(PWNDCLASSEX)
1722 DECL_WINELIB_TYPE_AW(LPWNDCLASSEX)
1723
1724 typedef struct tagMSG
1725 {
1726     HWND        hwnd;
1727     UINT        message;
1728     WPARAM      wParam;
1729     LPARAM      lParam;
1730     DWORD        time;
1731     POINT        pt;
1732 } MSG, *PMSG, *LPMSG;
1733
1734 #define POINTSTOPOINT(pt, pts) \
1735 { (pt).x = (LONG) (SHORT) LOWORD(*(LONG*)&pts); \
1736   (pt).y = (LONG) (SHORT) HIWORD(*(LONG*)&pts); }
1737
1738 #define POINTTOPOINTS(pt)      (MAKELONG((short)((pt).x), (short)((pt).y)))
1739
1740
1741 /* Cursors / Icons */
1742
1743 typedef struct {
1744     BOOL        fIcon;
1745     DWORD        xHotspot;
1746     DWORD        yHotspot;
1747     HBITMAP      hbmMask;
1748     HBITMAP      hbmColor;
1749 } ICONINFO, *PICONINFO;
1750
1751
1752 /* this is the 6 byte accel struct used in Win32 when presented to the user */
1753 typedef struct
1754 {
1755     BYTE        fVirt;
1756     BYTE        pad0;
1757     WORD        key;
1758     WORD        cmd;
1759 } ACCEL, *LPACCEL;
1760
1761
1762 /* Flags for TrackPopupMenu */
1763 #define TPM_LEFTBUTTON      0x0000
1764 #define TPM_RIGHTBUTTON     0x0002
1765 #define TPM_LEFTALIGN       0x0000
1766 #define TPM_CENTERALIGN     0x0004
1767 #define TPM_RIGHTALIGN      0x0008
1768 #define TPM_TOPALIGN        0x0000
1769 #define TPM_VCENTERALIGN    0x0010
1770 #define TPM_BOTTOMALIGN     0x0020
1771 #define TPM_HORIZONTAL      0x0000

```

```

1772 #define TPM_VERTICAL          0x0040
1773 #define TPM_NONOTIFY          0x0080
1774 #define TPM_RETURNCMD         0x0100
1775
1776 typedef struct
1777 {
1778     UINT    cbSize;
1779     RECT    rcExclude;
1780 } TPM_PARAMS, *LTPM_PARAMS;
1781
1782 /* FIXME: not sure this one is correct */
1783 typedef struct {
1784     UINT    cbSize;
1785     UINT    fMask;
1786     UINT    fType;
1787     UINT    fState;
1788     UINT    wID;
1789     HMENU    hSubMenu;
1790     HBITMAP  hbmpChecked;
1791     HBITMAP  hbmpUnchecked;
1792     DWORD    dwItemData;
1793     LPSTR    dwTypeData;
1794     UINT    cch;
1795     HBITMAP  hbmpItem;
1796 } MENUITEMINFOA, *LPMENUITEMINFOA;
1797
1798 typedef struct {
1799     UINT    cbSize;
1800     UINT    fMask;
1801     UINT    fType;
1802     UINT    fState;
1803     UINT    wID;
1804     HMENU    hSubMenu;
1805     HBITMAP  hbmpChecked;
1806     HBITMAP  hbmpUnchecked;
1807     DWORD    dwItemData;
1808     LPWSTR    dwTypeData;
1809     UINT    cch;
1810     HBITMAP  hbmpItem;
1811 } MENUITEMINFOW, *LPMENUITEMINFOW;
1812
1813 DECL_WINELIB_TYPE_AW(MENUITEMINFO)
1814 DECL_WINELIB_TYPE_AW(LPMENUITEMINFO)
1815 typedef const MENUITEMINFOA *LPCMENUITEMINFOA;
1816 typedef const MENUITEMINFOW *LPCMENUITEMINFOW;
1817 DECL_WINELIB_TYPE_AW(LPCMENUITEMINFO)
1818
1819 typedef struct {
1820     DWORD    cbSize;
1821     DWORD    fMask;
1822     DWORD    dwStyle;
1823     UINT    cyMax;
1824     HBRUSH    hbrBack;
1825     DWORD    dwContextHelpID;
1826     DWORD    dwMenuData;
1827 } MENUINFO, *LPMENUINFO;
1828
1829 typedef const MENUINFO *LPCMENUINFO;
1830
1831 #define MIM_MAXHEIGHT          0x00000001
1832 #define MIM_BACKGROUND         0x00000002
1833 #define MIM_HELPID             0x00000004
1834 #define MIM_MENUDATA           0x00000008
1835 #define MIM_STYLE               0x00000010
1836 #define MIM_APPLYTOSUBMENUS    0x80000000
1837
1838 typedef struct {
1839     WORD    versionNumber;
1840     WORD    offset;
1841 } MENUITEMTEMPLATEHEADER, *PMENUITEMTEMPLATEHEADER;
1842
1843
1844 typedef struct {
1845     WORD    mtOption;
1846     WORD    mtID;
1847     WCHAR    mtString[1];
1848 } MENUITEMTEMPLATE, *PMENUITEMTEMPLATE;
1849
1850
1851 typedef VOID    MENUITEMTEMPLATE;
1852 typedef PVOID    *LPMENUITEMTEMPLATE;
1853
1854 /* Field specifiers for MENUITEMINFO[A]W type.    */
1855 #define MIIM_STATE             0x00000001
1856 #define MIIM_ID                0x00000002
1857 #define MIIM_SUBMENU           0x00000004
1858 #define MIIM_CHECKMARKS       0x00000008

```

```
1859 #define MIIM_TYPE          0x00000010
1860 #define MIIM_DATA          0x00000020
1861 #define MIIM_STRING        0x00000040
1862 #define MIIM_BITMAP        0x00000080
1863 #define MIIM_FTYPE         0x00000100
1864
1865 #define HBMMENU_CALLBACK    ((HBITMAP) -1)
1866 #define HBMMENU_SYSTEM     ((HBITMAP) 1)
1867 #define HBMMENU_MBAR_RESTORE ((HBITMAP) 2)
1868 #define HBMMENU_MBAR_MINIMIZE ((HBITMAP) 3)
1869 #define HBMMENU_MBAR_CLOSE ((HBITMAP) 5)
1870 #define HBMMENU_MBAR_CLOSE_D ((HBITMAP) 6)
1871 #define HBMMENU_MBAR_MINIMIZE_D ((HBITMAP) 7)
1872 #define HBMMENU_POPUP_CLOSE ((HBITMAP) 8)
1873 #define HBMMENU_POPUP_RESTORE ((HBITMAP) 9)
1874 #define HBMMENU_POPUP_MAXIMIZE ((HBITMAP) 10)
1875 #define HBMMENU_POPUP_MINIMIZE ((HBITMAP) 11)
1876
1877 /* WM_H/VSCROLL commands */
1878 #define SB_LINEUP          0
1879 #define SB_LINELEFT        0
1880 #define SB_LINEDOWN        1
1881 #define SB_LINERIGHT       1
1882 #define SB_PAGEUP          2
1883 #define SB_PAGELEFT        2
1884 #define SB_PAGEDOWN        3
1885 #define SB_PAGERIGHT       3
1886 #define SB_THUMBPOSITION    4
1887 #define SB_THUMBTRACK       5
1888 #define SB_TOP              6
1889 #define SB_LEFT             6
1890 #define SB_BOTTOM           7
1891 #define SB_RIGHT            7
1892 #define SB_ENDSCROLL       8
1893
1894 /* Scroll bar selection constants */
1895 #define SB_HORZ             0
1896 #define SB_VERT             1
1897 #define SB_CTL              2
1898 #define SB_BOTH             3
1899
1900 /* Scrollbar styles */
1901 #define SBS_HORZ            0x0000L
1902 #define SBS_VERT            0x0001L
1903 #define SBS_TOPALIGN        0x0002L
1904 #define SBS_LEFTALIGN       0x0002L
1905 #define SBS_BOTTOMALIGN     0x0004L
1906 #define SBS_RIGHTALIGN      0x0004L
1907 #define SBS_SIZEBOXTOPLEFTALIGN 0x0002L
1908 #define SBS_SIZEBOXBOTTOMRIGHTALIGN 0x0004L
1909 #define SBS_SIZEBOX         0x0008L
1910 #define SBS_SIZEGRIP        0x0010L
1911
1912 /* EnableScrollBar() flags */
1913 #define ESB_ENABLE_BOTH     0x0000
1914 #define ESB_DISABLE_BOTH    0x0003
1915
1916 #define ESB_DISABLE_LEFT    0x0001
1917 #define ESB_DISABLE_RIGHT   0x0002
1918
1919 #define ESB_DISABLE_UP      0x0001
1920 #define ESB_DISABLE_DOWN    0x0002
1921
1922 #define ESB_DISABLE_LTUP    ESB_DISABLE_LEFT
1923 #define ESB_DISABLE_RTDN    ESB_DISABLE_RIGHT
1924
1925 /* Win32 button control messages */
1926 #define BM_GETCHECK         0x00f0
1927 #define BM_SETCHECK         0x00f1
1928 #define BM_GETSTATE         0x00f2
1929 #define BM_SETSTATE         0x00f3
1930 #define BM_SETSTYLE         0x00f4
1931 #define BM_CLICK            0x00f5
1932 #define BM_GETIMAGE         0x00f6
1933 #define BM_SETIMAGE         0x00f7
1934 /* Winelib button control messages */
1935
1936 /* Button notification codes */
1937 #define BN_CLICKED          0
1938 #define BN_PAINT            1
1939 #define BN_HILITE           2
1940 #define BN_UNHILITE        3
1941 #define BN_DISABLE          4
1942 #define BN_DOUBLECLICKED    5
1943 #define BN_DBLCLK           BN_DOUBLECLICKED
1944
1945 /* Button states */
```

```

1946 #define BST_UNCHECKED          0x0000
1947 #define BST_CHECKED             0x0001
1948 #define BST_INDETERMINATE       0x0002
1949 #define BST_PUSHED              0x0004
1950 #define BST_FOCUS               0x0008
1951
1952 /* Static Control Styles */
1953 #define SS_LEFT                 0x00000000L
1954 #define SS_CENTER               0x00000001L
1955 #define SS_RIGHT                0x00000002L
1956 #define SS_ICON                 0x00000003L
1957 #define SS_BLACKRECT            0x00000004L
1958 #define SS_GRAYRECT            0x00000005L
1959 #define SS_WHITERECT           0x00000006L
1960 #define SS_BLACKFRAME           0x00000007L
1961 #define SS_GRAYFRAME            0x00000008L
1962 #define SS_WHITEFRAME           0x00000009L
1963
1964 #define SS_SIMPLE                0x0000000BL
1965 #define SS_LEFTNOWORDWRAP       0x0000000CL
1966
1967 #define SS_OWNERDRAW             0x0000000DL
1968 #define SS_BITMAP               0x0000000EL
1969 #define SS_ENHMETAFILE          0x0000000FL
1970
1971 #define SS_ETCHEDHORZ           0x00000010L
1972 #define SS_ETCHEDVERT           0x00000011L
1973 #define SS_ETCHEDFRAME          0x00000012L
1974 #define SS_TYPEMASK             0x0000001FL
1975
1976 #define SS_NOPREFIX              0x00000080L
1977 #define SS_NOTIFY               0x00000100L
1978 #define SS_CENTERIMAGE          0x00000200L
1979 #define SS_RIGHTJUST            0x00000400L
1980 #define SS_REALSIZEIMAGE        0x00000800L
1981 #define SS_SUNKEN               0x00001000L
1982
1983 /* Static Control Messages */
1984 #define STM_SETICON              0x0170
1985 #define STM_GETICON              0x0171
1986 #define STM_SETIMAGE             0x0172
1987 #define STM_GETIMAGE             0x0173
1988 #define STM_MSGMAX               0x0174
1989
1990 #define STN_CLICKED              0
1991 #define STN_DBLCLK               1
1992 #define STN_ENABLE               2
1993 #define STN_DISABLE              3
1994
1995 /* Scrollbar messages */
1996 #define SBM_SETPOS                0x00e0
1997 #define SBM_GETPOS                0x00e1
1998 #define SBM_SETRANGE              0x00e2
1999 #define SBM_GETRANGE              0x00e3
2000 #define SBM_ENABLE_ARROWS        0x00e4
2001 #define SBM_SETRANGEREDRAW        0x00e6
2002 #define SBM_SETSCROLLINFO         0x00e9
2003 #define SBM_GETSCROLLINFO         0x00ea
2004
2005 /* Scrollbar info */
2006 typedef struct
2007 {
2008     UINT    cbSize;
2009     UINT    fMask;
2010     INT     nMin;
2011     INT     nMax;
2012     UINT    nPage;
2013     INT     nPos;
2014     INT     nTrackPos;
2015 } SCROLLINFO, *LPSCROLLINFO;
2016
2017 typedef const SCROLLINFO *LPCSCROLLINFO;
2018
2019 /* GetScrollInfo() flags */
2020 #define SIF_RANGE                 0x0001
2021 #define SIF_PAGE                  0x0002
2022 #define SIF_POS                   0x0004
2023 #define SIF_DISABLENOSCROLL       0x0008
2024 #define SIF_TRACKPOS              0x0010
2025 #define SIF_ALL                   (SIF_RANGE | SIF_PAGE | SIF_POS | SIF_TRACKPOS)
2026
2027 /* Listbox styles */
2028 #define LBS_NOTIFY                 0x0001
2029 #define LBS_SORT                  0x0002
2030 #define LBS_NOREDRA              0x0004
2031 #define LBS_MULTIPLESEL           0x0008
2032 #define LBS_OWNERDRAWFIXED        0x0010

```

```

2033 #define LBS_OWNERDRAWVARIABLE    0x0020
2034 #define LBS_HASSTRINGS           0x0040
2035 #define LBS_USETABSTOPS          0x0080
2036 #define LBS_NOINTEGRALHEIGHT     0x0100
2037 #define LBS_MULTICOLUMN          0x0200
2038 #define LBS_WANTKEYBOARDINPUT    0x0400
2039 #define LBS_EXTENDEDSEL          0x0800
2040 #define LBS_DISABLENOSCROLL      0x1000
2041 #define LBS_NODATA               0x2000
2042 #define LBS_NOSEL                0x4000
2043 #define LBS_STANDARD (LBS_NOTIFY | LBS_SORT | WS_VSCROLL | WS_BORDER)
2044
2045 /* Listbox messages */
2046 #define LB_ADDSTRING              0x0180
2047 #define LB_INSERTSTRING           0x0181
2048 #define LB_DELETESTRING          0x0182
2049 #define LB_SELITEMRANGEEX        0x0183
2050 #define LB_RESETCONTENT          0x0184
2051 #define LB_SETSEL                0x0185
2052 #define LB_SETCURSEL             0x0186
2053 #define LB_GETSEL                0x0187
2054 #define LB_GETCURSEL             0x0188
2055 #define LB_GETTEXT               0x0189
2056 #define LB_GETTEXTLEN            0x018a
2057 #define LB_GETCOUNT            0x018b
2058 #define LB_SELECTSTRING          0x018c
2059 #define LB_DIR                   0x018d
2060 #define LB_GETTOPINDEX           0x018e
2061 #define LB_FINDSTRING            0x018f
2062 #define LB_GETSELCOUNT           0x0190
2063 #define LB_GETSELITEMS           0x0191
2064 #define LB_SETTABSTOPS           0x0192
2065 #define LB_GETHORIZONTALEXTENT   0x0193
2066 #define LB_SETHORIZONTALEXTENT   0x0194
2067 #define LB_SETCOLUMNWIDTH        0x0195
2068 #define LB_ADDFILE               0x0196
2069 #define LB_SETTOPINDEX           0x0197
2070 #define LB_GETITEMRECT           0x0198
2071 #define LB_GETITEMDATA           0x0199
2072 #define LB_SETITEMDATA           0x019a
2073 #define LB_SELITEMRANGE          0x019b
2074 #define LB_SETANCHORINDEX        0x019c
2075 #define LB_GETANCHORINDEX        0x019d
2076 #define LB_SETCARETINDEX         0x019e
2077 #define LB_GETCARETINDEX         0x019f
2078 #define LB_SETITEMHEIGHT         0x01a0
2079 #define LB_GETITEMHEIGHT         0x01a1
2080 #define LB_FINDSTRINGEXACT       0x01a2
2081 #define LB_CARETON               0x01a3
2082 #define LB_CARETOFF              0x01a4
2083 #define LB_SETLOCALE             0x01a5
2084 #define LB_GETLOCALE             0x01a6
2085 #define LB_SETCOUNT              0x01a7
2086 #define LB_INITSTORAGE           0x01a8
2087 #define LB_ITEMFROMPOINT         0x01a9
2088
2089 /* Listbox notification codes */
2090 #define LBN_ERRSPACE             (-2)
2091 #define LBN_SELCHANGE            1
2092 #define LBN_DBLCLK               2
2093 #define LBN_SELCANCEL            3
2094 #define LBN_SETFOCUS            4
2095 #define LBN_KILLFOCUS           5
2096
2097 /* Listbox message return values */
2098 #define LB_OKAY                  0
2099 #define LB_ERR                   (-1)
2100 #define LB_ERRSPACE              (-2)
2101
2102 #define LB_CTLCODE               0L
2103
2104 /* Combo box styles */
2105 #define CBS_SIMPLE                0x0001L
2106 #define CBS_DROPDOWN             0x0002L
2107 #define CBS_DROPDOWNLIST         0x0003L
2108 #define CBS_OWNERDRAWFIXED       0x0010L
2109 #define CBS_OWNERDRAWVARIABLE    0x0020L
2110 #define CBS_AUTOHSCROLL          0x0040L
2111 #define CBS_OEMCONVERT           0x0080L
2112 #define CBS_SORT                 0x0100L
2113 #define CBS_HASSTRINGS           0x0200L
2114 #define CBS_NOINTEGRALHEIGHT     0x0400L
2115 #define CBS_DISABLENOSCROLL      0x0800L
2116
2117 #define CBS_UPPERCASE            0x2000L
2118 #define CBS_LOWERCASE            0x4000L
2119

```

```

2120
2121 /* Combo box messages */
2122 #define CB_GETEDITSEL          0x0140
2123 #define CB_LIMITTEXT          0x0141
2124 #define CB_SETEXTEDITSEL      0x0142
2125 #define CB_ADDSTRING          0x0143
2126 #define CB_DELETESTRING      0x0144
2127 #define CB_DIR                0x0145
2128 #define CB_GETCOUNT         0x0146
2129 #define CB_GETCURSEL          0x0147
2130 #define CB_GETLBTEXT          0x0148
2131 #define CB_GETLBTEXTLEN      0x0149
2132 #define CB_INSERTSTRING       0x014a
2133 #define CB_RESETCONTENT       0x014b
2134 #define CB_FINDSTRING         0x014c
2135 #define CB_SELECTSTRING       0x014d
2136 #define CB_SETCURSEL          0x014e
2137 #define CB_SHOWDROPDOWN      0x014f
2138 #define CB_GETITEMDATA        0x0150
2139 #define CB_SETITEMDATA        0x0151
2140 #define CB_GETDROPPEDCONTROLRECT 0x0152
2141 #define CB_SETITEMHEIGHT      0x0153
2142 #define CB_GETITEMHEIGHT      0x0154
2143 #define CB_SETEXTENDEDUI      0x0155
2144 #define CB_GETEXTENDEDUI      0x0156
2145 #define CB_GETDROPPEDSTATE    0x0157
2146 #define CB_FINDSTRINGEXACT    0x0158
2147 #define CB_SETLOCALE          0x0159
2148 #define CB_GETLOCALE          0x015a
2149 #define CB_GETTOPINDEX        0x015b
2150 #define CB_SETTOPINDEX        0x015c
2151 #define CB_GETHORIZONTALEXTENT 0x015d
2152 #define CB_SETHORIZONTALEXTENT 0x015e
2153 #define CB_GETDROPPEDWIDTH    0x015f
2154 #define CB_SETDROPPEDWIDTH    0x0160
2155 #define CB_INITSTORAGE        0x0161
2156
2157 /* Combo box notification codes */
2158 #define CBN_ERRSPACE          (-1)
2159 #define CBN_SELCHANGE         1
2160 #define CBN_DBLCLK            2
2161 #define CBN_SETFOCUS          3
2162 #define CBN_KILLFOCUS         4
2163 #define CBN_EDITCHANGE        5
2164 #define CBN_EDITUPDATE        6
2165 #define CBN_DROPDOWN          7
2166 #define CBN_CLOSEUP           8
2167 #define CBN_SELENDOK          9
2168 #define CBN_SELENCANCEL       10
2169
2170 /* Combo box message return values */
2171 #define CB_OKAY                0
2172 #define CB_ERR                 (-1)
2173 #define CB_ERRSPACE            (-2)
2174
2175 #define MB_OK                  0x00000000
2176 #define MB_OKCANCEL           0x00000001
2177 #define MB_ABORTRETRYIGNORE    0x00000002
2178 #define MB_YESNOCANCEL         0x00000003
2179 #define MB_YESNO               0x00000004
2180 #define MB_RETRYCANCEL         0x00000005
2181 #define MB_TYPEMASK           0x0000000F
2182
2183 #define MB_ICONHAND            0x00000010
2184 #define MB_ICONQUESTION        0x00000020
2185 #define MB_ICONEXCLAMATION     0x00000030
2186 #define MB_ICONASTERISK        0x00000040
2187 #define MB_USERICON            0x00000080
2188 #define MB_ICONMASK            0x000000F0
2189
2190 #define MB_ICONINFORMATION      MB_ICONASTERISK
2191 #define MB_ICONSTOP            MB_ICONHAND
2192 #define MB_ICONWARNING          MB_ICONEXCLAMATION
2193 #define MB_ICONERROR           MB_ICONHAND
2194
2195 #define MB_DEFBUTTON1          0x00000000
2196 #define MB_DEFBUTTON2          0x00000100
2197 #define MB_DEFBUTTON3          0x00000200
2198 #define MB_DEFBUTTON4          0x00000300
2199 #define MB_DEFMASK             0x00000F00
2200
2201 #define MB_APPLMODAL            0x00000000
2202 #define MB_SYSTEMMODAL         0x00001000
2203 #define MB_TASKMODAL           0x00002000
2204 #define MB_MODEMASK            0x00003000
2205
2206 #define MB_HELP                0x00004000

```

```

2207 #define MB_NOFOCUS          0x00008000
2208 #define MB_MISCMASK         0x0000C000
2209
2210 #define MB_SETFOREGROUND     0x00010000
2211 #define MB_DEFAULT_DESKTOP_ONLY 0x00020000
2212 #define MB_SERVICE_NOTIFICATION 0x00040000
2213 #define MB_TOPMOST          0x00040000
2214 #define MB_RIGHT            0x00080000
2215 #define MB_RTLREADING       0x00100000
2216
2217 #define HELPINFO_WINDOW     0x0001
2218 #define HELPINFO_MENUITEM   0x0002
2219
2220 /* Structure pointed to by lParam of WM_HELP */
2221 typedef struct
2222 {
2223     UINT    cbSize;          /* Size in bytes of this struct */
2224     INT     iContextType;    /* Either HELPINFO_WINDOW or HELPINFO_MENUITEM */
2225     INT     iCtrlId;         /* Control Id or a Menu item Id. */
2226     HANDLE  hItemHandle;     /* hWnd of control or hMenu. */
2227     DWORD   dwContextId;     /* Context Id associated with this item */
2228     POINT   MousePos;        /* Mouse Position in screen co-ordinates */
2229 } HELPINFO, *LPHELPINFO;
2230
2231 typedef void CALLBACK (*MSGBOXCALLBACK) (LPHELPINFO lpHelpInfo);
2232
2233 typedef struct
2234 {
2235     UINT    cbSize;
2236     HWND    hwndOwner;
2237     HINSTANCE hInstance;
2238     LPCSTR   lpszText;
2239     LPCSTR   lpszCaption;
2240     DWORD    dwStyle;
2241     LPCSTR   lpszIcon;
2242     DWORD    dwContextHelpId;
2243     MSGBOXCALLBACK lpfnMsgBoxCallback;
2244     DWORD    dwLanguageId;
2245 } MSGBOXPARAMSA, *PMSGBOXPARAMSA, *LPMSGBOXPARAMSA;
2246
2247 typedef struct
2248 {
2249     UINT    cbSize;
2250     HWND    hwndOwner;
2251     HINSTANCE hInstance;
2252     LPCWSTR  lpszText;
2253     LPCWSTR  lpszCaption;
2254     DWORD    dwStyle;
2255     LPCWSTR  lpszIcon;
2256     DWORD    dwContextHelpId;
2257     MSGBOXCALLBACK lpfnMsgBoxCallback;
2258     DWORD    dwLanguageId;
2259 } MSGBOXPARAMSW, *PMSGBOXPARAMSW, *LPMSGBOXPARAMSW;
2260
2261 DECL_WINELIB_TYPE_AW (MSGBOXPARAMS)
2262 DECL_WINELIB_TYPE_AW (PMSGBOXPARAMS)
2263 DECL_WINELIB_TYPE_AW (LPMSGBOXPARAMS)
2264
2265 #define MONITOR_DEFAULTTONULL          0x00000000
2266 #define MONITOR_DEFAULTTOPRIMARY      0x00000001
2267 #define MONITOR_DEFAULTTONEAREST     0x00000002
2268
2269 #define MONITORINFOF_PRIMARY          0x00000001
2270
2271 #ifndef CCHDEVICENAME
2272 #define CCHDEVICENAME 32
2273 #endif
2274
2275 typedef struct tagMONITORINFO
2276 {
2277     DWORD    cbSize;
2278     RECT     rcMonitor;
2279     RECT     rcWork;
2280     DWORD    dwFlags;
2281 } MONITORINFO, *LPMONITORINFO;
2282
2283 typedef struct tagMONITORINFOEXA
2284 {
2285     MONITORINFO dummy;
2286     CHAR        szDevice[CCHDEVICENAME];
2287 } MONITORINFOEXA, *LPMONITORINFOEXA;
2288
2289 typedef struct tagMONITORINFOEXW
2290 {
2291     MONITORINFO dummy;
2292     WCHAR       szDevice[CCHDEVICENAME];
2293 } MONITORINFOEXW, *LPMONITORINFOEXW;

```

```

2294
2295 DECL_WINELIB_TYPE_AW(MONITORINFOEX)
2296 DECL_WINELIB_TYPE_AW(LPMONITORINFOEX)
2297
2298 typedef BOOL CALLBACK (*MONITORENUMPROC) (HMONITOR, HDC, LPRECT, LPARAM);
2299
2300 #include "pshpack2.h"
2301
2302 /* FIXME: use this instead of LPCVOID for CreateDialogIndirectParam
2303 and DialogBoxIndirectParam */
2304 typedef struct tagDLGTEMPLATE
2305 {
2306     DWORD style;
2307     DWORD dwExtendedStyle;
2308     WORD cedit;
2309     short x;
2310     short y;
2311     short cx;
2312     short cy;
2313 } DLGTEMPLATE;
2314
2315 typedef DLGTEMPLATE *LPDLGTEMPLATEA;
2316 typedef DLGTEMPLATE *LPDLGTEMPLATEW;
2317 DECL_WINELIB_TYPE_AW(LPDLGTEMPLATE)
2318 typedef const DLGTEMPLATE *LPCDLGTEMPLATEA;
2319 typedef const DLGTEMPLATE *LPCDLGTEMPLATEW;
2320 DECL_WINELIB_TYPE_AW(LPCDLGTEMPLATE)
2321
2322 typedef struct tagDLGITEMTEMPLATE
2323 {
2324     DWORD style;
2325     DWORD dwExtendedStyle;
2326     short x;
2327     short y;
2328     short cx;
2329     short cy;
2330     WORD id;
2331 } DLGITEMTEMPLATE;
2332
2333 typedef DLGITEMTEMPLATE *PDLGITEMTEMPLATEA;
2334 typedef DLGITEMTEMPLATE *PDLGITEMTEMPLATEW;
2335 DECL_WINELIB_TYPE_AW(PDLGITEMTEMPLATE)
2336 typedef DLGITEMTEMPLATE *LPDLGITEMTEMPLATEA;
2337 typedef DLGITEMTEMPLATE *LPDLGITEMTEMPLATEW;
2338 DECL_WINELIB_TYPE_AW(LPDLGITEMTEMPLATE)
2339
2340 #include "poppack.h"
2341
2342 /* CBT hook values */
2343 #define HCBT_MOVE_SIZE 0
2344 #define HCBT_MINMAX 1
2345 #define HCBT_QS 2
2346 #define HCBT_CREATEWND 3
2347 #define HCBT_DESTROYWND 4
2348 #define HCBT_ACTIVATE 5
2349 #define HCBT_CLICKSKIPPED 6
2350 #define HCBT_KEYSKIPPED 7
2351 #define HCBT_SYSCOMMAND 8
2352 #define HCBT_SETFOCUS 9
2353
2354 /* CBT hook structures */
2355
2356 typedef struct
2357 {
2358     CREATESTRUCTA *lpcs;
2359     HWND hwndInsertAfter;
2360 } CBT_CREATEWND, *LPCBT_CREATEWND;
2361
2362 typedef struct
2363 {
2364     CREATESTRUCTW *lpcs;
2365     HWND hwndInsertAfter;
2366 } CBT_CREATEWNDW, *LPCBT_CREATEWNDW;
2367
2368 DECL_WINELIB_TYPE_AW(CBT_CREATEWND)
2369 DECL_WINELIB_TYPE_AW(LPCBT_CREATEWND)
2370
2371 typedef struct
2372 {
2373     BOOL fMouse;
2374     HWND hwndActive;
2375 } CBT_ACTIVATESTRUCT, *LPCBT_ACTIVATESTRUCT;
2376
2377
2378 /* modifiers for RegisterHotKey */
2379 #define MOD_ALT 0x0001
2380 #define MOD_CONTROL 0x0002

```



```

2381 #define MOD_SHIFT    0x0004
2382 #define MOD_WIN      0x0008
2383
2384 /* ids for RegisterHotKey */
2385 #define IDHOT_SNAPWINDOW    (-1)    /* SHIFT-PRINTSCRN */
2386 #define IDHOT_SNAPDESKTOP  (-2)    /* PRINTSCRN */
2387
2388 /* keybd_event flags */
2389 #define KEYEVENTF_EXTENDEDKEY    0x0001
2390 #define KEYEVENTF_KEYUP          0x0002
2391
2392 /* mouse_event flags */
2393 #define MOUSEEVENTF_MOVE          0x0001
2394 #define MOUSEEVENTF_LEFTDOWN     0x0002
2395 #define MOUSEEVENTF_LEFTUP      0x0004
2396 #define MOUSEEVENTF_RIGHTDOWN    0x0008
2397 #define MOUSEEVENTF_RIGHTUP     0x0010
2398 #define MOUSEEVENTF_MIDDLEDOWN   0x0020
2399 #define MOUSEEVENTF_MIDDLEUP     0x0040
2400 #define MOUSEEVENTF_WHEEL        0x0800
2401 #define MOUSEEVENTF_ABSOLUTE     0x8000
2402
2403 /* ExitWindows() flags */
2404 #define EW_RESTARTWINDOWS    0x0042
2405 #define EW_REBOOTSYSYSTEM    0x0043
2406 #define EW_EXITANDEXECCAPP   0x0044
2407
2408 /* ExitWindowsEx() flags */
2409 #define EWX_LOGOFF          0
2410 #define EWX_SHUTDOWN        1
2411 #define EWX_REBOOT          2
2412 #define EWX_FORCE           4
2413 #define EWX_POWEROFF        8
2414
2415 /* SetLastErrorEx types */
2416 #define SLE_ERROR           0x00000001
2417 #define SLE_MINORERROR      0x00000002
2418 #define SLE_WARNING         0x00000003
2419
2420 /* Predefined resources */
2421 #define IDI_APPLICATIONA    MAKEINTRESOURCEA(32512)
2422 #define IDI_APPLICATIONW    MAKEINTRESOURCEW(32512)
2423 #define IDI_APPLICATION     WINELIB_NAME_AW(IDI_APPLICATION)
2424 #define IDI_HANDA           MAKEINTRESOURCEA(32513)
2425 #define IDI_HANDW           MAKEINTRESOURCEW(32513)
2426 #define IDI_HAND            WINELIB_NAME_AW(IDI_HAND)
2427 #define IDI_QUESTIONA       MAKEINTRESOURCEA(32514)
2428 #define IDI_QUESTIONW       MAKEINTRESOURCEW(32514)
2429 #define IDI_QUESTION        WINELIB_NAME_AW(IDI_QUESTION)
2430 #define IDI_EXCLAMATIONA    MAKEINTRESOURCEA(32515)
2431 #define IDI_EXCLAMATIONW    MAKEINTRESOURCEW(32515)
2432 #define IDI_EXCLAMATION     WINELIB_NAME_AW(IDI_EXCLAMATION)
2433 #define IDI_ASTERISKA       MAKEINTRESOURCEA(32516)
2434 #define IDI_ASTERISKW       MAKEINTRESOURCEW(32516)
2435 #define IDI_ASTERISK        WINELIB_NAME_AW(IDI_ASTERISK)
2436 #define IDI_WINLOGOA        MAKEINTRESOURCEA(32517)
2437 #define IDI_WINLOGOW        MAKEINTRESOURCEW(32517)
2438 #define IDI_WINLOGO         WINELIB_NAME_AW(IDI_WINLOGO)
2439
2440 #define IDI_WARNING          IDI_EXCLAMATION
2441 #define IDI_ERROR            IDI_HAND
2442 #define IDI_INFORMATION      IDI_ASTERISK
2443
2444 #define IDC_ARROWA           MAKEINTRESOURCEA(32512)
2445 #define IDC_ARROWW           MAKEINTRESOURCEW(32512)
2446 #define IDC_ARROW            WINELIB_NAME_AW(IDC_ARROW)
2447 #define IDC_IBEAMA           MAKEINTRESOURCEA(32513)
2448 #define IDC_IBEAMW           MAKEINTRESOURCEW(32513)
2449 #define IDC_IBEAM            WINELIB_NAME_AW(IDC_IBEAM)
2450 #define IDC_WAITA            MAKEINTRESOURCEA(32514)
2451 #define IDC_WAITW            MAKEINTRESOURCEW(32514)
2452 #define IDC_WAIT             WINELIB_NAME_AW(IDC_WAIT)
2453 #define IDC_CROSSA           MAKEINTRESOURCEA(32515)
2454 #define IDC_CROSSW           MAKEINTRESOURCEW(32515)
2455 #define IDC_CROSS            WINELIB_NAME_AW(IDC_CROSS)
2456 #define IDC_UPARROWA         MAKEINTRESOURCEA(32516)
2457 #define IDC_UPARROWW         MAKEINTRESOURCEW(32516)
2458 #define IDC_UPARROW          WINELIB_NAME_AW(IDC_UPARROW)
2459 #define IDC_SIZEA            MAKEINTRESOURCEA(32640)
2460 #define IDC_SIZEW            MAKEINTRESOURCEW(32640)
2461 #define IDC_SIZE             WINELIB_NAME_AW(IDC_SIZE)
2462 #define IDC_ICONA            MAKEINTRESOURCEA(32641)
2463 #define IDC_ICONW            MAKEINTRESOURCEW(32641)
2464 #define IDC_ICON             WINELIB_NAME_AW(IDC_ICON)
2465 #define IDC_SIZENWSEA        MAKEINTRESOURCEA(32642)
2466 #define IDC_SIZENWSEW        MAKEINTRESOURCEW(32642)
2467 #define IDC_SIZENWSE         WINELIB_NAME_AW(IDC_SIZENWSE)

```

```
2468 #define IDC_SIZENESWA MAKEINTRESOURCEA(32643)
2469 #define IDC_SIZENESWW MAKEINTRESOURCEW(32643)
2470 #define IDC_SIZENESW WINELIB_NAME_AW(IDC_SIZENESW)
2471 #define IDC_SIZEWEA MAKEINTRESOURCEA(32644)
2472 #define IDC_SIZEWEW MAKEINTRESOURCEW(32644)
2473 #define IDC_SIZEWE WINELIB_NAME_AW(IDC_SIZEWE)
2474 #define IDC_SIZENSA MAKEINTRESOURCEA(32645)
2475 #define IDC_SIZENSW MAKEINTRESOURCEW(32645)
2476 #define IDC_SIZENS WINELIB_NAME_AW(IDC_SIZENS)
2477 #define IDC_SIZEALLA MAKEINTRESOURCEA(32646)
2478 #define IDC_SIZEALLW MAKEINTRESOURCEW(32646)
2479 #define IDC_SIZEALL WINELIB_NAME_AW(IDC_SIZEALL)
2480 #define IDC_NOA MAKEINTRESOURCEA(32648)
2481 #define IDC_NOW MAKEINTRESOURCEW(32648)
2482 #define IDC_NO WINELIB_NAME_AW(IDC_NO)
2483 #define IDC_HANDA MAKEINTRESOURCEA(32649)
2484 #define IDC_HANDW MAKEINTRESOURCEW(32649)
2485 #define IDC_HAND WINELIB_NAME_AW(IDC_HAND)
2486 #define IDC_APPSTARTINGA MAKEINTRESOURCEA(32650)
2487 #define IDC_APPSTARTINGW MAKEINTRESOURCEW(32650)
2488 #define IDC_APPSTARTING WINELIB_NAME_AW(IDC_APPSTARTING)
2489 #define IDC_HELPA MAKEINTRESOURCEA(32651)
2490 #define IDC_HELPW MAKEINTRESOURCEW(32651)
2491 #define IDC_HELP WINELIB_NAME_AW(IDC_HELP)
2492
2493 #define MNC_IGNORE 0
2494 #define MNC_CLOSE 1
2495 #define MNC_EXECUTE 2
2496 #define MNC_SELECT 3
2497
2498 /* SystemParametersInfo */
2499 /* defines below are for all win versions */
2500 #define SPI_GETBEEP 1
2501 #define SPI_SETBEEP 2
2502 #define SPI_GETMOUSE 3
2503 #define SPI_SETMOUSE 4
2504 #define SPI_GETBORDER 5
2505 #define SPI_SETBORDER 6
2506 #define SPI_GETKEYBOARDSPEED 10
2507 #define SPI_SETKEYBOARDSPEED 11
2508 #define SPI_LANGDRIVER 12
2509 #define SPI_ICONHORIZONTALSPACING 13
2510 #define SPI_GETSCREENSAVETIMEOUT 14
2511 #define SPI_SETSCREENSAVETIMEOUT 15
2512 #define SPI_GETSCREENSAVEACTIVE 16
2513 #define SPI_SETSCREENSAVEACTIVE 17
2514 #define SPI_GETGRIDGRANULARITY 18
2515 #define SPI_SETGRIDGRANULARITY 19
2516 #define SPI_SETDESKWALLPAPER 20
2517 #define SPI_SETDESKPATTERN 21
2518 #define SPI_GETKEYBOARDDELAY 22
2519 #define SPI_SETKEYBOARDDELAY 23
2520 #define SPI_ICONVERTICALSPACING 24
2521 #define SPI_GETTICONTITLEWRAP 25
2522 #define SPI_SETTICONTITLEWRAP 26
2523 #define SPI_GETMENUDROPALIGNMENT 27
2524 #define SPI_SETMENUDROPALIGNMENT 28
2525 #define SPI_SETDOUBLECLKWIDTH 29
2526 #define SPI_SETDOUBLECLKHEIGHT 30
2527 #define SPI_GETTICONTITLELOGFONT 31
2528 #define SPI_SETDOUBLECLICKTIME 32
2529 #define SPI_SETMOUSEBUTTONSWAP 33
2530 #define SPI_SETTICONTITLELOGFONT 34
2531 #define SPI_GETFASTTASKSWITCH 35
2532 #define SPI_SETFASTTASKSWITCH 36
2533 #define SPI_SETDRAGFULLWINDOWS 37
2534 #define SPI_GETDRAGFULLWINDOWS 38
2535
2536 #define SPI_GETFILTERKEYS 50
2537 #define SPI_SETFILTERKEYS 51
2538 #define SPI_GETTOGGLEKEYS 52
2539 #define SPI_SETTOGGLEKEYS 53
2540 #define SPI_GETMOUSEKEYS 54
2541 #define SPI_SETMOUSEKEYS 55
2542 #define SPI_GETSHOWSOUNDS 56
2543 #define SPI_SETSHOWSOUNDS 57
2544 #define SPI_GETSTICKYKEYS 58
2545 #define SPI_SETSTICKYKEYS 59
2546 #define SPI_GETACCESSTIMEOUT 60
2547 #define SPI_SETACCESSTIMEOUT 61
2548
2549 #define SPI_GETSOUNDSENTRY 64
2550 #define SPI_SETSOUNDSENTRY 65
2551
2552 /* defines below are for all win versions WINVER >= 0x0400 */
2553 #define SPI_SETDRAGFULLWINDOWS 37
2554 #define SPI_GETDRAGFULLWINDOWS 38
```

```
2555 #define SPI_GETNONCLIENTMETRICS 41
2556 #define SPI_SETNONCLIENTMETRICS 42
2557 #define SPI_GETMINIMIZEDMETRICS 43
2558 #define SPI_SETMINIMIZEDMETRICS 44
2559 #define SPI_GETICONMETRICS 45
2560 #define SPI_SETICONMETRICS 46
2561 #define SPI_SETWORKAREA 47
2562 #define SPI_GETWORKAREA 48
2563 #define SPI_SETPENWINDOWS 49
2564
2565 #define SPI_GETSERIALKEYS 62
2566 #define SPI_SETSERIALKEYS 63
2567 #define SPI_GETHIGHCONTRAST 66
2568 #define SPI_SETHIGHCONTRAST 67
2569 #define SPI_GETKEYBOARDPREF 68
2570 #define SPI_SETKEYBOARDPREF 69
2571 #define SPI_GETSCREENREADER 70
2572 #define SPI_SETSCREENREADER 71
2573 #define SPI_GETANIMATION 72
2574 #define SPI_SETANIMATION 73
2575 #define SPI_GETFONTSMOOTHING 74
2576 #define SPI_SETFONTSMOOTHING 75
2577 #define SPI_SETDRAGWIDTH 76
2578 #define SPI_SETDRAGHEIGHT 77
2579 #define SPI_SETHANDHELD 78
2580 #define SPI_GETLOWPOWERTIMEOUT 79
2581 #define SPI_GETPOWEROFFTIMEOUT 80
2582 #define SPI_SETLOWPOWERTIMEOUT 81
2583 #define SPI_SETPOWEROFFTIMEOUT 82
2584 #define SPI_GETLOWPOWERACTIVE 83
2585 #define SPI_GETPOWEROFFACTIVE 84
2586 #define SPI_SETLOWPOWERACTIVE 85
2587 #define SPI_SETPOWEROFFACTIVE 86
2588 #define SPI_SETCURSORS 87
2589 #define SPI_SETICONS 88
2590 #define SPI_GETDEFAULTINPUTLANG 89
2591 #define SPI_SETDEFAULTINPUTLANG 90
2592 #define SPI_SETLANGTOGGLE 91
2593 #define SPI_GETWINDOWSEXTENSION 92
2594 #define SPI_SETMOUSETRAILS 93
2595 #define SPI_GETMOUSETRAILS 94
2596 #define SPI_SETSCREENSAVERRUNNING 97
2597 #define SPI_SCREENSAVERRUNNING SPI_SETSCREENSAVERRUNNING
2598
2599 /* defines below are for all win versions (_WIN32_WINNT >= 0x0400) ||
2600 * (_WIN32_WINDOWS > 0x0400) */
2601 #define SPI_GETMOUSEHOVERWIDTH 98
2602 #define SPI_SETMOUSEHOVERWIDTH 99
2603 #define SPI_GETMOUSEHOVERHEIGHT 100
2604 #define SPI_SETMOUSEHOVERHEIGHT 101
2605 #define SPI_GETMOUSEHOVERTIME 102
2606 #define SPI_SETMOUSEHOVERTIME 103
2607 #define SPI_GETWHEELSCROLLLINES 104
2608 #define SPI_SETWHEELSCROLLLINES 105
2609 #define SPI_GETMENUSHOWDELAY 106
2610 #define SPI_SETMENUSHOWDELAY 107
2611
2612 #define SPI_GETSHOWIMEUI 110
2613 #define SPI_SETSHOWIMEUI 111
2614
2615 /* defines below are for all win versions WINVER >= 0x0500 */
2616 #define SPI_GETMOUSESPEED 112
2617 #define SPI_SETMOUSESPEED 113
2618 #define SPI_GETSCREENSAVERRUNNING 114
2619 #define SPI_GETDESKWALLPAPER 115
2620
2621 #define SPI_GETACTIVEWINDOWTRACKING 0x1000
2622 #define SPI_SETACTIVEWINDOWTRACKING 0x1001
2623 #define SPI_GETMENUANIMATION 0x1002
2624 #define SPI_SETMENUANIMATION 0x1003
2625 #define SPI_GETCOMBOBOXANIMATION 0x1004
2626 #define SPI_SETCOMBOBOXANIMATION 0x1005
2627 #define SPI_GETLISTBOXSMOOTHSCROLLING 0x1006
2628 #define SPI_SETLISTBOXSMOOTHSCROLLING 0x1007
2629 #define SPI_GETGRADIENTCAPTIONS 0x1008
2630 #define SPI_SETGRADIENTCAPTIONS 0x1009
2631 #define SPI_GETMENUUNDERLINES 0x100A
2632 #define SPI_SETMENUUNDERLINES 0x100B
2633 #define SPI_GETACTIVEWNDTRKZORDER 0x100C
2634 #define SPI_SETACTIVEWNDTRKZORDER 0x100D
2635 #define SPI_GETHOTTRACKING 0x100E
2636 #define SPI_SETHOTTRACKING 0x100F
2637 #define SPI_GETFOREGROUNDLOCKTIMEOUT 0x2000
2638 #define SPI_SETFOREGROUNDLOCKTIMEOUT 0x2001
2639 #define SPI_GETACTIVEWNDTRKTIMEOUT 0x2002
2640 #define SPI_SETACTIVEWNDTRKTIMEOUT 0x2003
2641 #define SPI_GETFOREGROUNDFLASHCOUNT 0x2004
```

```

2642 #define SPI_SETFOREGROUNDFLASHCOUNT    0x2005
2643
2644 /* SystemParametersInfo flags */
2645
2646 #define SPIF_UPDATEINIFILE                1
2647 #define SPIF_SENDWININICHANGE            2
2648 #define SPIF_SENDCHANGE                   SPIF_SENDWININICHANGE
2649
2650 #if defined(_WINGDI_) && !defined(NOCDI)
2651 typedef struct {
2652     UINT        cbSize;
2653     INT         iBorderWidth;
2654     INT         iScrollWidth;
2655     INT         iScrollHeight;
2656     INT         iCaptionWidth;
2657     INT         iCaptionHeight;
2658     LOGFONTA    lfCaptionFont;
2659     INT         iSmCaptionWidth;
2660     INT         iSmCaptionHeight;
2661     LOGFONTA    lfSmCaptionFont;
2662     INT         iMenuWidth;
2663     INT         iMenuHeight;
2664     LOGFONTA    lfMenuFont;
2665     LOGFONTA    lfStatusFont;
2666     LOGFONTA    lfMessageFont;
2667 } NONCLIENTMETRICS, *PNONCLIENTMETRICS, *LPNONCLIENTMETRICS;
2668
2669 typedef struct {
2670     UINT        cbSize;
2671     INT         iBorderWidth;
2672     INT         iScrollWidth;
2673     INT         iScrollHeight;
2674     INT         iCaptionWidth;
2675     INT         iCaptionHeight;
2676     LOGFONTW    lfCaptionFont;
2677     INT         iSmCaptionWidth;
2678     INT         iSmCaptionHeight;
2679     LOGFONTW    lfSmCaptionFont;
2680     INT         iMenuWidth;
2681     INT         iMenuHeight;
2682     LOGFONTW    lfMenuFont;
2683     LOGFONTW    lfStatusFont;
2684     LOGFONTW    lfMessageFont;
2685 } NONCLIENTMETRICSW, *PNONCLIENTMETRICSW, *LPNONCLIENTMETRICSW;
2686
2687 DECL_WINELIB_TYPE_AW(NONCLIENTMETRICS)
2688 DECL_WINELIB_TYPE_AW(PNONCLIENTMETRICS)
2689 DECL_WINELIB_TYPE_AW(LPNONCLIENTMETRICS)
2690
2691 typedef struct tagICONMETRICS {
2692     UINT        cbSize;
2693     int         iHorzSpacing;
2694     int         iVertSpacing;
2695     int         iTitleWrap;
2696     LOGFONTA    lfFont;
2697 } ICONMETRICS, *PICONMETRICS, *LPICONMETRICS;
2698
2699 typedef struct tagICONMETRICSW {
2700     UINT        cbSize;
2701     int         iHorzSpacing;
2702     int         iVertSpacing;
2703     int         iTitleWrap;
2704     LOGFONTW    lfFont;
2705 } ICONMETRICSW, *PICONMETRICSW, *LPICONMETRICSW;
2706
2707 DECL_WINELIB_TYPE_AW(ICONMETRICS)
2708 DECL_WINELIB_TYPE_AW(PICONMETRICS)
2709 DECL_WINELIB_TYPE_AW(LPICONMETRICS)
2710 #endif /* defined(_WINGDI_) && !defined(NOCDI) */
2711
2712 #define ARW_BOTTOMLEFT                    0x0000L
2713 #define ARW_BOTTOMRIGHT                   0x0001L
2714 #define ARW_TOPLEFT                       0x0002L
2715 #define ARW_TOPRIGHT                      0x0003L
2716 #define ARW_STARTMASK                     0x0003L
2717 #define ARW_STARTRIGHT                    0x0001L
2718 #define ARW_STARTTOP                      0x0002L
2719
2720 #define ARW_LEFT                          0x0000L
2721 #define ARW_RIGHT                         0x0000L
2722 #define ARW_UP                            0x0004L
2723 #define ARW_DOWN                          0x0004L
2724 #define ARW_HIDE                          0x0008L
2725
2726 typedef struct tagMINIMIZEDMETRICS {
2727     UINT        cbSize;
2728     int         iWidth;

```

```

2729     int iHorzGap;
2730     int iVertGap;
2731     int iArrange;
2732 } MINIMIZEDMETRICS, *PMINIMIZEDMETRICS, *LPMINIMIZEDMETRICS;
2733
2734 /* Window Styles */
2735 #define WS_OVERLAPPED 0x00000000L
2736 #define WS_POPUP 0x80000000L
2737 #define WS_CHILD 0x40000000L
2738 #define WS_MINIMIZE 0x20000000L
2739 #define WS_VISIBLE 0x10000000L
2740 #define WS_DISABLED 0x08000000L
2741 #define WS_CLIPSIBLINGS 0x04000000L
2742 #define WS_CLIPCHILDREN 0x02000000L
2743 #define WS_MAXIMIZE 0x01000000L
2744 #define WS_CAPTION 0x00C00000L
2745 #define WS_BORDER 0x00800000L
2746 #define WS_DLGFRAME 0x00400000L
2747 #define WS_VSCROLL 0x00200000L
2748 #define WS_HSCROLL 0x00100000L
2749 #define WS_SYSMENU 0x00080000L
2750 #define WS_THICKFRAME 0x00040000L
2751 #define WS_GROUP 0x00020000L
2752 #define WS_TABSTOP 0x00010000L
2753 #define WS_MINIMIZEBOX 0x00020000L
2754 #define WS_MAXIMIZEBOX 0x00010000L
2755 #define WS_TILED WS_OVERLAPPED
2756 #define WS_ICONIC WS_MINIMIZE
2757 #define WS_SIZEBOX WS_THICKFRAME
2758 #define WS_OVERLAPPEDWINDOW (WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU | WS_THICKFRAME | WS_MINIMIZEBOX |
    WS_MAXIMIZEBOX)
2759 #define WS_POPUPWINDOW (WS_POPUP | WS_BORDER | WS_SYSMENU)
2760 #define WS_CHILDWINDOW (WS_CHILD)
2761 #define WS_TILEDWINDOW (WS_OVERLAPPEDWINDOW)
2762
2763 /* Window extended styles */
2764 #define WS_EX_DLGMODALFRAME 0x00000001L
2765 #define WS_EX_DRAGDETECT 0x00000002L
2766 #define WS_EX_NOPARENTNOTIFY 0x00000004L
2767 #define WS_EX_TOPMOST 0x00000008L
2768 #define WS_EX_ACCEPTFILES 0x00000010L
2769 #define WS_EX_TRANSPARENT 0x00000020L
2770
2771 /* New Win95/WinNT4 styles */
2772 #define WS_EX_MDICHILD 0x00000040L
2773 #define WS_EX_TOOLWINDOW 0x00000080L
2774 #define WS_EX_WINDOWEDGE 0x00000100L
2775 #define WS_EX_CLIENTEDGE 0x00000200L
2776 #define WS_EX_CONTEXTHELP 0x00000400L
2777 #define WS_EX_RIGHT 0x00001000L
2778 #define WS_EX_LEFT 0x00000000L
2779 #define WS_EX_RTLREADING 0x00002000L
2780 #define WS_EX_LTRREADING 0x00000000L
2781 #define WS_EX_LEFTSCROLLBAR 0x00004000L
2782 #define WS_EX_RIGHTSCROLLBAR 0x00000000L
2783 #define WS_EX_CONTROLPARENT 0x00010000L
2784 #define WS_EX_STATICEDGE 0x00020000L
2785 #define WS_EX_APPWINDOW 0x00040000L
2786
2787 #define WS_EX_OVERLAPPEDWINDOW (WS_EX_WINDOWEDGE | WS_EX_CLIENTEDGE)
2788 #define WS_EX_PALETTEWINDOW (WS_EX_WINDOWEDGE | WS_EX_TOOLWINDOW | WS_EX_TOPMOST)
2789
2790 /* New Win2000 styles */
2791 #define WS_EX_LAYERED 0x00080000L
2792
2793 /* WINE internal... */
2794 #define WS_EX_TRAYWINDOW 0x80000000L
2795 #define WS_EX_MANAGED 0x40000000L /* Window managed by the window system */
2796
2797 /* Window scrolling */
2798 #define SW_SCROLLCHILDREN 0x0001
2799 #define SW_INVALIDATE 0x0002
2800 #define SW_ERASE 0x0004
2801
2802 /* CreateWindow() coordinates */
2803 #define CW_USEDEFAULT ((INT)0x80000000)
2804
2805 /* ChildWindowFromPointEx Flags */
2806 #define CWP_ALL 0x0000
2807 #define CWP_SKIPINVISIBLE 0x0001
2808 #define CWP_SKIPDISABLED 0x0002
2809 #define CWP_SKIPTRANSPARENT 0x0004
2810
2811 /* PeekMessage() options */
2812 #define PM_NOREMOVE 0x0000
2813 #define PM_REMOVE 0x0001
2814 #define PM_NOYIELD 0x0002

```

```
2815
2816 /* AnimateWindow() flags */
2817 #define AW_SLIDE 0x00040000
2818 #define AW_ACTIVATE 0x00020000
2819 #define AW_BLEND 0x00080000
2820 #define AW_HIDE 0x00010000
2821 #define AW_CENTER 0x00000010
2822 #define AW_HOR_POSITIVE 0x00000001
2823 #define AW_HOR_NEGATIVE 0x00000002
2824 #define AW_VER_POSITIVE 0x00000004
2825 #define AW_VER_NEGATIVE 0x00000008
2826
2827 /* WM_SHOWWINDOW wParam codes */
2828 #define SW_PARENTCLOSING 1
2829 #define SW_OTHERMAXIMIZED 2
2830 #define SW_PARENTOPENING 3
2831 #define SW_OTHERRESTORED 4
2832
2833 /* ShowWindow() codes */
2834 #define SW_HIDE 0
2835 #define SW_SHOWNORMAL 1
2836 #define SW_NORMAL 1
2837 #define SW_SHOWMINIMIZED 2
2838 #define SW_SHOWMAXIMIZED 3
2839 #define SW_MAXIMIZE 3
2840 #define SW_SHOWNOACTIVATE 4
2841 #define SW_SHOW 5
2842 #define SW_MINIMIZE 6
2843 #define SW_SHOWMINNOACTIVE 7
2844 #define SW_SHOWNA 8
2845 #define SW_RESTORE 9
2846 #define SW_SHOWDEFAULT 10
2847 #define SW_MAX 10
2848 #define SW_NORMALNA 0xCC /* undoc. flag in MinMaximize */
2849
2850 /* WM_SIZE message wParam values */
2851 #define SIZE_RESTORED 0
2852 #define SIZE_MINIMIZED 1
2853 #define SIZE_MAXIMIZED 2
2854 #define SIZE_MAXSHOW 3
2855 #define SIZE_MAXHIDE 4
2856 #define SIZEENORMAL SIZE_RESTORED
2857 #define SIZEEICONIC SIZE_MINIMIZED
2858 #define SIZEFULLSCREEN SIZE_MAXIMIZED
2859 #define SIZEZOOMSHOW SIZE_MAXSHOW
2860 #define SIZEZOOMHIDE SIZE_MAXHIDE
2861
2862 /* SetWindowPos() and WINDOWPOS flags */
2863 #define SWP_NOSIZE 0x0001
2864 #define SWP_NOMOVE 0x0002
2865 #define SWP_NOZORDER 0x0004
2866 #define SWP_NOREDRAW 0x0008
2867 #define SWP_NOACTIVATE 0x0010
2868 #define SWP_FRAMECHANGED 0x0020 /* The frame changed: send WM_NCCALCSIZE */
2869 #define SWP_SHOWWINDOW 0x0040
2870 #define SWP_HIDEWINDOW 0x0080
2871 #define SWP_NOCOPYBITS 0x0100
2872 #define SWP_NOOWNERZORDER 0x0200 /* Don't do owner Z ordering */
2873
2874 #define SWP_DRAWFRAME SWP_FRAMECHANGED
2875 #define SWP_NOREPOSITION SWP_NOOWNERZORDER
2876
2877 #define SWP_NOSENDCHANGING 0x0400
2878 #define SWP_DEFERERASE 0x2000
2879 #define SWP_ASYNCWINDOWPOS 0x4000
2880
2881 #define HWND_DESKTOP ((HWND)0)
2882 #define HWND_BROADCAST ((HWND)0xffff)
2883
2884 /* SetWindowPos() hwndInsertAfter field values */
2885 #define HWND_TOP ((HWND)0)
2886 #define HWND_BOTTOM ((HWND)1)
2887 #define HWND_TOPMOST ((HWND)-1)
2888 #define HWND_NOTOPMOST ((HWND)-2)
2889 #define HWND_MESSAGE ((HWND)-3)
2890
2891 /* GetDCEX flags */
2892 #define DCX_WINDOW 0x00000001
2893 #define DCX_CACHE 0x00000002
2894 #define DCX_NORESETATTRS 0x00000004
2895 #define DCX_CLIPCHILDREN 0x00000008
2896 #define DCX_CLIPSIBLINGS 0x00000010
2897 #define DCX_PARENTCLIP 0x00000020
2898 #define DCX_EXCLUDERGN 0x00000040
2899 #define DCX_INTERSECTRGN 0x00000080
2900 #define DCX_EXCLUDEUPDATE 0x00000100
2901 #define DCX_INTERSECTUPDATE 0x00000200
```

```
2902 #define DCX_LOCKWINDOWUPDATE 0x00000400
2903 #define DCX_USESTYLE 0x00010000
2904 #define DCX_NORECOMPUTE 0x00100000
2905 #define DCX_VALIDATE 0x00200000
2906
2907 #define MF_INSERT 0x0000
2908 #define MF_CHANGE 0x0080
2909 #define MF_APPEND 0x0100
2910 #define MF_DELETE 0x0200
2911 #define MF_REMOVE 0x1000
2912 #define MF_END 0x0080
2913
2914 #define MF_ENABLED 0x0000
2915 #define MF_GRAYED 0x0001
2916 #define MF_DISABLED 0x0002
2917 #define MF_STRING 0x0000
2918 #define MF_BITMAP 0x0004
2919 #define MF_UNCHECKED 0x0000
2920 #define MF_CHECKED 0x0008
2921 #define MF_POPUP 0x0010
2922 #define MF_MENUBARBREAK 0x0020
2923 #define MF_MENUBREAK 0x0040
2924 #define MF_UNHILITE 0x0000
2925 #define MF_HILITE 0x0080
2926 #define MF_OWNERDRAW 0x0100
2927 #define MF_USECHECKBITMAPS 0x0200
2928 #define MF_BYCOMMAND 0x0000
2929 #define MF_BYPOSITION 0x0400
2930 #define MF_SEPARATOR 0x0800
2931 #define MF_DEFAULT 0x1000
2932 #define MF_SYSMENU 0x2000
2933 #define MF_HELP 0x4000
2934 #define MF_RIGHTJUSTIFY 0x4000
2935 #define MF_MOUSESELECT 0x8000
2936
2937 /* Flags for extended menu item types. */
2938 #define MFT_STRING MF_STRING
2939 #define MFT_BITMAP MF_BITMAP
2940 #define MFT_MENUBARBREAK MF_MENUBARBREAK
2941 #define MFT_MENUBREAK MF_MENUBREAK
2942 #define MFT_OWNERDRAW MF_OWNERDRAW
2943 #define MFT_RADIOCHECK 0x0000200L
2944 #define MFT_SEPARATOR MF_SEPARATOR
2945 #define MFT_RIGHTORDER 0x0000200L
2946 #define MFT_RIGHTJUSTIFY MF_RIGHTJUSTIFY
2947
2948 /* Flags for extended menu item states. */
2949 #define MFS_GRAYED 0x00000003L
2950 #define MFS_DISABLED MFS_GRAYED
2951 #define MFS_CHECKED MF_CHECKED
2952 #define MFS_HILITE MF_HILITE
2953 #define MFS_ENABLED MF_ENABLED
2954 #define MFS_UNCHECKED MF_UNCHECKED
2955 #define MFS_UNHILITE MF_UNHILITE
2956 #define MFS_DEFAULT MF_DEFAULT
2957 #define MFS_MASK 0x0000108BL
2958 #define MFS_HOTTRACKDRAWN 0x10000000L
2959 #define MFS_CACHEDBMP 0x20000000L
2960 #define MFS_BOTTOMGAPDROP 0x40000000L
2961 #define MFS_TOPGAPDROP 0x80000000L
2962 #define MFS_GAPDROP 0xC0000000L
2963
2964 /* for GetMenuDefaultItem */
2965 #define GMDI_USEDISABLED 0x0001L
2966 #define GMDI_GOINTOPOPUPS 0x0002L
2967
2968 #define DT_TOP 0
2969 #define DT_LEFT 0
2970 #define DT_CENTER 1
2971 #define DT_RIGHT 2
2972 #define DT_VCENTER 4
2973 #define DT_BOTTOM 8
2974 #define DT_WORDBREAK 16
2975 #define DT_SINGLELINE 32
2976 #define DT_EXPANDTABS 64
2977 #define DT_TABSTOP 128
2978 #define DT_NOCLIP 256
2979 #define DT_EXTERNALLEADING 512
2980 #define DT_CALCRECT 1024
2981 #define DT_NOPREFIX 2048
2982 #define DT_INTERNAL 4096
2983
2984 /* DrawCaption()/DrawCaptionTemp() flags */
2985 #define DC_ACTIVE 0x0001
2986 #define DC_SMALLCAP 0x0002
2987 #define DC_ICON 0x0004
2988 #define DC_TEXT 0x0008
```



```

2989 #define DC_INBUTTON      0x0010
2990
2991 /* DrawEdge() flags */
2992 #define BDR_RAISEDOUTER    0x0001
2993 #define BDR_SUNKENOUTER    0x0002
2994 #define BDR_RAISEDINNER    0x0004
2995 #define BDR_SUNKENINNER    0x0008
2996
2997 #define BDR_OUTER          0x0003
2998 #define BDR_INNER          0x000c
2999 #define BDR_RAISED         0x0005
3000 #define BDR_SUNKEN         0x000a
3001
3002 #define EDGE_RAISED        (BDR_RAISEDOUTER | BDR_RAISEDINNER)
3003 #define EDGE_SUNKEN        (BDR_SUNKENOUTER | BDR_SUNKENINNER)
3004 #define EDGE_ETCHED        (BDR_SUNKENOUTER | BDR_RAISEDINNER)
3005 #define EDGE_BUMP          (BDR_RAISEDOUTER | BDR_SUNKENINNER)
3006
3007 /* border flags */
3008 #define BF_LEFT             0x0001
3009 #define BF_TOP              0x0002
3010 #define BF_RIGHT            0x0004
3011 #define BF_BOTTOM           0x0008
3012 #define BF_DIAGONAL         0x0010
3013 #define BF_MIDDLE           0x0800 /* Fill in the middle */
3014 #define BF_SOFT             0x1000 /* For softer buttons */
3015 #define BF_ADJUST           0x2000 /* Calculate the space left over */
3016 #define BF_FLAT             0x4000 /* For flat rather than 3D borders */
3017 #define BF_MONO             0x8000 /* For monochrome borders */
3018 #define BF_TOPLEFT          (BF_TOP | BF_LEFT)
3019 #define BF_TOPRIGHT         (BF_TOP | BF_RIGHT)
3020 #define BF_BOTTOMLEFT       (BF_BOTTOM | BF_LEFT)
3021 #define BF_BOTTOMRIGHT      (BF_BOTTOM | BF_RIGHT)
3022 #define BF_RECT             (BF_LEFT | BF_TOP | BF_RIGHT | BF_BOTTOM)
3023 #define BF_DIAGONAL_ENDTOPRIGHT (BF_DIAGONAL | BF_TOP | BF_RIGHT)
3024 #define BF_DIAGONAL_ENDTOPLEFT (BF_DIAGONAL | BF_TOP | BF_LEFT)
3025 #define BF_DIAGONAL_ENDBOTTOMLEFT (BF_DIAGONAL | BF_BOTTOM | BF_LEFT)
3026 #define BF_DIAGONAL_ENDBOTTOMRIGHT (BF_DIAGONAL | BF_BOTTOM | BF_RIGHT)
3027
3028 /* DrawFrameControl() uType's */
3029
3030 #define DFC_CAPTION          1
3031 #define DFC_MENU             2
3032 #define DFC_SCROLL           3
3033 #define DFC_BUTTON           4
3034
3035 /* uState's */
3036
3037 #define DFCS_CAPTIONCLOSE    0x0000
3038 #define DFCS_CAPTIONMIN     0x0001
3039 #define DFCS_CAPTIONMAX     0x0002
3040 #define DFCS_CAPTIONRESTORE 0x0003
3041 #define DFCS_CAPTIONHELP    0x0004 /* Windows 95 only */
3042
3043 #define DFCS_MENUARROW       0x0000
3044 #define DFCS_MENUCHECK       0x0001
3045 #define DFCS_MENUBULLET     0x0002
3046 #define DFCS_MENUARROWRIGHT 0x0004
3047
3048 #define DFCS_SCROLLUP        0x0000
3049 #define DFCS_SCROLLDOWN     0x0001
3050 #define DFCS_SCROLLLEFT     0x0002
3051 #define DFCS_SCROLLRIGHT    0x0003
3052 #define DFCS_SCROLLCOMBOBOX 0x0005
3053 #define DFCS_SCROLLSIZEGRIP  0x0008
3054 #define DFCS_SCROLLSIZEGRIPRIGHT 0x0010
3055
3056 #define DFCS_BUTTONCHECK     0x0000
3057 #define DFCS_BUTTONRADIOIMAGE 0x0001
3058 #define DFCS_BUTTONRADIOMASK 0x0002 /* to draw nonsquare button */
3059 #define DFCS_BUTTONRADIO     0x0004
3060 #define DFCS_BUTTON3STATE    0x0008
3061 #define DFCS_BUTTONPUSH      0x0010
3062
3063 /* additional state of the control */
3064
3065 #define DFCS_INACTIVE        0x0100
3066 #define DFCS_PUSHED          0x0200
3067 #define DFCS_CHECKED         0x0400
3068 #define DFCS_ADJUSTRECT     0x2000 /* exclude surrounding edge */
3069 #define DFCS_FLAT            0x4000
3070 #define DFCS_MONO            0x8000
3071
3072 /* Image type */
3073 #define DST_COMPLEX          0x0000
3074 #define DST_TEXT             0x0001
3075 #define DST_PREFIXTEXT       0x0002

```



```

3076 #define DST_ICON      0x0003
3077 #define DST_BITMAP     0x0004
3078
3079 /* State type */
3080 #define DSS_NORMAL      0x0000
3081 #define DSS_UNION       0x0010 /* Gray string appearance */
3082 #define DSS_DISABLED    0x0020
3083 #define DSS_DEFAULT     0x0040 /* Make it bold */
3084 #define DSS_MONO        0x0080
3085 #define DSS_RIGHT       0x8000
3086
3087 typedef struct
3088 {
3089     UINT        CtlType;
3090     UINT        CtlID;
3091     UINT        itemID;
3092     UINT        itemAction;
3093     UINT        itemState;
3094     HWND        hwndItem;
3095     HDC          hdc;
3096     RECT        rcItem WINE_PACKED;
3097     DWORD       itemData WINE_PACKED;
3098 } DRAWITEMSTRUCT, *PDRAWITEMSTRUCT, *LPDRAWITEMSTRUCT;
3099
3100
3101 typedef struct
3102 {
3103     UINT        CtlType;
3104     UINT        CtlID;
3105     UINT        itemID;
3106     UINT        itemWidth;
3107     UINT        itemHeight;
3108     DWORD       itemData;
3109 } MEASUREITEMSTRUCT, *PMEASUREITEMSTRUCT, *LPMEASUREITEMSTRUCT;
3110
3111
3112 typedef struct
3113 {
3114     UINT        CtlType;
3115     UINT        CtlID;
3116     UINT        itemID;
3117     HWND        hwndItem;
3118     DWORD       itemData;
3119 } DELETEITEMSTRUCT, *PDELETEITEMSTRUCT, *LPDELETEITEMSTRUCT;
3120
3121
3122 typedef struct
3123 {
3124     UINT        CtlType;
3125     UINT        CtlID;
3126     HWND        hwndItem;
3127     UINT        itemID1;
3128     DWORD       itemData1;
3129     UINT        itemID2;
3130     DWORD       itemData2;
3131     DWORD       dwLocaleId;
3132 } COMPAREITEMSTRUCT, *PCOMPAREITEMSTRUCT, *LPCOMPAREITEMSTRUCT;
3133
3134
3135 /* WM_KEYUP/DOWN/CHAR HIWORD(lParam) flags */
3136 #define KF_EXTENDED      0x0100
3137 #define KF_DLGMODE       0x0800
3138 #define KF_MENUMODE      0x1000
3139 #define KF_ALTDOWN       0x2000
3140 #define KF_REPEAT        0x4000
3141 #define KF_UP            0x8000
3142
3143 /* Virtual key codes */
3144 #define VK_LBUTTON       0x01
3145 #define VK_RBUTTON       0x02
3146 #define VK_CANCEL        0x03
3147 #define VK_MBUTTON       0x04
3148 #define VK_XBUTTON1      0x05
3149 #define VK_XBUTTON2      0x06
3150 /*                                0x07 Undefined */
3151 #define VK_BACK          0x08
3152 #define VK_TAB           0x09
3153 /*                                0x0A-0x0B Undefined */
3154 #define VK_CLEAR         0x0C
3155 #define VK_RETURN        0x0D
3156 /*                                0x0E-0x0F Undefined */
3157 #define VK_SHIFT         0x10
3158 #define VK_CONTROL       0x11
3159 #define VK_MENU          0x12
3160 #define VK_PAUSE         0x13
3161 #define VK_CAPITAL       0x14
3162 /*                                0x15-0x19 Reserved for Kanji systems */

```

```
3163 /*                                0x1A      Undefined */
3164 #define VK_ESCAPE                    0x1B
3165 /*                                0x1C-0x1F  Reserved for Kanji systems */
3166 #define VK_SPACE                      0x20
3167 #define VK_PRIOR                     0x21
3168 #define VK_NEXT                      0x22
3169 #define VK_END                       0x23
3170 #define VK_HOME                      0x24
3171 #define VK_LEFT                      0x25
3172 #define VK_UP                        0x26
3173 #define VK_RIGHT                     0x27
3174 #define VK_DOWN                      0x28
3175 #define VK_SELECT                    0x29
3176 #define VK_PRINT                     0x2A /* OEM specific in Windows 3.1 SDK */
3177 #define VK_EXECUTE                   0x2B
3178 #define VK_SNAPSHOT                  0x2C
3179 #define VK_INSERT                    0x2D
3180 #define VK_DELETE                    0x2E
3181 #define VK_HELP                      0x2F
3182 #define VK_0                        0x30
3183 #define VK_1                        0x31
3184 #define VK_2                        0x32
3185 #define VK_3                        0x33
3186 #define VK_4                        0x34
3187 #define VK_5                        0x35
3188 #define VK_6                        0x36
3189 #define VK_7                        0x37
3190 #define VK_8                        0x38
3191 #define VK_9                        0x39
3192 /*                                0x3A-0x40  Undefined */
3193 #define VK_A                         0x41
3194 #define VK_B                         0x42
3195 #define VK_C                         0x43
3196 #define VK_D                         0x44
3197 #define VK_E                         0x45
3198 #define VK_F                         0x46
3199 #define VK_G                         0x47
3200 #define VK_H                         0x48
3201 #define VK_I                         0x49
3202 #define VK_J                         0x4A
3203 #define VK_K                         0x4B
3204 #define VK_L                         0x4C
3205 #define VK_M                         0x4D
3206 #define VK_N                         0x4E
3207 #define VK_O                         0x4F
3208 #define VK_P                         0x50
3209 #define VK_Q                         0x51
3210 #define VK_R                         0x52
3211 #define VK_S                         0x53
3212 #define VK_T                         0x54
3213 #define VK_U                         0x55
3214 #define VK_V                         0x56
3215 #define VK_W                         0x57
3216 #define VK_X                         0x58
3217 #define VK_Y                         0x59
3218 #define VK_Z                         0x5A
3219
3220 #define VK_LWIN                      0x5B
3221 #define VK_RWIN                      0x5C
3222 #define VK_APPS                      0x5D
3223 /*                                0x5E-0x5F  Unassigned */
3224 #define VK_NUMPAD0                   0x60
3225 #define VK_NUMPAD1                   0x61
3226 #define VK_NUMPAD2                   0x62
3227 #define VK_NUMPAD3                   0x63
3228 #define VK_NUMPAD4                   0x64
3229 #define VK_NUMPAD5                   0x65
3230 #define VK_NUMPAD6                   0x66
3231 #define VK_NUMPAD7                   0x67
3232 #define VK_NUMPAD8                   0x68
3233 #define VK_NUMPAD9                   0x69
3234 #define VK_MULTIPLY                   0x6A
3235 #define VK_ADD                        0x6B
3236 #define VK_SEPARATOR                 0x6C
3237 #define VK_SUBTRACT                   0x6D
3238 #define VK_DECIMAL                   0x6E
3239 #define VK_DIVIDE                     0x6F
3240 #define VK_F1                        0x70
3241 #define VK_F2                        0x71
3242 #define VK_F3                        0x72
3243 #define VK_F4                        0x73
3244 #define VK_F5                        0x74
3245 #define VK_F6                        0x75
3246 #define VK_F7                        0x76
3247 #define VK_F8                        0x77
3248 #define VK_F9                        0x78
3249 #define VK_F10                       0x79
```

```

3250 #define VK_F11            0x7A
3251 #define VK_F12            0x7B
3252 #define VK_F13            0x7C
3253 #define VK_F14            0x7D
3254 #define VK_F15            0x7E
3255 #define VK_F16            0x7F
3256 #define VK_F17            0x80
3257 #define VK_F18            0x81
3258 #define VK_F19            0x82
3259 #define VK_F20            0x83
3260 #define VK_F21            0x84
3261 #define VK_F22            0x85
3262 #define VK_F23            0x86
3263 #define VK_F24            0x87
3264 /*            0x88-0x8F    Unassigned */
3265 #define VK_NUMLOCK         0x90
3266 #define VK_SCROLL          0x91
3267 /*            0x92-0x9F    Unassigned */
3268 /*
3269 * differencing between right and left shift/control/alt key.
3270 * Used only by GetAsyncKeyState() and GetKeyState().
3271 */
3272 #define VK_LSHIFT          0xA0
3273 #define VK_RSHIFT          0xA1
3274 #define VK_LCONTROL        0xA2
3275 #define VK_RCONTROL        0xA3
3276 #define VK_LMENU           0xA4
3277 #define VK_RMENU           0xA5
3278 /*            0xA6-0xB9    Unassigned */
3279 #define VK_OEM_1           0xBA
3280 #define VK_OEM_PLUS        0xBB
3281 #define VK_OEM_COMMA       0xBC
3282 #define VK_OEM_MINUS       0xBD
3283 #define VK_OEM_PERIOD      0xBE
3284 #define VK_OEM_2           0xBF
3285 #define VK_OEM_3           0xC0
3286 /*            0xC1-0xDA    Unassigned */
3287 #define VK_OEM_4           0xDB
3288 #define VK_OEM_5           0xDC
3289 #define VK_OEM_6           0xDD
3290 #define VK_OEM_7           0xDE
3291 #define VK_OEM_8           0xDF
3292 /*            0xE0          OEM specific */
3293 #define VK_OEM_AX           0xE1 /* "AX" key on Japanese AX keyboard */
3294 #define VK_OEM_102         0xE2 /* "<>" or "\|" on RT 102-key keyboard */
3295 #define VK_ICO_HELP        0xE3 /* Help key on ICO */
3296 #define VK_ICO_00          0xE4 /* 00 key on ICO */
3297 #define VK_PROCESSKEY      0xE5
3298
3299 /*            0xE6          OEM specific */
3300 /*            0xE7-0xE8    Unassigned */
3301 /*            0xE9-0xF5    OEM specific */
3302
3303 #define VK_ATTN            0xF6
3304 #define VK_CRSEL           0xF7
3305 #define VK_EXSEL           0xF8
3306 #define VK_EREOF           0xF9
3307 #define VK_PLAY            0xFA
3308 #define VK_ZOOM            0xFB
3309 #define VK_NONAME          0xFC
3310 #define VK_PA1             0xFD
3311 #define VK_OEM_CLEAR       0xFE
3312
3313 /* Key status flags for mouse events */
3314 #define MK_LBUTTON         0x0001
3315 #define MK_RBUTTON         0x0002
3316 #define MK_SHIFT           0x0004
3317 #define MK_CONTROL         0x0008
3318 #define MK_MBUTTON         0x0010
3319 #define MK_XBUTTON1        0x0020
3320 #define MK_XBUTTON2        0x0040
3321
3322 /* Queue status flags */
3323 #define QS_KEY             0x0001
3324 #define QS_MOUSEMOVE       0x0002
3325 #define QS_MOUSEBUTTON     0x0004
3326 #define QS_MOUSE           (QS_MOUSEMOVE | QS_MOUSEBUTTON)
3327 #define QS_POSTMESSAGE     0x0008
3328 #define QS_TIMER           0x0010
3329 #define QS_PAINT           0x0020
3330 #define QS_SENDMESSAGE     0x0040
3331 #define QS_HOTKEY          0x0080
3332 #define QS_INPUT           (QS_MOUSE | QS_KEY)
3333 #define QS_ALLEVENTS       (QS_INPUT | QS_POSTMESSAGE | QS_TIMER | QS_PAINT | QS_HOTKEY)
3334 #define QS_ALLINPUT        (QS_ALLEVENTS | QS_SENDMESSAGE)
3335
3336 /* Extra (undocumented) queue wake bits - see "Undoc. Windows" */

```

```
3337 #define QS_SMRESULT      0x8000
3338
3339 /* InSendMessageEx flags */
3340 #define ISMEX_NOSEND      0x00000000
3341 #define ISMEX_SEND       0x00000001
3342 #define ISMEX_NOTIFY     0x00000002
3343 #define ISMEX_CALLBACK   0x00000004
3344 #define ISMEX_REPLIED    0x00000008
3345
3346 #define DDL_READWRITE    0x0000
3347 #define DDL_READONLY    0x0001
3348 #define DDL_HIDDEN      0x0002
3349 #define DDL_SYSTEM      0x0004
3350 #define DDL_DIRECTORY   0x0010
3351 #define DDL_ARCHIVE     0x0020
3352
3353 #define DDL_POSTMSG     0x2000
3354 #define DDL_DRIVES      0x4000
3355 #define DDL_EXCLUSIVE   0x8000
3356
3357 /* Shell hook values */
3358 #define HSHELL_WINDOWCREATED 1
3359 #define HSHELL_WINDOWDESTROYED 2
3360 #define HSHELL_ACTIVATESHELLWINDOW 3
3361
3362 /* Predefined Clipboard Formats */
3363 #define CF_TEXT          1
3364 #define CF_BITMAP        2
3365 #define CF_METAFILEPICT  3
3366 #define CF_SYLK          4
3367 #define CF_DIF           5
3368 #define CF_TIFF          6
3369 #define CF_OEMTEXT       7
3370 #define CF_DIB           8
3371 #define CF_PALETTE       9
3372 #define CF_PENDATA       10
3373 #define CF_RIFF          11
3374 #define CF_WAVE          12
3375 #define CF_UNICODETEXT   13
3376 #define CF_ENHMETAFILE   14
3377 #define CF_HDROP         15
3378 #define CF_LOCALE        16
3379 #define CF_DIBV5         17
3380 #define CF_MAX           18
3381
3382 #define CF_OWNERDISPLAY   0x0080
3383 #define CF_DSPTEXT       0x0081
3384 #define CF_DSPBITMAP     0x0082
3385 #define CF_DSPMETAFILEPICT 0x0083
3386 #define CF_DSPENHMETAFILE 0x008E
3387
3388 /* "Private" formats don't get GlobalFree()'d */
3389 #define CF_PRIVATEFIRST   0x0200
3390 #define CF_PRIVATELAST   0x02FF
3391
3392 /* "GDI OBJ" formats do get DeleteObject()'d */
3393 #define CF_GDIOBJFIRST    0x0300
3394 #define CF_GDIOBJLAST    0x03FF
3395
3396
3397 /* types of LoadImage */
3398 #define IMAGE_BITMAP      0
3399 #define IMAGE_ICON       1
3400 #define IMAGE_CURSOR     2
3401 #define IMAGE_ENHMETAFILE 3
3402
3403 /* loadflags to LoadImage */
3404 #define LR_DEFAULTCOLOR   0x0000
3405 #define LR_MONOCHROME     0x0001
3406 #define LR_COLOR          0x0002
3407 #define LR_COPYRETURNORG  0x0004
3408 #define LR_COPYDELETEORG  0x0008
3409 #define LR_LOADFROMFILE   0x0010
3410 #define LR_LOADTRANSPARENT 0x0020
3411 #define LR_DEFAULTSIZE    0x0040
3412 #define LR_VGA_COLOR      0x0080
3413 #define LR_LOADMAP3DCOLORS 0x1000
3414 #define LR_CREATEDIBSECTION 0x2000
3415 #define LR_COPYFROMRESOURCE 0x4000
3416 #define LR_SHARED         0x8000
3417
3418 /* Flags for DrawIconEx. */
3419 #define DI_MASK           1
3420 #define DI_IMAGE          2
3421 #define DI_NORMAL         (DI_MASK | DI_IMAGE)
3422 #define DI_COMPAT         4
3423 #define DI_DEFAULTSIZE    8
```

```

3424
3425 /* WM_NOTIFYFORMAT commands and return values */
3426 #define NFR_ANSI 1
3427 #define NFR_UNICODE 2
3428 #define NF_QUERY 3
3429 #define NF_QUERY 4
3430
3431 /* RegisterDeviceNotification stuff */
3432 typedef PVOID HDEVNOTIFY;
3433 typedef HDEVNOTIFY *PHDEVNOTIFY;
3434
3435 #define DEVICE_NOTIFY_WINDOW_HANDLE 0x00000000
3436
3437 #define EnumTaskWindows(handle,proc,lparam) \
3438 EnumThreadWindows(handle,proc,lparam)
3439 #define OemToAnsiA OemToCharA
3440 #define OemToAnsiW OemToCharW
3441 #define OemToAnsi WINELIB_NAME_AW(OemToAnsi)
3442 #define OemToAnsiBuffA OemToCharBuffA
3443 #define OemToAnsiBuffW OemToCharBuffW
3444 #define OemToAnsiBuff WINELIB_NAME_AW(OemToAnsiBuff)
3445 #define AnsiToOemA CharToOemA
3446 #define AnsiToOemW CharToOemW
3447 #define AnsiToOem WINELIB_NAME_AW(AnsiToOem)
3448 #define AnsiToOemBuffA CharToOemBuffA
3449 #define AnsiToOemBuffW CharToOemBuffW
3450 #define AnsiToOemBuff WINELIB_NAME_AW(AnsiToOemBuff)
3451
3452 #if defined(_WINGDI_) && !defined(NO_GDI)
3453 LONG WINAPI ChangeDisplaySettingsA(LPDEVMODEA,DWORD);
3454 LONG WINAPI ChangeDisplaySettingsW(LPDEVMODEW,DWORD);
3455 #define ChangeDisplaySettings WINELIB_NAME_AW(ChangeDisplaySettings)
3456 LONG WINAPI ChangeDisplaySettingsExA(LPCSTR,LPDEVMODEA,HWND,DWORD,LPARAM);
3457 LONG WINAPI ChangeDisplaySettingsExW(LPCWSTR,LPDEVMODEW,HWND,DWORD,LPARAM);
3458 #define ChangeDisplaySettingsEx WINELIB_NAME_AW(ChangeDisplaySettingsEx)
3459 BOOL WINAPI EnumDisplayDevicesA(LPVOID,DWORD,LPDISPLAY_DEVICEA,DWORD);
3460 BOOL WINAPI EnumDisplayDevicesW(LPVOID,DWORD,LPDISPLAY_DEVICEW,DWORD);
3461 #define EnumDisplayDevices WINELIB_NAME_AW(EnumDisplayDevices)
3462 BOOL WINAPI EnumDisplaySettingsA(LPCSTR,DWORD,LPDEVMODEA);
3463 BOOL WINAPI EnumDisplaySettingsW(LPCWSTR,DWORD,LPDEVMODEW);
3464 #define EnumDisplaySettings WINELIB_NAME_AW(EnumDisplaySettings)
3465 #endif /* defined(_WINGDI_) && !defined(NO_GDI) */
3466
3467 HKL WINAPI ActivateKeyboardLayout(HKL,UINT);
3468 LONG WINAPI BroadcastSystemMessage(DWORD,LPDWORD,UINT,WPARAM,LPARAM);
3469 WORD WINAPI CascadeWindows(HWND,UINT,const LPRECT,UINT,const HWND *);
3470 INT WINAPI CopyAcceleratorTableA(HACCEL,LPACCEL,INT);
3471 INT WINAPI CopyAcceleratorTableW(HACCEL,LPACCEL,INT);
3472 #define CopyAcceleratorTable WINELIB_NAME_AW(CopyAcceleratorTable)
3473 HACCEL WINAPI CreateAcceleratorTableA(LPACCEL,INT);
3474 HACCEL WINAPI CreateAcceleratorTableW(LPACCEL,INT);
3475 #define CreateAcceleratorTable WINELIB_NAME_AW(CreateAcceleratorTable)
3476 HICON WINAPI CreateIconIndirect(PICONINFO);
3477 BOOL WINAPI DestroyAcceleratorTable(HACCEL);
3478 BOOL WINAPI EnumDesktopsA(HWINSTA,DESKTOPENUMPROCA,LPARAM);
3479 BOOL WINAPI EnumDesktopsW(HWINSTA,DESKTOPENUMPROCW,LPARAM);
3480 #define EnumDesktops WINELIB_NAME_AW(EnumDesktops)
3481 BOOL WINAPI EnumDisplayMonitors(HDC,LPRECT,MONITORENUMPROC,LPARAM);
3482 INT WINAPI EnumPropsExA(HWND,PROPNUMPROCXA,LPARAM);
3483 INT WINAPI EnumPropsExW(HWND,PROPNUMPROCEXW,LPARAM);
3484 #define EnumPropsEx WINELIB_NAME_AW(EnumPropsEx)
3485 BOOL WINAPI EnumThreadWindows(DWORD,WNDENUMPROC,LPARAM);
3486 BOOL WINAPI ExitWindowsEx(UINT,DWORD);
3487 BOOL WINAPI GetIconInfo(HICON,PICONINFO);
3488 HKL WINAPI GetKeyboardLayout(DWORD);
3489 INT WINAPI GetKeyboardLayoutList(INT,HKL *);
3490 DWORD WINAPI GetMenuContextHelpId(HMENU);
3491 UINT WINAPI GetMenuDefaultItem(HMENU,UINT,UINT);
3492 BOOL WINAPI GetMenuInfo(HMENU,LPMENUINFO);
3493 BOOL WINAPI GetMenuItemInfoA(HMENU,UINT,BOOL,MENUITEMINFOA *);
3494 BOOL WINAPI GetMenuItemInfoW(HMENU,UINT,BOOL,MENUITEMINFOW *);
3495 #define GetMenuItemInfo WINELIB_NAME_AW(GetMenuItemInfo)
3496 BOOL WINAPI GetMonitorInfoA(HMONITOR,LPMONITORINFO);
3497 BOOL WINAPI GetMonitorInfoW(HMONITOR,LPMONITORINFO);
3498 #define GetMonitorInfo WINELIB_NAME_AW(GetMonitorInfo)
3499 DWORD WINAPI GetWindowContextHelpId(HWND);
3500 DWORD WINAPI GetWindowThreadProcessId(HWND,LPDWORD);
3501 BOOL WINAPI IsWindowUnicode(HWND);
3502 HKL WINAPI LoadKeyboardLayoutA(LPCSTR,UINT);
3503 HKL WINAPI LoadKeyboardLayoutW(LPCWSTR,UINT);
3504 #define LoadKeyboardLayout WINELIB_NAME_AW(LoadKeyboardLayout)
3505 INT WINAPI MessageBoxExA(HWND,LPCSTR,LPCSTR,UINT,WORD);
3506 INT WINAPI MessageBoxExW(HWND,LPCWSTR,LPCWSTR,UINT,WORD);
3507 #define MessageBoxEx WINELIB_NAME_AW(MessageBoxEx)
3508 HMONITOR WINAPI MonitorFromPoint(POINT,DWORD);
3509 HMONITOR WINAPI MonitorFromRect(LPRECT,DWORD);
3510 HMONITOR WINAPI MonitorFromWindow(HWND,DWORD);

```

```

3511 DWORD      WINAPI MsgWaitForMultipleObjects (DWORD, CONST HANDLE*, BOOL, DWORD, DWORD);
3512 DWORD      WINAPI MsgWaitForMultipleObjectsEx (DWORD, CONST HANDLE*, DWORD, DWORD, DWORD);
3513 BOOL        WINAPI PaintDesktop (HDC);
3514 BOOL        WINAPI PostThreadMessageA (DWORD, UINT, WPARAM, LPARAM);
3515 BOOL        WINAPI PostThreadMessageW (DWORD, UINT, WPARAM, LPARAM);
3516 #define      PostThreadMessage WINELIB_NAME_AW (PostThreadMessage)
3517 BOOL        WINAPI RegisterHotKey (HWND, INT, UINT, UINT);
3518 HDEVNOTIFY  WINAPI RegisterDeviceNotificationA (HANDLE, LPVOID, DWORD);
3519 HDEVNOTIFY  WINAPI RegisterDeviceNotificationW (HANDLE, LPVOID, DWORD);
3520 #define      RegisterDeviceNotification WINELIB_NAME_AW (RegisterDeviceNotification)
3521 BOOL        WINAPI SendMessageCallbackA (HWND, UINT, WPARAM, LPARAM, SENDASYNCPROC, ULONG_PTR);
3522 BOOL        WINAPI SendMessageCallbackW (HWND, UINT, WPARAM, LPARAM, SENDASYNCPROC, ULONG_PTR);
3523 #define      SendMessageCallback WINELIB_NAME_AW (SendMessageCallback)
3524 BOOL        WINAPI SendNotifyMessageA (HWND, UINT, WPARAM, LPARAM);
3525 BOOL        WINAPI SendNotifyMessageW (HWND, UINT, WPARAM, LPARAM);
3526 #define      SendNotifyMessage WINELIB_NAME_AW (SendNotifyMessage)
3527 VOID        WINAPI SetDebugErrorLevel (DWORD);
3528 VOID        WINAPI SetLastErrorEx (DWORD, DWORD);
3529 BOOL        WINAPI SetMenuDefaultItem (HMENU, UINT, UINT);
3530 BOOL        WINAPI SetMenuInfo (HMENU, LPCMENUINFO);
3531 BOOL        WINAPI SetMenuItemInfoA (HMENU, UINT, BOOL, const MENUITEMINFOA*);
3532 BOOL        WINAPI SetMenuItemInfoW (HMENU, UINT, BOOL, const MENUITEMINFOW*);
3533 #define      SetMenuItemInfo WINELIB_NAME_AW (SetMenuItemInfo)
3534 BOOL        WINAPI SetWindowContextHelpId (HWND, DWORD);
3535 WORD        WINAPI TileWindows (HWND, UINT, const LPRECT,
3536                                UINT, const HWND *);
3537 INT         WINAPI ToUnicode (UINT, UINT, PBYTE, LPWSTR, int, UINT);
3538 BOOL        WINAPI TrackPopupMenuEx (HMENU, UINT, INT, INT, HWND,
3539                                     LPTMPARAMS);
3540 BOOL        WINAPI UnregisterDeviceNotification (HDEVNOTIFY);
3541 BOOL        WINAPI UnregisterHotKey (HWND, INT);
3542 DWORD       WINAPI WaitForInputIdle (HANDLE, DWORD);
3543 VOID        WINAPI keybd_event (BYTE, BYTE, DWORD, DWORD);
3544 VOID        WINAPI mouse_event (DWORD, DWORD, DWORD, DWORD, DWORD);
3545
3546 /* Declarations for functions that are the same in Win16 and Win32 */
3547 VOID        WINAPI EndMenu (void);
3548 DWORD       WINAPI GetDialogBaseUnits (void);
3549 BOOL        WINAPI GetKeyboardState (LPBYTE);
3550 DWORD       WINAPI GetMenuCheckMarkDimensions (void);
3551 LONG        WINAPI GetMessageExtraInfo (void);
3552 DWORD       WINAPI GetMessagePos (void);
3553 LONG        WINAPI GetMessageTime (void);
3554 DWORD       WINAPI OemKeyScan (WORD);
3555 BOOL        WINAPI ReleaseCapture (void);
3556 BOOL        WINAPI SetKeyboardState (LPBYTE);
3557
3558 /* Declarations for functions that change between Win16 and Win32 */
3559
3560 BOOL        WINAPI AdjustWindowRect (LPRECT, DWORD, BOOL);
3561 BOOL        WINAPI AdjustWindowRectEx (LPRECT, DWORD, BOOL, DWORD);
3562 BOOL        WINAPI AnimateWindow (HWND, DWORD, DWORD);
3563 #define      AnsiLowerA CharLowerA
3564 #define      AnsiLowerW CharLowerW
3565 #define      AnsiLower WINELIB_NAME_AW (AnsiLower)
3566 #define      AnsiLowerBuffA CharLowerBuffA
3567 #define      AnsiLowerBuffW CharLowerBuffW
3568 #define      AnsiLowerBuff WINELIB_NAME_AW (AnsiLowerBuff)
3569 #define      AnsiNextA CharNextA
3570 #define      AnsiNextW CharNextW
3571 #define      AnsiNext WINELIB_NAME_AW (AnsiNext)
3572 #define      AnsiPrevA CharPrevA
3573 #define      AnsiPrevW CharPrevW
3574 #define      AnsiPrev WINELIB_NAME_AW (AnsiPrev)
3575 #define      AnsiUpperA CharUpperA
3576 #define      AnsiUpperW CharUpperW
3577 #define      AnsiUpper WINELIB_NAME_AW (AnsiUpper)
3578 #define      AnsiUpperBuffA CharUpperBuffA
3579 #define      AnsiUpperBuffW CharUpperBuffW
3580 #define      AnsiUpperBuff WINELIB_NAME_AW (AnsiUpperBuff)
3581 BOOL        WINAPI AnyPopup (void);
3582 BOOL        WINAPI AppendMenuA (HMENU, UINT, UINT, LPCSTR);
3583 BOOL        WINAPI AppendMenuW (HMENU, UINT, UINT, LPCWSTR);
3584 #define      AppendMenu WINELIB_NAME_AW (AppendMenu)
3585 UINT        WINAPI ArrangeIconicWindows (HWND);
3586 HDWP        WINAPI BeginDeferWindowPos (INT);
3587 HDC         WINAPI BeginPaint (HWND, LPPAINTSTRUCT);
3588 BOOL        WINAPI BringWindowToTop (HWND);
3589 void        WINAPI CalcChildScroll (HWND, INT);
3590 BOOL        WINAPI CallMsgFilterA (LPMSG, INT);
3591 BOOL        WINAPI CallMsgFilterW (LPMSG, INT);
3592 #define      CallMsgFilter WINELIB_NAME_AW (CallMsgFilter)
3593 LRESULT      WINAPI CallNextHookEx (HOOK, INT, WPARAM, LPARAM);
3594 LRESULT      WINAPI CallWindowProcA (WNDPROC, HWND, UINT, WPARAM, LPARAM);
3595 LRESULT      WINAPI CallWindowProcW (WNDPROC, HWND, UINT, WPARAM, LPARAM);
3596 #define      CallWindowProc WINELIB_NAME_AW (CallWindowProc)
3597 BOOL        WINAPI ChangeClipboardChain (HWND, HWND);

```

```

3598 BOOL        WINAPI ChangeMenuA(HMENU,UINT,LPCSTR,UINT,UINT);
3599 BOOL        WINAPI ChangeMenuW(HMENU,UINT,LPCWSTR,UINT,UINT);
3600 #define      ChangeMenu WINELIB_NAME_AW(ChangeMenu)
3601 LPSTR        WINAPI CharLowerA(LPSTR);
3602 LPWSTR       WINAPI CharLowerW(LPWSTR);
3603 #define      CharLower WINELIB_NAME_AW(CharLower)
3604 DWORD        WINAPI CharLowerBuffA(LPSTR,DWORD);
3605 DWORD        WINAPI CharLowerBuffW(LPWSTR,DWORD);
3606 #define      CharLowerBuff WINELIB_NAME_AW(CharLowerBuff)
3607 LPSTR        WINAPI CharNextA(LPCSTR);
3608 LPWSTR       WINAPI CharNextW(LPCWSTR);
3609 #define      CharNext WINELIB_NAME_AW(CharNext)
3610 LPSTR        WINAPI CharNextExA(WORD,LPCSTR,DWORD);
3611 /* no CharNextExW (doesn't make sense) */
3612 LPSTR        WINAPI CharPrevA(LPCSTR,LPCSTR);
3613 LPWSTR       WINAPI CharPrevW(LPCWSTR,LPCWSTR);
3614 #define      CharPrev WINELIB_NAME_AW(CharPrev)
3615 LPSTR        WINAPI CharPrevExA(WORD,LPCSTR,LPCSTR,DWORD);
3616 /* no CharPrevExW (doesn't make sense) */
3617 LPSTR        WINAPI CharUpperA(LPSTR);
3618 LPWSTR       WINAPI CharUpperW(LPWSTR);
3619 #define      CharUpper WINELIB_NAME_AW(CharUpper)
3620 DWORD        WINAPI CharUpperBuffA(LPSTR,DWORD);
3621 DWORD        WINAPI CharUpperBuffW(LPWSTR,DWORD);
3622 #define      CharUpperBuff WINELIB_NAME_AW(CharUpperBuff)
3623 BOOL        WINAPI CharToOemA(LPCSTR,LPSTR);
3624 BOOL        WINAPI CharToOemW(LPCWSTR,LPSTR);
3625 #define      CharToOem WINELIB_NAME_AW(CharToOem)
3626 BOOL        WINAPI CharToOemBuffA(LPCSTR,LPSTR,DWORD);
3627 BOOL        WINAPI CharToOemBuffW(LPCWSTR,LPSTR,DWORD);
3628 #define      CharToOemBuff WINELIB_NAME_AW(CharToOemBuff)
3629 BOOL        WINAPI CheckDlgButton(HWND,INT,UINT);
3630 DWORD        WINAPI CheckMenuItem(HMENU,UINT,UINT);
3631 BOOL        WINAPI CheckMenuRadioItem(HMENU,UINT,UINT,UINT,UINT);
3632 BOOL        WINAPI CheckRadioButton(HWND,UINT,UINT,UINT);
3633 HWND        WINAPI ChildWindowFromPoint(HWND,POINT);
3634 HWND        WINAPI ChildWindowFromPointEx(HWND,POINT,UINT);
3635 BOOL        WINAPI ClientToScreen(HWND,LPPPOINT);
3636 BOOL        WINAPI ClipCursor(const RECT*);
3637 BOOL        WINAPI CloseClipboard(void);
3638 BOOL        WINAPI CloseDesktop(HDESK);
3639 BOOL        WINAPI CloseWindow(HWND);
3640 BOOL        WINAPI CloseWindowStation(HWINSTA);
3641 #define      CopyCursor(cur) ((HCURSOR)CopyIcon((HICON)(cur)))
3642 HICON        WINAPI CopyIcon(HICON);
3643 HICON        WINAPI CopyImage(HANDLE,UINT,INT,INT,UINT);
3644 BOOL        WINAPI CopyRect(RECT*,const RECT*);
3645 INT          WINAPI CountClipboardFormats(void);
3646 BOOL        WINAPI CreateCaret(HWND,HBITMAP,INT,INT);
3647 HCURSOR      WINAPI CreateCursor(HINSTANCE,INT,INT,INT,LPCVOID,LPCVOID);
3648 #define      CreateDialogA(inst,ptr,hwnd,dlg) \
3649 CreateDialogParamA(inst,ptr,hwnd,dlg,0)
3650 #define      CreateDialogW(inst,ptr,hwnd,dlg) \
3651 CreateDialogParamW(inst,ptr,hwnd,dlg,0)
3652 #define      CreateDialog WINELIB_NAME_AW(CreateDialog)
3653 #define      CreateDialogIndirectA(inst,ptr,hwnd,dlg) \
3654 CreateDialogIndirectParamA(inst,ptr,hwnd,dlg,0)
3655 #define      CreateDialogIndirectW(inst,ptr,hwnd,dlg) \
3656 CreateDialogIndirectParamW(inst,ptr,hwnd,dlg,0)
3657 #define      CreateDialogIndirect WINELIB_NAME_AW(CreateDialogIndirect)
3658 HWND        WINAPI CreateDialogIndirectParamA(HINSTANCE,LPCVOID,HWND,
3659         DLGPROC,LPARAM);
3660 HWND        WINAPI CreateDialogIndirectParamW(HINSTANCE,LPCVOID,HWND,
3661         DLGPROC,LPARAM);
3662 #define      CreateDialogIndirectParam WINELIB_NAME_AW(CreateDialogIndirectParam)
3663 HWND        WINAPI CreateDialogParamA(HINSTANCE,LPCSTR,HWND,DLGPROC,LPARAM);
3664 HWND        WINAPI CreateDialogParamW(HINSTANCE,LPCWSTR,HWND,DLGPROC,LPARAM);
3665 #define      CreateDialogParam WINELIB_NAME_AW(CreateDialogParam)
3666 HICON        WINAPI CreateIcon(HINSTANCE,INT,INT,BYTE,BYTE,LPCVOID);
3667 HICON        WINAPI CreateIconFromResource(LPBYTE,UINT,BOOL,DWORD);
3668 HICON        WINAPI CreateIconFromResourceEx(LPBYTE,UINT,BOOL,DWORD,INT,INT,UINT);
3669 HMENU        WINAPI CreateMenu(void);
3670 HMENU        WINAPI CreatePopupMenu(void);
3671 #define      CreateWindowA(className,titleName,style,x,y,width,height,\
3672 parent,menu,instance,param) \
3673 CreateWindowExA(0,className,titleName,style,x,y,width,height,\
3674 parent,menu,instance,param)
3675 #define      CreateWindowW(className,titleName,style,x,y,width,height,\
3676 parent,menu,instance,param) \
3677 CreateWindowExW(0,className,titleName,style,x,y,width,height,\
3678 parent,menu,instance,param)
3679 #define      CreateWindow WINELIB_NAME_AW(CreateWindow)
3680 HWND        WINAPI CreateWindowExA(DWORD,LPCSTR,LPCSTR,DWORD,INT,INT,
3681         INT,INT,HWND,HMENU,HINSTANCE,LPCVOID);
3682 HWND        WINAPI CreateWindowExW(DWORD,LPCWSTR,LPCWSTR,DWORD,INT,INT,
3683         INT,INT,HWND,HMENU,HINSTANCE,LPCVOID);
3684 #define      CreateWindowEx WINELIB_NAME_AW(CreateWindowEx)

```



```

3685 HWINSTA      WINAPI CreateWindowStationA(LPSTR,DWORD,DWORD,LPSECURITY_ATTRIBUTES);
3686 HWINSTA      WINAPI CreateWindowStationW(LPWSTR,DWORD,DWORD,LPSECURITY_ATTRIBUTES);
3687 #define        CreateWindowStation WINELIB_NAME_AW(CreateWindowStation)
3688 HWND           WINAPI CreateMDIWindowA(LPCSTR,LPCSTR,DWORD,INT,INT,
3689                                     INT,INT,HWND,HINSTANCE,LPARAM);
3690 HWND           WINAPI CreateMDIWindowW(LPCWSTR,LPCWSTR,DWORD,INT,INT,
3691                                     INT,INT,HWND,HINSTANCE,LPARAM);
3692 #define        CreateMDIWindow WINELIB_NAME_AW(CreateMDIWindow)
3693 LRESULT        WINAPI DefDlgProcA(HWND,UINT,WPARAM,LPARAM);
3694 LRESULT        WINAPI DefDlgProcW(HWND,UINT,WPARAM,LPARAM);
3695 #define        DefDlgProc WINELIB_NAME_AW(DefDlgProc)
3696 HDWP           WINAPI DeferWindowPos(HDWP,HWND,HWND,INT,INT,INT,UINT);
3697 LRESULT        WINAPI DefFrameProcA(HWND,HWND,UINT,WPARAM,LPARAM);
3698 LRESULT        WINAPI DefFrameProcW(HWND,HWND,UINT,WPARAM,LPARAM);
3699 #define        DefFrameProc WINELIB_NAME_AW(DefFrameProc)
3700 #define        DefHookProc(code,wparam,lparam,phhook) \
3701 CallNextHookEx(*(phhook),code,wparam,lparam)
3702 LRESULT        WINAPI DefMDIChildProcA(HWND,UINT,WPARAM,LPARAM);
3703 LRESULT        WINAPI DefMDIChildProcW(HWND,UINT,WPARAM,LPARAM);
3704 #define        DefMDIChildProc WINELIB_NAME_AW(DefMDIChildProc)
3705 LRESULT        WINAPI DefWindowProcA(HWND,UINT,WPARAM,LPARAM);
3706 LRESULT        WINAPI DefWindowProcW(HWND,UINT,WPARAM,LPARAM);
3707 #define        DefWindowProc WINELIB_NAME_AW(DefWindowProc)
3708 BOOL           WINAPI DeleteMenu(HMENU,UINT,UINT);
3709 BOOL           WINAPI DestroyCaret(void);
3710 BOOL           WINAPI DestroyCursor(HCURSOR);
3711 BOOL           WINAPI DestroyIcon(HICON);
3712 BOOL           WINAPI DestroyMenu(HMENU);
3713 BOOL           WINAPI DestroyWindow(HWND);
3714 #define        DialogBoxA(inst,template,owner,func) \
3715 DialogBoxParamA(inst,template,owner,func,0)
3716 #define        DialogBoxW(inst,template,owner,func) \
3717 DialogBoxParamW(inst,template,owner,func,0)
3718 #define        DialogBox WINELIB_NAME_AW(DialogBox)
3719 #define        DialogBoxIndirectA(inst,template,owner,func) \
3720 DialogBoxIndirectParamA(inst,template,owner,func,0)
3721 #define        DialogBoxIndirectW(inst,template,owner,func) \
3722 DialogBoxIndirectParamW(inst,template,owner,func,0)
3723 #define        DialogBoxIndirect WINELIB_NAME_AW(DialogBoxIndirect)
3724 INT            WINAPI DialogBoxIndirectParamA(HINSTANCE,LPCVOID,HWND,DLGPROC,LPARAM);
3725 INT            WINAPI DialogBoxIndirectParamW(HINSTANCE,LPCVOID,HWND,DLGPROC,LPARAM);
3726 #define        DialogBoxIndirectParam WINELIB_NAME_AW(DialogBoxIndirectParam)
3727 INT            WINAPI DialogBoxParamA(HINSTANCE,LPCSTR,HWND,DLGPROC,LPARAM);
3728 INT            WINAPI DialogBoxParamW(HINSTANCE,LPCWSTR,HWND,DLGPROC,LPARAM);
3729 #define        DialogBoxParam WINELIB_NAME_AW(DialogBoxParam)
3730 LONG           WINAPI DispatchMessageA(const MSG*);
3731 LONG           WINAPI DispatchMessageW(const MSG*);
3732 #define        DispatchMessage WINELIB_NAME_AW(DispatchMessage)
3733 INT            WINAPI DlgDirListA(HWND,LPSTR,INT,INT,UINT);
3734 INT            WINAPI DlgDirListW(HWND,LPWSTR,INT,INT,UINT);
3735 #define        DlgDirList WINELIB_NAME_AW(DlgDirList)
3736 INT            WINAPI DlgDirListComboBoxA(HWND,LPSTR,INT,INT,UINT);
3737 INT            WINAPI DlgDirListComboBoxW(HWND,LPWSTR,INT,INT,UINT);
3738 #define        DlgDirListComboBox WINELIB_NAME_AW(DlgDirListComboBox)
3739 BOOL           WINAPI DlgDirSelectComboBoxExA(HWND,LPSTR,INT,INT);
3740 BOOL           WINAPI DlgDirSelectComboBoxExW(HWND,LPWSTR,INT,INT);
3741 #define        DlgDirSelectComboBoxEx WINELIB_NAME_AW(DlgDirSelectComboBoxEx)
3742 BOOL           WINAPI DlgDirSelectExA(HWND,LPSTR,INT,INT);
3743 BOOL           WINAPI DlgDirSelectExW(HWND,LPWSTR,INT,INT);
3744 #define        DlgDirSelectEx WINELIB_NAME_AW(DlgDirSelectEx)
3745 BOOL           WINAPI DragDetect(HWND,POINT);
3746 DWORD          WINAPI DragObject(HWND,HWND,UINT,DWORD,HCURSOR);
3747 BOOL           WINAPI DrawAnimatedRects(HWND,int,const RECT*,const RECT*);
3748 BOOL           WINAPI DrawCaption(HWND,HDC,const RECT*,UINT);
3749 BOOL           WINAPI DrawCaptionTempA(HWND,HDC,const RECT*,HFONT,HICON,LPCSTR,UINT);
3750 BOOL           WINAPI DrawCaptionTempW(HWND,HDC,const RECT*,HFONT,HICON,LPCWSTR,UINT);
3751 #define        DrawCaptionTemp WINELIB_NAME_AW(DrawCaptionTemp)
3752 BOOL           WINAPI DrawEdge(HDC,LPRECT,UINT,UINT);
3753 BOOL           WINAPI DrawFocusRect(HDC,const RECT*);
3754 BOOL           WINAPI DrawFrameControl(HDC,LPRECT,UINT,UINT);
3755 BOOL           WINAPI DrawIcon(HDC,INT,INT,HICON);
3756 BOOL           WINAPI DrawIconEx(HDC,INT,INT,HICON,INT,INT,UINT,HBRUSH,UINT);
3757 BOOL           WINAPI DrawMenuBar(HWND);
3758 BOOL           WINAPI DrawStateA(HDC,HBRUSH,DRAWSTATEPROC,LPARAM,WPARAM,INT,INT,INT,INT,UINT);
3759 BOOL           WINAPI DrawStateW(HDC,HBRUSH,DRAWSTATEPROC,LPARAM,WPARAM,INT,INT,INT,INT,UINT);
3760 #define        DrawState WINELIB_NAME_AW(DrawState)
3761 INT            WINAPI DrawTextA(HDC,LPCSTR,INT,LPRECT,UINT);
3762 INT            WINAPI DrawTextW(HDC,LPCWSTR,INT,LPRECT,UINT);
3763 #define        DrawText WINELIB_NAME_AW(DrawText)
3764 INT            WINAPI DrawTextExA(HDC,LPSTR,INT,LPRECT,UINT,LPDRAWTEXTPARAMS);
3765 INT            WINAPI DrawTextExW(HDC,LPWSTR,INT,LPRECT,UINT,LPDRAWTEXTPARAMS);
3766 #define        DrawTextEx WINELIB_NAME_AW(DrawTextEx)
3767 BOOL           WINAPI EmptyClipboard(void);
3768 UINT           WINAPI EnableMenuItem(HMENU,UINT,UINT);
3769 BOOL           WINAPI EnableScrollBar(HWND,INT,UINT);
3770 BOOL           WINAPI EnableWindow(HWND,BOOL);
3771 BOOL           WINAPI EndDeferWindowPos(HDWP);

```



```

3772 BOOL        WINAPI EndDialog (HWND, INT);
3773 BOOL        WINAPI EndPaint (HWND, const PAINTSTRUCT*);
3774 BOOL        WINAPI EnumChildWindows (HWND, WNDENUMPROC, LPARAM);
3775 UINT        WINAPI EnumClipboardFormats (UINT);
3776 INT         WINAPI EnumPropsA (HWND, PROPNUMPROC);
3777 INT         WINAPI EnumPropsW (HWND, PROPNUMPROC);
3778 #define      EnumProps WINELIB_NAME_AW(EnumProps)
3779 BOOL        WINAPI EnumWindows (WNDENUMPROC, LPARAM);
3780 BOOL        WINAPI EnumWindowStationsA (WINSTAENUMPROC, LPARAM);
3781 BOOL        WINAPI EnumWindowStationsW (WINSTAENUMPROC, LPARAM);
3782 #define      EnumWindowStations WINELIB_NAME_AW(EnumWindowStations)
3783 BOOL        WINAPI EqualRect (const RECT*, const RECT*);
3784 INT         WINAPI ExcludeUpdateRgn (HDC, HWND);
3785 #define      ExitWindows(a,b) ExitWindowsEx(EWX_LOGOFF, 0xffffffff)
3786 INT         WINAPI FillRect (HDC, const RECT*, HBRUSH);
3787 HWND        WINAPI FindWindowA (LPCSTR, LPCSTR);
3788 HWND        WINAPI FindWindowW (LPCWSTR, LPCWSTR);
3789 #define      FindWindow WINELIB_NAME_AW(FindWindow)
3790 HWND        WINAPI FindWindowExA (HWND, HWND, LPCSTR, LPCSTR);
3791 HWND        WINAPI FindWindowExW (HWND, HWND, LPCWSTR, LPCWSTR);
3792 #define      FindWindowEx WINELIB_NAME_AW(FindWindowEx)
3793 BOOL        WINAPI FlashWindow (HWND, BOOL);
3794 INT         WINAPI FrameRect (HDC, const RECT*, HBRUSH);
3795 HWND        WINAPI GetActiveWindow (void);
3796 HWND        WINAPI GetAncestor (HWND, UINT);
3797 DWORD       WINAPI GetAppCompatFlags (HTASK);
3798 WORD        WINAPI GetAsyncKeyState (INT);
3799 HWND        WINAPI GetCapture (void);
3800 UINT        WINAPI GetCaretBlinkTime (void);
3801 BOOL        WINAPI GetCaretPos (LPPOINT);
3802 BOOL        WINAPI GetClassInfoA (HINSTANCE, LPCSTR, WNDCLASSA *);
3803 BOOL        WINAPI GetClassInfoW (HINSTANCE, LPCWSTR, WNDCLASSW *);
3804 #define      GetClassInfo WINELIB_NAME_AW(GetClassInfo)
3805 BOOL        WINAPI GetClassInfoExA (HINSTANCE, LPCSTR, WNDCLASSEXA *);
3806 BOOL        WINAPI GetClassInfoExW (HINSTANCE, LPCWSTR, WNDCLASSEXW *);
3807 #define      GetClassInfoEx WINELIB_NAME_AW(GetClassInfoEx)
3808 LONG        WINAPI GetClassLongA (HWND, INT);
3809 LONG        WINAPI GetClassLongW (HWND, INT);
3810 #define      GetClassLong WINELIB_NAME_AW(GetClassLong)
3811 INT         WINAPI GetClassNameA (HWND, LPSTR, INT);
3812 INT         WINAPI GetClassNameW (HWND, LPWSTR, INT);
3813 #define      GetClassName WINELIB_NAME_AW(GetClassName)
3814 WORD        WINAPI GetClassWord (HWND, INT);
3815 BOOL        WINAPI GetClientRect (HWND, LPRECT);
3816 HANDLE       WINAPI GetClipboardData (UINT);
3817 INT         WINAPI GetClipboardFormatNameA (UINT, LPSTR, INT);
3818 INT         WINAPI GetClipboardFormatNameW (UINT, LPWSTR, INT);
3819 #define      GetClipboardFormatName WINELIB_NAME_AW(GetClipboardFormatName)
3820 HWND        WINAPI GetClipboardOwner (void);
3821 HWND        WINAPI GetClipboardViewer (void);
3822 BOOL        WINAPI GetClipCursor (LPRECT);
3823 HCURSOR      WINAPI GetCursor (void);
3824 BOOL        WINAPI GetCursorPos (LPPOINT);
3825 HDC         WINAPI GetDC (HWND);
3826 HDC         WINAPI GetDCEX (HWND, HRGN, DWORD);
3827 HWND        WINAPI GetDesktopWindow (void);
3828 INT         WINAPI GetDlgCtrlID (HWND);
3829 HWND        WINAPI GetDlgItem (HWND, INT);
3830 UINT        WINAPI GetDlgItemInt (HWND, INT, BOOL*, BOOL);
3831 INT         WINAPI GetDlgItemTextA (HWND, INT, LPSTR, UINT);
3832 INT         WINAPI GetDlgItemTextW (HWND, INT, LPWSTR, UINT);
3833 #define      GetDlgItemText WINELIB_NAME_AW(GetDlgItemText)
3834 UINT        WINAPI GetDoubleClickTime (void);
3835 HWND        WINAPI GetFocus (void);
3836 HWND        WINAPI GetForegroundWindow (void);
3837 BOOL        WINAPI GetInputState (void);
3838 UINT        WINAPI GetInternalWindowPos (HWND, LPRECT, LPPOINT);
3839 UINT        WINAPI GetKBCodePage (void);
3840 INT         WINAPI GetKeyboardType (INT);
3841 INT         WINAPI GetKeyNameTextA (LONG, LPSTR, INT);
3842 INT         WINAPI GetKeyNameTextW (LONG, LPWSTR, INT);
3843 #define      GetKeyNameText WINELIB_NAME_AW(GetKeyNameText)
3844 INT         WINAPI GetKeyboardLayoutNameA (LPSTR);
3845 INT         WINAPI GetKeyboardLayoutNameW (LPWSTR);
3846 #define      GetKeyboardLayoutName WINELIB_NAME_AW(GetKeyboardLayoutName)
3847 SHORT       WINAPI GetKeyState (INT);
3848 HWND        WINAPI GetLastActivePopup (HWND);
3849 HMENU        WINAPI GetMenu (HWND);
3850 INT         WINAPI GetMenuItemCount (HMENU);
3851 UINT        WINAPI GetMenuItemID (HMENU, INT);
3852 BOOL        WINAPI GetMenuItemRect (HWND, HMENU, UINT, LPRECT);
3853 UINT        WINAPI GetMenuState (HMENU, UINT, UINT);
3854 INT         WINAPI GetMenuStringA (HMENU, UINT, LPSTR, INT, UINT);
3855 INT         WINAPI GetMenuStringW (HMENU, UINT, LPWSTR, INT, UINT);
3856 #define      GetMenuString WINELIB_NAME_AW(GetMenuString)
3857 BOOL        WINAPI GetMessageA (LPMSG, HWND, UINT, UINT);
3858 BOOL        WINAPI GetMessageW (LPMSG, HWND, UINT, UINT);

```

```

3859 #define      GetMessage WINELIB_NAME_AW(GetMessage)
3860 HWND      WINAPI  GetNextDlgGroupItem(HWND,HWND,BOOL);
3861 HWND      WINAPI  GetNextDlgTabItem(HWND,HWND,BOOL);
3862 #define      GetNextWindow GetWindow
3863 HWND      WINAPI  GetOpenClipboardWindow(void);
3864 HWND      WINAPI  GetParent(HWND);
3865 INT        WINAPI  GetPriorityClipboardFormat(UINT*,INT);
3866 BOOL       WINAPI  GetProcessDefaultLayout(DWORD*);
3867 HANDLE     WINAPI  GetPropA(HWND,LPCSTR);
3868 HANDLE     WINAPI  GetPropW(HWND,LPCWSTR);
3869 #define      GetProp WINELIB_NAME_AW(GetProp)
3870 DWORD      WINAPI  GetQueueStatus(UINT);
3871 BOOL       WINAPI  GetScrollInfo(HWND,INT,LPSCROLLINFO);
3872 INT        WINAPI  GetScrollPos(HWND,INT);
3873 BOOL       WINAPI  GetScrollRange(HWND,INT,LPINT,LPINT);
3874 HWND      WINAPI  GetShellWindow(void);
3875 HMENU      WINAPI  GetSubMenu(HMENU,INT);
3876 HBRUSH     WINAPI  GetSysColorBrush(INT);
3877 #define      GetSysModalWindow() ((HWND)0)
3878 HMENU      WINAPI  GetSystemMenu(HWND,BOOL);
3879 INT        WINAPI  GetSystemMetrics(INT);
3880 DWORD      WINAPI  GetTabbedTextExtentA(HDC,LPCSTR,INT,INT,const INT*);
3881 DWORD      WINAPI  GetTabbedTextExtentW(HDC,LPCWSTR,INT,INT,const INT*);
3882 #define      GetTabbedTextExtent WINELIB_NAME_AW(GetTabbedTextExtent)
3883 HWND      WINAPI  GetTopWindow(HWND);
3884 BOOL       WINAPI  GetUpdateRect(HWND,LPRECT,BOOL);
3885 INT        WINAPI  GetUpdateRgn(HWND,HRGN,BOOL);
3886 BOOL       WINAPI  GetUserObjectInformationA(HANDLE,INT,LPOVOID,DWORD,LPDWORD);
3887 BOOL       WINAPI  GetUserObjectInformationW(HANDLE,INT,LPOVOID,DWORD,LPDWORD);
3888 #define      GetUserObjectInformation WINELIB_NAME_AW(GetUserObjectInformation)
3889 HWND      WINAPI  GetWindow(HWND,UINT);
3890 HDC        WINAPI  GetWindowDC(HWND);
3891 LONG       WINAPI  GetWindowLongA(HWND,INT);
3892 LONG       WINAPI  GetWindowLongW(HWND,INT);
3893 #define      GetWindowLong WINELIB_NAME_AW(GetWindowLong)
3894 BOOL       WINAPI  GetWindowPlacement(HWND,LPWINDOWPLACEMENT);
3895 BOOL       WINAPI  GetWindowRect(HWND,LPRECT);
3896 INT        WINAPI  GetWindowRgn(HWND,HRGN);
3897 HWINSTA    WINAPI  GetProcessWindowStation(void);
3898 #define      GetWindowTask(hwnd) ((HTASK)GetWindowThreadProcessId(hwnd,NULL))
3899 INT        WINAPI  GetWindowTextA(HWND,LPSTR,INT);
3900 INT        WINAPI  GetWindowTextW(HWND,LPWSTR,INT);
3901 #define      GetWindowText WINELIB_NAME_AW(GetWindowText)
3902 INT        WINAPI  GetWindowTextLengthA(HWND);
3903 INT        WINAPI  GetWindowTextLengthW(HWND);
3904 #define      GetWindowTextLength WINELIB_NAME_AW(GetWindowTextLength)
3905 WORD       WINAPI  GetWindowWord(HWND,INT);
3906 BOOL       WINAPI  GrayStringA(HDC,HBRUSH,GRAYSTRINGPROC,LPARAM,
3907                                INT,INT,INT,INT,INT);
3908 BOOL       WINAPI  GrayStringW(HDC,HBRUSH,GRAYSTRINGPROC,LPARAM,
3909                                INT,INT,INT,INT,INT);
3910 #define      GrayString WINELIB_NAME_AW(GrayString)
3911 BOOL       WINAPI  HideCaret(HWND);
3912 BOOL       WINAPI  HiliteMenuItem(HWND,HMENU,UINT,UINT);
3913 BOOL       WINAPI  InflateRect(LPRECT,INT,INT);
3914 BOOL       WINAPI  InSendMessage(void);
3915 DWORD      WINAPI  InSendMessageEx(LPOVOID);
3916 BOOL       WINAPI  InsertMenuA(HMENU,UINT,UINT,UINT,LPCSTR);
3917 BOOL       WINAPI  InsertMenuW(HMENU,UINT,UINT,UINT,LPCWSTR);
3918 #define      InsertMenu WINELIB_NAME_AW(InsertMenu)
3919 BOOL       WINAPI  InsertMenuItemA(HMENU,UINT,BOOL,const MENUITEMINFOA*);
3920 BOOL       WINAPI  InsertMenuItemW(HMENU,UINT,BOOL,const MENUITEMINFOW*);
3921 #define      InsertMenuItem WINELIB_NAME_AW(InsertMenuItem)
3922 INT        WINAPI  InternalGetWindowText(HWND,LPWSTR,INT);
3923 BOOL       WINAPI  IntersectRect(LPRECT,const RECT*,const RECT*);
3924 BOOL       WINAPI  InvalidateRect(HWND,const RECT*,BOOL);
3925 BOOL       WINAPI  InvalidateRgn(HWND,HRGN,BOOL);
3926 BOOL       WINAPI  InvertRect(HDC,const RECT*);
3927 BOOL       WINAPI  IsCharAlphaA(CHAR);
3928 BOOL       WINAPI  IsCharAlphaW(WCHAR);
3929 #define      IsCharAlpha WINELIB_NAME_AW(IsCharAlpha)
3930 BOOL       WINAPI  IsCharAlphaNumericA(CHAR);
3931 BOOL       WINAPI  IsCharAlphaNumericW(WCHAR);
3932 #define      IsCharAlphaNumeric WINELIB_NAME_AW(IsCharAlphaNumeric)
3933 BOOL       WINAPI  IsCharLowerA(CHAR);
3934 BOOL       WINAPI  IsCharLowerW(WCHAR);
3935 #define      IsCharLower WINELIB_NAME_AW(IsCharLower)
3936 BOOL       WINAPI  IsCharUpperA(CHAR);
3937 BOOL       WINAPI  IsCharUpperW(WCHAR);
3938 #define      IsCharUpper WINELIB_NAME_AW(IsCharUpper)
3939 BOOL       WINAPI  IsChild(HWND,HWND);
3940 BOOL       WINAPI  IsClipboardFormatAvailable(UINT);
3941 BOOL       WINAPI  IsDialogMessageA(HWND,LPMMSG);
3942 BOOL       WINAPI  IsDialogMessageW(HWND,LPMMSG);
3943 #define      IsDialogMessage WINELIB_NAME_AW(IsDialogMessage)
3944 UINT       WINAPI  IsDlgButtonChecked(HWND,UINT);
3945 BOOL       WINAPI  IsIconic(HWND);

```

```

3946 BOOL        WINAPI IsMenu(HMENU);
3947 BOOL        WINAPI IsRectEmpty(const RECT*);
3948 BOOL        WINAPI IsWindow(HWND);
3949 BOOL        WINAPI IsWindowEnabled(HWND);
3950 BOOL        WINAPI IsWindowVisible(HWND);
3951 BOOL        WINAPI IsZoomed(HWND);
3952 BOOL        WINAPI KillSystemTimer(HWND, UINT);
3953 BOOL        WINAPI KillTimer(HWND, UINT);
3954 HACCEL       WINAPI LoadAcceleratorsA(HINSTANCE, LPCSTR);
3955 HACCEL       WINAPI LoadAcceleratorsW(HINSTANCE, LPCWSTR);
3956 #define       LoadAccelerators WINELIB_NAME_AW(LoadAccelerators)
3957 HBITMAP       WINAPI LoadBitmapA(HINSTANCE, LPCSTR);
3958 HBITMAP       WINAPI LoadBitmapW(HINSTANCE, LPCWSTR);
3959 #define       LoadBitmap WINELIB_NAME_AW(LoadBitmap)
3960 HCURSOR       WINAPI LoadCursorA(HINSTANCE, LPCSTR);
3961 HCURSOR       WINAPI LoadCursorW(HINSTANCE, LPCWSTR);
3962 #define       LoadCursor WINELIB_NAME_AW(LoadCursor)
3963 HCURSOR       WINAPI LoadCursorFromFileA(LPCSTR);
3964 HCURSOR       WINAPI LoadCursorFromFileW(LPCWSTR);
3965 #define       LoadCursorFromFile WINELIB_NAME_AW(LoadCursorFromFile)
3966 HICON         WINAPI LoadIconA(HINSTANCE, LPCSTR);
3967 HICON         WINAPI LoadIconW(HINSTANCE, LPCWSTR);
3968 #define       LoadIcon WINELIB_NAME_AW(LoadIcon)
3969 HANDLE        WINAPI LoadImageA(HINSTANCE, LPCSTR, UINT, INT, INT, UINT);
3970 HANDLE        WINAPI LoadImageW(HINSTANCE, LPCWSTR, UINT, INT, INT, UINT);
3971 #define       LoadImage WINELIB_NAME_AW(LoadImage)
3972 HMENU         WINAPI LoadMenuA(HINSTANCE, LPCSTR);
3973 HMENU         WINAPI LoadMenuW(HINSTANCE, LPCWSTR);
3974 #define       LoadMenu WINELIB_NAME_AW(LoadMenu)
3975 HMENU         WINAPI LoadMenuIndirectA(LPCVOID);
3976 HMENU         WINAPI LoadMenuIndirectW(LPCVOID);
3977 #define       LoadMenuIndirect WINELIB_NAME_AW(LoadMenuIndirect)
3978 INT           WINAPI LoadStringA(HINSTANCE, UINT, LPSTR, INT);
3979 INT           WINAPI LoadStringW(HINSTANCE, UINT, LPWSTR, INT);
3980 #define       LoadString WINELIB_NAME_AW(LoadString)
3981 BOOL         WINAPI LockWindowUpdate(HWND);
3982 INT           WINAPI LookupIconIdFromDirectory(LPBYTE, BOOL);
3983 INT           WINAPI LookupIconIdFromDirectoryEx(LPBYTE, BOOL, INT, INT, UINT);
3984 UINT          WINAPI MapVirtualKeyA(UINT, UINT);
3985 UINT          WINAPI MapVirtualKeyW(UINT, UINT);
3986 #define       MapVirtualKey WINELIB_NAME_AW(MapVirtualKey)
3987 UINT          WINAPI MapVirtualKeyExA(UINT, UINT, HKL);
3988 UINT          WINAPI MapVirtualKeyExW(UINT, UINT, HKL);
3989 #define       MapVirtualKeyEx WINELIB_NAME_AW(MapVirtualKeyEx)
3990 BOOL         WINAPI MapDialogRect(HWND, LPRECT);
3991 INT           WINAPI MapWindowPoints(HWND, HWND, LPPOINT, UINT);
3992 UINT          WINAPI MenuItemFromPoint(HWND, HMENU, POINT);
3993 BOOL         WINAPI MessageBeep(UINT);
3994 INT           WINAPI MessageBoxA(HWND, LPCSTR, LPCSTR, UINT);
3995 INT           WINAPI MessageBoxW(HWND, LPCWSTR, LPCWSTR, UINT);
3996 #define       MessageBox WINELIB_NAME_AW(MessageBox)
3997 INT           WINAPI MessageBoxIndirectA(LPMSGBOXPARAMSA);
3998 INT           WINAPI MessageBoxIndirectW(LPMSGBOXPARAMSW);
3999 #define       MessageBoxIndirect WINELIB_NAME_AW(MessageBoxIndirect)
4000 BOOL         WINAPI ModifyMenuA(HMENU, UINT, UINT, UINT, LPCSTR);
4001 BOOL         WINAPI ModifyMenuW(HMENU, UINT, UINT, UINT, LPCWSTR);
4002 #define       ModifyMenu WINELIB_NAME_AW(ModifyMenu)
4003 BOOL         WINAPI MoveWindow(HWND, INT, INT, INT, INT, BOOL);
4004 BOOL         WINAPI OemToCharA(LPCSTR, LPSTR);
4005 BOOL         WINAPI OemToCharW(LPCSTR, LPWSTR);
4006 #define       OemToChar WINELIB_NAME_AW(OemToChar)
4007 BOOL         WINAPI OemToCharBuffA(LPCSTR, LPSTR, DWORD);
4008 BOOL         WINAPI OemToCharBuffW(LPCSTR, LPWSTR, DWORD);
4009 #define       OemToCharBuff WINELIB_NAME_AW(OemToCharBuff)
4010 BOOL         WINAPI OffsetRect(LPRECT, INT, INT);
4011 BOOL         WINAPI OpenClipboard(HWND);
4012 BOOL         WINAPI OpenIcon(HWND);
4013 HWINSTA       WINAPI OpenWindowStationA(LPSTR, BOOL, ACCESS_MASK);
4014 HWINSTA       WINAPI OpenWindowStationW(LPWSTR, BOOL, ACCESS_MASK);
4015 #define       OpenWindowStation WINELIB_NAME_AW(OpenWindowStation)
4016 BOOL         WINAPI PeekMessageA(LPMSG, HWND, UINT, UINT, UINT);
4017 BOOL         WINAPI PeekMessageW(LPMSG, HWND, UINT, UINT, UINT);
4018 #define       PeekMessage WINELIB_NAME_AW(PeekMessage)
4019 #define       PostAppMessageA(thread, msg, wparam, lparam) \
4020 PostThreadMessageA((DWORD)(thread), msg, wparam, lparam)
4021 #define       PostAppMessageW(thread, msg, wparam, lparam) \
4022 PostThreadMessageW((DWORD)(thread), msg, wparam, lparam)
4023 #define       PostAppMessage WINELIB_NAME_AW(PostAppMessage)
4024 BOOL         WINAPI PostMessageA(HWND, UINT, WPARAM, LPARAM);
4025 BOOL         WINAPI PostMessageW(HWND, UINT, WPARAM, LPARAM);
4026 #define       PostMessage WINELIB_NAME_AW(PostMessage)
4027 void          WINAPI PostQuitMessage(INT);
4028 BOOL         WINAPI PtInRect(const RECT*, POINT);
4029 BOOL         WINAPI RedrawWindow(HWND, const RECT*, HRGN, UINT);
4030 ATOM          WINAPI RegisterClassA(const WNDCLASSA *);
4031 ATOM          WINAPI RegisterClassW(const WNDCLASSW *);
4032 #define       RegisterClass WINELIB_NAME_AW(RegisterClass)

```

```

4033 ATOM      WINAPI RegisterClassExA(const WNDCLASSEX *);
4034 ATOM      WINAPI RegisterClassExW(const WNDCLASSEX *);
4035 #define      RegisterClassEx WINELIB_NAME_AW(RegisterClassEx)
4036 UINT      WINAPI RegisterClipboardFormatA(LPCSTR);
4037 UINT      WINAPI RegisterClipboardFormatW(LPCWSTR);
4038 #define      RegisterClipboardFormat WINELIB_NAME_AW(RegisterClipboardFormat)
4039 WORD      WINAPI RegisterWindowMessageA(LPCSTR);
4040 WORD      WINAPI RegisterWindowMessageW(LPCWSTR);
4041 #define      RegisterWindowMessage WINELIB_NAME_AW(RegisterWindowMessage)
4042 INT      WINAPI ReleaseDC(HWND, HDC);
4043 BOOL      WINAPI RemoveMenu(HMENU, UINT, UINT);
4044 HANDLE     WINAPI RemovePropA(HWND, LPCSTR);
4045 HANDLE     WINAPI RemovePropW(HWND, LPCWSTR);
4046 #define      RemoveProp WINELIB_NAME_AW(RemoveProp)
4047 BOOL      WINAPI ReplyMessage(LRESULT);
4048 BOOL      WINAPI ScreenToClient(HWND, LPPOINT);
4049 VOID      WINAPI ScrollChildren(HWND, UINT, WPARAM, LPARAM);
4050 BOOL      WINAPI ScrollDC(HDC, INT, INT, const RECT*, const RECT*, HRGN, LPRECT);
4051 BOOL      WINAPI ScrollWindow(HWND, INT, INT, const RECT*, const RECT*);
4052 INT      WINAPI ScrollWindowEx(HWND, INT, INT, const RECT*, const RECT*, HRGN, LPRECT, UINT);
4053 LRESULT    WINAPI SendDlgItemMessageA(HWND, INT, UINT, WPARAM, LPARAM);
4054 LRESULT    WINAPI SendDlgItemMessageW(HWND, INT, UINT, WPARAM, LPARAM);
4055 #define      SendDlgItemMessage WINELIB_NAME_AW(SendDlgItemMessage)
4056 UINT      WINAPI SendInput(UINT, LPINPUT, int);
4057 LRESULT    WINAPI SendMessageA(HWND, UINT, WPARAM, LPARAM);
4058 LRESULT    WINAPI SendMessageW(HWND, UINT, WPARAM, LPARAM);
4059 #define      SendMessage WINELIB_NAME_AW(SendMessage)
4060 LRESULT    WINAPI SendMessageTimeoutA(HWND, UINT, WPARAM, LPARAM, UINT, UINT, LPDWORD);
4061 LRESULT    WINAPI SendMessageTimeoutW(HWND, UINT, WPARAM, LPARAM, UINT, UINT, LPDWORD);
4062 #define      SendMessageTimeout WINELIB_NAME_AW(SendMessageTimeout)
4063 HWND      WINAPI SetActiveWindow(HWND);
4064 HWND      WINAPI SetCapture(HWND);
4065 BOOL      WINAPI SetCaretBlinkTime(UINT);
4066 BOOL      WINAPI SetCaretPos(INT, INT);
4067 LONG      WINAPI SetClassLongA(HWND, INT, LONG);
4068 LONG      WINAPI SetClassLongW(HWND, INT, LONG);
4069 #define      SetClassLong WINELIB_NAME_AW(SetClassLong)
4070 WORD      WINAPI SetClassWord(HWND, INT, WORD);
4071 HANDLE     WINAPI SetClipboardData(UINT, HANDLE);
4072 HWND      WINAPI SetClipboardViewer(HWND);
4073 HCURSOR    WINAPI SetCursor(HCURSOR);
4074 BOOL      WINAPI SetCursorPos(INT, INT);
4075 BOOL      WINAPI SetDeskWallPaper(LPCSTR);
4076 BOOL      WINAPI SetDlgItemInt(HWND, INT, UINT, BOOL);
4077 BOOL      WINAPI SetDlgItemTextA(HWND, INT, LPCSTR);
4078 BOOL      WINAPI SetDlgItemTextW(HWND, INT, LPCWSTR);
4079 #define      SetDlgItemText WINELIB_NAME_AW(SetDlgItemText)
4080 BOOL      WINAPI SetDoubleClickTime(UINT);
4081 HWND      WINAPI SetFocus(HWND);
4082 BOOL      WINAPI SetForegroundWindow(HWND);
4083 void      WINAPI SetInternalWindowPos(HWND, UINT, LPRECT, LPPOINT);
4084 BOOL      WINAPI SetMenu(HWND, HMENU);
4085 BOOL      WINAPI SetMenuContextHelpId(HMENU, DWORD);
4086 BOOL      WINAPI SetMenuItemBitmaps(HMENU, UINT, UINT, HBITMAP, HBITMAP);
4087 BOOL      WINAPI SetMessageQueue(INT);
4088 BOOL      WINAPI SetProcessDefaultLayout(DWORD);
4089 BOOL      WINAPI SetProcessWindowStation(HWINSTA);
4090 HWND      WINAPI SetParent(HWND, HWND);
4091 BOOL      WINAPI SetPropA(HWND, LPCSTR, HANDLE);
4092 BOOL      WINAPI SetPropW(HWND, LPCWSTR, HANDLE);
4093 #define      SetProp WINELIB_NAME_AW(SetProp)
4094 BOOL      WINAPI SetRect(LPRECT, INT, INT, INT, INT);
4095 BOOL      WINAPI SetRectEmpty(LPRECT);
4096 INT      WINAPI SetScrollInfo(HWND, INT, const SCROLLINFO*, BOOL);
4097 INT      WINAPI SetScrollPos(HWND, INT, INT, BOOL);
4098 BOOL      WINAPI SetScrollRange(HWND, INT, INT, INT, BOOL);
4099 #define      SetSysModalWindow(hwnd) ((HWND)0)
4100 BOOL      WINAPI SetSystemCursor(HCURSOR, DWORD);
4101 BOOL      WINAPI SetSystemMenu(HWND, HMENU);
4102 UINT      WINAPI SetSystemTimer(HWND, UINT, UINT, TIMERPROC);
4103 UINT      WINAPI SetTimer(HWND, UINT, UINT, TIMERPROC);
4104 BOOL      WINAPI SetUserObjectSecurity(HANDLE, PSECURITY_INFORMATION, PSECURITY_DESCRIPTOR);
4105 LONG      WINAPI SetWindowLongA(HWND, INT, LONG);
4106 LONG      WINAPI SetWindowLongW(HWND, INT, LONG);
4107 #define      SetWindowLong WINELIB_NAME_AW(SetWindowLong)
4108 BOOL      WINAPI SetWindowPlacement(HWND, const WINDOWPLACEMENT*);
4109 HHOOK      WINAPI SetWindowsHookA(INT, HOOKPROC);
4110 HHOOK      WINAPI SetWindowsHookW(INT, HOOKPROC);
4111 #define      SetWindowsHook WINELIB_NAME_AW(SetWindowsHook)
4112 HHOOK      WINAPI SetWindowsHookExA(INT, HOOKPROC, HINSTANCE, DWORD);
4113 HHOOK      WINAPI SetWindowsHookExW(INT, HOOKPROC, HINSTANCE, DWORD);
4114 #define      SetWindowsHookEx WINELIB_NAME_AW(SetWindowsHookEx)
4115 BOOL      WINAPI SetWindowPos(HWND, HWND, INT, INT, INT, INT, UINT);
4116 INT      WINAPI SetWindowRgn(HWND, HRGN, BOOL);
4117 BOOL      WINAPI SetWindowTextA(HWND, LPCSTR);
4118 BOOL      WINAPI SetWindowTextW(HWND, LPCWSTR);
4119 #define      SetWindowText WINELIB_NAME_AW(SetWindowText)

```

```

4120 WORD          WINAPI SetWindowWord (HWND, INT, WORD);
4121 BOOL          WINAPI ShowCaret (HWND);
4122 INT           WINAPI ShowCursor (BOOL);
4123 BOOL          WINAPI ShowScrollBar (HWND, INT, BOOL);
4124 BOOL          WINAPI ShowOwnedPopups (HWND, BOOL);
4125 BOOL          WINAPI ShowWindow (HWND, INT);
4126 BOOL          WINAPI SubtractRect (LPRECT, const RECT*, const RECT*);
4127 BOOL          WINAPI SwapMouseButton (BOOL);
4128 VOID          WINAPI SwitchToThisWindow (HWND, BOOL);
4129 BOOL          WINAPI SystemParametersInfoA (UINT, UINT, LPVOID, UINT);
4130 BOOL          WINAPI SystemParametersInfoW (UINT, UINT, LPVOID, UINT);
4131 #define         SystemParametersInfo WINELIB_NAME_AW (SystemParametersInfo)
4132 LONG          WINAPI TabbedTextOutA (HDC, INT, INT, LPCSTR, INT, INT, const INT*, INT);
4133 LONG          WINAPI TabbedTextOutW (HDC, INT, INT, LPCWSTR, INT, INT, const INT*, INT);
4134 #define         TabbedTextOut WINELIB_NAME_AW (TabbedTextOut)
4135 INT           WINAPI ToAscii (UINT, UINT, LPBYTE, LPWORD, UINT);
4136 INT           WINAPI ToAsciiEx (UINT, UINT, LPBYTE, LPWORD, UINT, HKL);
4137 BOOL          WINAPI TrackPopupMenu (HMENU, UINT, INT, INT, INT, HWND, const RECT*);
4138 INT           WINAPI TranslateAccelerator (HWND, HACCEL, LPMSG);
4139 BOOL          WINAPI TranslateMDISysAccel (HWND, LPMSG);
4140 BOOL          WINAPI TranslateMessage (const MSG*);
4141 BOOL          WINAPI UnhookWindowsHook (INT, HOOKPROC);
4142 BOOL          WINAPI UnhookWindowsHookEx (HHOOK);
4143 BOOL          WINAPI UnionRect (LPRECT, const RECT*, const RECT*);
4144 BOOL          WINAPI UnregisterClassA (LPCSTR, HINSTANCE);
4145 BOOL          WINAPI UnregisterClassW (LPCWSTR, HINSTANCE);
4146 #define         UnregisterClass WINELIB_NAME_AW (UnregisterClass)
4147 VOID          WINAPI UpdateWindow (HWND);
4148 UINT          WINAPI UserRealizePalette (HDC);
4149 VOID          WINAPI ValidateRect (HWND, const RECT*);
4150 VOID          WINAPI ValidateRgn (HWND, HRGN);
4151 WORD          WINAPI VkKeyScanA (CHAR);
4152 WORD          WINAPI VkKeyScanW (WCHAR);
4153 #define         VkKeyScan WINELIB_NAME_AW (VkKeyScan)
4154 WORD          WINAPI VkKeyScanExA (CHAR, HKL);
4155 WORD          WINAPI VkKeyScanExW (WCHAR, HKL);
4156 #define         VkKeyScanEx WINELIB_NAME_AW (VkKeyScanEx)
4157 BOOL          WINAPI WaitMessage (void);
4158 HWND          WINAPI WindowFromDC (HDC);
4159 HWND          WINAPI WindowFromPoint (POINT);
4160 BOOL          WINAPI WinHelpA (HWND, LPCSTR, UINT, DWORD);
4161 BOOL          WINAPI WinHelpW (HWND, LPCWSTR, UINT, DWORD);
4162 #define         WinHelp WINELIB_NAME_AW (WinHelp)
4163 INT           WINAPI wprintfA (LPSTR, LPCSTR, ...);
4164 INT           WINAPI wprintfW (LPWSTR, LPCWSTR, ...);
4165 #define         wprintf WINELIB_NAME_AW (wprintf)
4166 INT           WINAPI wvprintfA (LPSTR, LPCSTR, va_list);
4167 INT           WINAPI wvprintfW (LPWSTR, LPCWSTR, va_list);
4168 #define         wvprintf WINELIB_NAME_AW (wvprintf)
4169
4170 /* Undocumented functions */
4171
4172 /* NOTE: This is SYSTEM.3, not USER.182, which is also named KillSystemTimer */
4173 WORD          WINAPI SYSTEM_KillSystemTimer (WORD);
4174
4175 HRESULT        WINAPI PrivateExtractIconsA (LPCSTR, INT, DWORD, DWORD, HICON*, DWORD, UINT, DWORD);
4176 HRESULT        WINAPI PrivateExtractIconsW (LPCWSTR, INT, DWORD, DWORD, HICON*, DWORD, UINT, DWORD);
4177
4178 /* Extra functions that don't exist in the Windows API */
4179
4180 HPEN          WINAPI GetSysColorPen (INT);
4181 INT           WINAPI wvsnprintfA (LPSTR, UINT, LPCSTR, va_list);
4182 INT           WINAPI wvsnprintfW (LPWSTR, UINT, LPCWSTR, va_list);
4183 #define         wvsnprintf WINELIB_NAME_AW (wvsnprintf)
4184
4185 #ifdef __cplusplus
4186 }
4187 #endif
4188
4189 #endif /* _WINUSER_ */

```

5.14 libemf.h

```

1 /* -*- c++ -*-
2 * EMF: A library for generating ECMA-234 Enhanced Metafiles
3 * Copyright (C) 2002, 2003 lignum Computing, Inc. <dallenbarnett@users.sourceforge.net>
4 * $Id: libemf.h 94 2020-04-25 18:46:06Z dallenbarnett $
5 *
6 * This library is free software; you can redistribute it and/or
7 * modify it under the terms of the GNU Lesser General Public
8 * License as published by the Free Software Foundation; either
9 * version 2.1 of the License, or (at your option) any later version.
10 *
11 * This library is distributed in the hope that it will be useful,

```

```

12 * but WITHOUT ANY WARRANTY; without even the implied warranty of
13 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
14 * Lesser General Public License for more details.
15 *
16 * You should have received a copy of the GNU Lesser General Public
17 * License along with this library; if not, write to the Free Software
18 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
19 *
20 */
21 #ifndef _LIBEMF_H
22 #define _LIBEMF_H 1
23
24 #include <cmath>
25 #include <vector>
26 #include <map>
27 #include <functional>
28 #include <algorithm>
29 #include <stdexcept>
30 #include <memory>
31
32 #include <config.h>
33 #include <libEMF/emf.h>
34
35 #include <libEMF/wine/w16.h>
36
37 #ifdef ENABLE_EDITING
38 #include <iconv.h>
39 #include <errno.h>
40 #endif
41
42 #define EMF_UNUSED(x) (void)x;
43
44 namespace EMF {
45 #if 1
46     const int XMAX_PIXELS = 1024; /*(INT_MAX)*/
47 #else
48     const int XMAX_PIXELS = 1280; /*(INT_MAX)*/
49 #endif
50 #if 1
51     const int YMAX_PIXELS = 768; /*(INT_MAX)*/
52 #else
53     const int YMAX_PIXELS = 1024; /*(INT_MAX)*/
54 #endif
55     const int XMAX_MM = 320;
56     const int YMAX_MM = 240;
57     const int RESOLUTION = 96;
58     static inline DWORD ROUND_TO_LONG ( DWORD n ) { return ((n+3)/4)*4; }
59
60     static bool bigEndian ( void );
61
62     struct WCHARSTR {
63         WCHAR *const string_;
64         const int length_;
65         WCHARSTR ( WCHAR *const string, const int length )
66             : string_( string ), length_( length ) {}
67     };
68
69     struct CHARSTR {
70         CHAR *const string_;
71         const int length_;
72         CHARSTR ( CHAR *const string, const int length )
73             : string_( string ), length_( length ) {}
74     };
75
76     struct BYTEARRAY {
77         BYTE *const array_;
78         const int n_;
79         BYTEARRAY ( BYTE *const array, const int n )
80             : array_( array ), n_( n ) {}
81     };
82
83     struct POINTLARRAY {
84         POINTL *const points_;
85         const DWORD n_;
86         POINTLARRAY ( POINTL *const points, const DWORD n )
87             : points_( points ), n_( n ) {}
88     };
89
90     struct POINT16ARRAY {
91         POINT16 *const points_;
92         const DWORD n_;
93         POINT16ARRAY ( POINT16 *const points, const DWORD n )

```

```

168     : points_( points ), n_( n ) {}
169 };
170
171
172
173 struct INTARRAY {
174     INT *const ints_;
175     const DWORD n_;
176     INTARRAY ( INT *const ints, const DWORD n )
177     : ints_( ints ), n_( n ) {}
178 };
179
180
181 struct DWORDARRAY {
182     DWORD *const dwords_;
183     const DWORD n_;
184     DWORDARRAY ( DWORD *const dwords, const DWORD n )
185     : dwords_( dwords ), n_( n ) {}
186 };
187
188
189 struct PADDING {
190     static const char padding_[4];
191     const int size_;
192     PADDING ( const int size ) : size_( size ) {}
193 };
194
195
196 class DATASTREAM {
197     bool swap_;
198     ::FILE* fp_;
199 public:
200     DATASTREAM ( ::FILE* fp = 0 ) : swap_( bigEndian() ), fp_( fp ) {}
201     void setStream ( ::FILE* fp ) { fp_ = fp; }
202     DATASTREAM& operator<< ( const BYTE& byte )
203     {
204         fwrite( &byte, sizeof(BYTE), 1, fp_ );
205         return *this;
206     }
207     DATASTREAM& operator>> ( BYTE& byte )
208     {
209         fread( &byte, sizeof(BYTE), 1, fp_ );
210         return *this;
211     }
212     DATASTREAM& operator<< ( const WORD& word )
213     {
214         if ( swap_ ) {
215             unsigned char const * p = (unsigned char const*)&word;
216             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
217             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
218         }
219         else
220             fwrite( &word, sizeof(WORD), 1, fp_ );
221         return *this;
222     }
223     DATASTREAM& operator>> ( WORD& word )
224     {
225         if ( swap_ ) {
226             unsigned char* p = (unsigned char*)&word;
227             fread( &p[1], sizeof(unsigned char), 1, fp_ );
228             fread( &p[0], sizeof(unsigned char), 1, fp_ );
229         }
230         else
231             fread( &word, sizeof(WORD), 1, fp_ );
232         return *this;
233     }
234     DATASTREAM& operator<< ( const INT16& word )
235     {
236         if ( swap_ ) {
237             unsigned char const * p = (unsigned char const*)&word;
238             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
239             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
240         }
241         else
242             fwrite( &word, sizeof(INT16), 1, fp_ );
243         return *this;
244     }
245     DATASTREAM& operator>> ( INT16& word )
246     {
247         if ( swap_ ) {
248             unsigned char* p = (unsigned char*)&word;
249             fread( &p[1], sizeof(unsigned char), 1, fp_ );
250             fread( &p[0], sizeof(unsigned char), 1, fp_ );
251         }
252         else
253             fread( &word, sizeof(INT16), 1, fp_ );
254         return *this;
255     }
256 }

```



```

322     DATASTREAM& operator<< ( const DWORD& dword )
323     {
324         if ( swap_ ) {
325             unsigned char const* p = (unsigned char const*)&dword;
326             fwrite( &p[3], sizeof(unsigned char), 1, fp_ );
327             fwrite( &p[2], sizeof(unsigned char), 1, fp_ );
328             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
329             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
330         }
331         else
332             fwrite( &dword, sizeof(DWORD), 1, fp_ );
333         return *this;
334     }
335     DATASTREAM& operator>> ( DWORD& dword )
336     {
337         if ( swap_ ) {
338             unsigned char* p = (unsigned char*)&dword;
339             fread( &p[3], sizeof(unsigned char), 1, fp_ );
340             fread( &p[2], sizeof(unsigned char), 1, fp_ );
341             fread( &p[1], sizeof(unsigned char), 1, fp_ );
342             fread( &p[0], sizeof(unsigned char), 1, fp_ );
343         }
344         else
345             fread( &dword, sizeof(DWORD), 1, fp_ );
346         return *this;
347     }
348 #if !defined( __LP64__ )
349     DATASTREAM& operator<< ( const LONG& long_ )
350     {
351         if ( swap_ ) {
352             unsigned char const* p = (unsigned char const*)&long_;
353             fwrite( &p[3], sizeof(unsigned char), 1, fp_ );
354             fwrite( &p[2], sizeof(unsigned char), 1, fp_ );
355             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
356             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
357         }
358         else
359             fwrite( &long_, sizeof(LONG), 1, fp_ );
360         return *this;
361     }
362     DATASTREAM& operator>> ( LONG& long_ )
363     {
364         if ( swap_ ) {
365             unsigned char* p = (unsigned char*)&long_;
366             fread( &p[3], sizeof(unsigned char), 1, fp_ );
367             fread( &p[2], sizeof(unsigned char), 1, fp_ );
368             fread( &p[1], sizeof(unsigned char), 1, fp_ );
369             fread( &p[0], sizeof(unsigned char), 1, fp_ );
370         }
371         else
372             fread( &long_, sizeof(LONG), 1, fp_ );
373         return *this;
374     }
375 #endif /* __x86_64__ */
376     DATASTREAM& operator<< ( const INT& int_ )
377     {
378         if ( swap_ ) {
379             unsigned char const* p = (unsigned char const*)&int_;
380             fwrite( &p[3], sizeof(unsigned char), 1, fp_ );
381             fwrite( &p[2], sizeof(unsigned char), 1, fp_ );
382             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
383             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
384         }
385         else
386             fwrite( &int_, sizeof(INT), 1, fp_ );
387         return *this;
388     }
389     DATASTREAM& operator>> ( INT& int_ )
390     {
391         if ( swap_ ) {
392             unsigned char* p = (unsigned char*)&int_;
393             fread( &p[3], sizeof(unsigned char), 1, fp_ );
394             fread( &p[2], sizeof(unsigned char), 1, fp_ );
395             fread( &p[1], sizeof(unsigned char), 1, fp_ );
396             fread( &p[0], sizeof(unsigned char), 1, fp_ );
397         }
398         else
399             fread( &int_, sizeof(INT), 1, fp_ );
400         return *this;
401     }
402 #if !defined( __LP64__ )
403     DATASTREAM& operator<< ( const UINT& uint )
404     {
405         if ( swap_ ) {
406             unsigned char const* p = (unsigned char const*)&uint;
407             fwrite( &p[3], sizeof(unsigned char), 1, fp_ );
408             fwrite( &p[2], sizeof(unsigned char), 1, fp_ );

```



```

433     fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
434     fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
435     }
436     else
437     fwrite( &uint, sizeof(UINT), 1, fp_ );
438     return *this;
439     }
440     DATASTREAM& operator<> ( UINT& uint )
441     {
442         if ( swap_ ) {
443             unsigned char* p = (unsigned char*)&uint;
444             fread( &p[3], sizeof(unsigned char), 1, fp_ );
445             fread( &p[2], sizeof(unsigned char), 1, fp_ );
446             fread( &p[1], sizeof(unsigned char), 1, fp_ );
447             fread( &p[0], sizeof(unsigned char), 1, fp_ );
448         }
449         else
450             fread( &uint, sizeof(UINT), 1, fp_ );
451         return *this;
452     }
453 #endif /* !_x86_64__ */
454     DATASTREAM& operator<> ( const FLOAT& float_ )
455     {
456         if ( swap_ ) {
457             unsigned char const* p = (unsigned char const*)&float_;
458             fwrite( &p[3], sizeof(unsigned char), 1, fp_ );
459             fwrite( &p[2], sizeof(unsigned char), 1, fp_ );
460             fwrite( &p[1], sizeof(unsigned char), 1, fp_ );
461             fwrite( &p[0], sizeof(unsigned char), 1, fp_ );
462         }
463         else
464             fwrite( &float_, sizeof(FLOAT), 1, fp_ );
465         return *this;
466     }
467     DATASTREAM& operator<> ( FLOAT& float_ )
468     {
469         if ( swap_ ) {
470             unsigned char* p = (unsigned char*)&float_;
471             fread( &p[3], sizeof(unsigned char), 1, fp_ );
472             fread( &p[2], sizeof(unsigned char), 1, fp_ );
473             fread( &p[1], sizeof(unsigned char), 1, fp_ );
474             fread( &p[0], sizeof(unsigned char), 1, fp_ );
475         }
476         else
477             fread( &float_, sizeof(FLOAT), 1, fp_ );
478         return *this;
479     }
480     DATASTREAM& operator<> ( const PADDING& padding )
481     {
482         if ( padding.size_ != 0 )
483             fwrite( &padding.padding_, sizeof(CHAR), padding.size_, fp_ );
484         return *this;
485     }
486     DATASTREAM& operator<> ( const RECTL& rectl )
487     {
488         *this << rectl.left << rectl.top << rectl.right << rectl.bottom;
489         return *this;
490     }
491     DATASTREAM& operator<> ( RECTL& rectl )
492     {
493         *this >> rectl.left >> rectl.top >> rectl.right >> rectl.bottom;
494         return *this;
495     }
496     DATASTREAM& operator<> ( const SIZEL& sizel )
497     {
498         *this << sizel.cx << sizel.cy;
499         return *this;
500     }
501     DATASTREAM& operator<> ( SIZEL& sizel )
502     {
503         *this >> sizel.cx >> sizel.cy;
504         return *this;
505     }
506     DATASTREAM& operator<> ( const WCHARSTR& wcharstr )
507     {
508         for ( int i = 0; i < wcharstr.length_; i++ )
509             *this << wcharstr.string_[i];
510         return *this;
511     }
512     DATASTREAM& operator<> ( WCHARSTR& wcharstr )
513     {
514         for ( int i = 0; i < wcharstr.length_; i++ )
515             *this >> wcharstr.string_[i];
516         return *this;
517     }
518     DATASTREAM& operator<> ( const CHARSTR& charstr )
519     {

```

```

564     fwrite( charstr.string_, sizeof(CHAR), charstr.length_, fp_ );
565     return *this;
566 }
571 DATASTREAM& operator< ( CHARSTR& charstr )
572 {
573     fread( charstr.string_, sizeof(CHAR), charstr.length_, fp_ );
574     return *this;
575 }
580 DATASTREAM& operator< ( const ::EMR& emr )
581 {
582     *this < emr.iType < emr.nSize;
583     return *this;
584 }
589 DATASTREAM& operator< ( ::EMR& emr )
590 {
591     *this > emr.iType > emr.nSize;
592     return *this;
593 }
598 DATASTREAM& operator< ( const POINT& point )
599 {
600     *this < point.x < point.y;
601     return *this;
602 }
607 DATASTREAM& operator< ( POINT& point )
608 {
609     *this > point.x > point.y;
610     return *this;
611 }
616 DATASTREAM& operator< ( const POINTL& pointl )
617 {
618     *this < pointl.x < pointl.y;
619     return *this;
620 }
625 DATASTREAM& operator< ( POINTL& pointl )
626 {
627     *this > pointl.x > pointl.y;
628     return *this;
629 }
634 DATASTREAM& operator< ( const POINT16& point )
635 {
636     *this < point.x < point.y;
637     return *this;
638 }
643 DATASTREAM& operator< ( POINT16& point )
644 {
645     *this > point.x > point.y;
646     return *this;
647 }
652 DATASTREAM& operator< ( const XFORM& xform )
653 {
654     *this < xform.eM11 < xform.eM12 < xform.eM21 < xform.eM22
655     < xform.eDx < xform.eDy;
656     return *this;
657 }
662 DATASTREAM& operator< ( XFORM& xform )
663 {
664     *this > xform.eM11 > xform.eM12 > xform.eM21 > xform.eM22
665     > xform.eDx > xform.eDy;
666     return *this;
667 }
672 DATASTREAM& operator< ( const BYTEARRAY& array )
673 {
674     fwrite( array.array_, sizeof(BYTE), array.n_, fp_ );
675     return *this;
676 }
681 DATASTREAM& operator< ( BYTEARRAY& array )
682 {
683     fread( array.array_, sizeof(BYTE), array.n_, fp_ );
684     return *this;
685 }
690 DATASTREAM& operator< ( const POINTLARRAY& array )
691 {
692     for ( unsigned int i = 0; i < array.n_; i++ )
693         *this < array.points_[i];
694     return *this;
695 }
700 DATASTREAM& operator< ( POINTLARRAY& array )
701 {
702     for ( unsigned int i = 0; i < array.n_; i++ )
703         *this > array.points_[i];
704     return *this;
705 }
710 DATASTREAM& operator< ( const POINT16ARRAY& array )
711 {
712     for ( unsigned int i = 0; i < array.n_; i++ )
713         *this < array.points_[i];
714     return *this;

```

```

715     }
720     DATASTREAM& operator<> ( POINT16ARRAY& array )
721     {
722         for ( unsigned int i = 0; i < array.n_; i++ )
723             *this >> array.points_[i];
724         return *this;
725     }
730     DATASTREAM& operator<> ( const INTARRAY& array )
731     {
732         for ( unsigned int i = 0; i < array.n_; i++ )
733             *this << array.ints_[i];
734         return *this;
735     }
740     DATASTREAM& operator<> ( INTARRAY& array )
741     {
742         for ( unsigned int i = 0; i < array.n_; i++ )
743             *this >> array.ints_[i];
744         return *this;
745     }
750     DATASTREAM& operator<> ( const DWORDARRAY& array )
751     {
752         for ( unsigned int i = 0; i < array.n_; i++ )
753             *this << array.dwords_[i];
754         return *this;
755     }
760     DATASTREAM& operator<> ( DWORDARRAY& array )
761     {
762         for ( unsigned int i = 0; i < array.n_; i++ )
763             *this >> array.dwords_[i];
764         return *this;
765     }
770     DATASTREAM& operator<> ( const ::EMRTEXT& text )
771     {
772         *this << text.ptlReference << text.nChars << text.offString << text.fOptions
773         << text.rc1 << text.offDx;
774         return *this;
775     }
780     DATASTREAM& operator<> ( ::EMRTEXT& text )
781     {
782         *this >> text.ptlReference >> text.nChars >> text.offString >> text.fOptions
783         >> text.rc1 >> text.offDx;
784         return *this;
785     }
790     DATASTREAM& operator<> ( const LOGPEN& pen )
791     {
792         *this << pen.lopnStyle << pen.lopnWidth << pen.lopnColor;
793         return *this;
794     }
799     DATASTREAM& operator<> ( LOGPEN& pen )
800     {
801         *this >> pen.lopnStyle >> pen.lopnWidth >> pen.lopnColor;
802         return *this;
803     }
808     DATASTREAM& operator<> ( const EXTLOGPEN& pen )
809     {
810         // *** How big is this structure if there are no style entries? ***
811         *this << pen.elpPenStyle << pen.elpWidth << pen.elpBrushStyle << pen.elpColor
812         << pen.elpHatch << pen.elpNumEntries;
813         return *this;
814     }
819     DATASTREAM& operator<> ( EXTLOGPEN& pen )
820     {
821         // *** How big is this structure if there are no style entries? ***
822         *this >> pen.elpPenStyle >> pen.elpWidth >> pen.elpBrushStyle >> pen.elpColor
823         >> pen.elpHatch >> pen.elpNumEntries;
824         return *this;
825     }
830     DATASTREAM& operator<> ( const LOGBRUSH& brush )
831     {
832         *this << brush.lbStyle << brush.lbColor << brush.lbHatch;
833         return *this;
834     }
839     DATASTREAM& operator<> ( LOGBRUSH& brush )
840     {
841         *this >> brush.lbStyle >> brush.lbColor >> brush.lbHatch;
842         return *this;
843     }
848     DATASTREAM& operator<> ( const LOGFONTW& font )
849     {
850         *this << font.lfHeight << font.lfWidth << font.lfEscapement
851         << font.lfOrientation << font.lfWeight << font.lfItalic
852         << font.lfUnderline << font.lfStrikeOut << font.lfCharSet
853         << font.lfOutPrecision << font.lfClipPrecision << font.lfQuality
854         << font.lfPitchAndFamily
855         << WCHARSTR( const_cast<WCHAR*>(font.lfFaceName), LF_FACESIZE );
856         return *this;
857     }

```

```

862 DATASTREAM& operator» ( LOGFONTW& font )
863 {
864     WCHARSTR wFaceName( font.lfFaceName, LF_FACESIZE );
865
866     *this » font.lfHeight » font.lfWidth » font.lfEscapement
867     » font.lfOrientation » font.lfWeight » font.lfItalic
868     » font.lfUnderline » font.lfStrikeOut » font.lfCharSet
869     » font.lfOutPrecision » font.lfClipPrecision » font.lfQuality
870     » font.lfPitchAndFamily
871     » wFaceName;
872     return *this;
873 }
874 DATASTREAM& operator« ( const PANOSE& panose )
875 {
876     fwrite( &panose, sizeof(PANOSE), 1, fp_ );
877     return *this;
878 }
879 DATASTREAM& operator» ( PANOSE& panose )
880 {
881     fread( &panose, sizeof(PANOSE), 1, fp_ );
882     return *this;
883 }
884 DATASTREAM& operator« ( const EXTLOGFONTW& font )
885 {
886     *this « font.elfLogFont
887     « WCHARSTR( const_cast<WCHAR*>(font.elfFullName),
888     LF_FULLFACESIZE )
889     « WCHARSTR( const_cast<WCHAR*>(font.elfStyle), LF_FACESIZE )
890     « font.elfVersion « font.elfStyleSize « font.elfMatch
891     « font.elfReserved
892     « BYTEARRAY( const_cast<BYTE*>(font.elfVendorId),
893     ELF_VENDOR_SIZE )
894     « font.elfCulture « font.elfPanose;
895     return *this;
896 }
897 DATASTREAM& operator» ( EXTLOGFONTW& font )
898 {
899     WCHARSTR wFullName( font.elfFullName, LF_FULLFACESIZE );
900     WCHARSTR wStyle( font.elfStyle, LF_FACESIZE );
901     BYTEARRAY bVendorId( font.elfVendorId, ELF_VENDOR_SIZE );
902     *this » font.elfLogFont
903     » wFullName » wStyle
904     » font.elfVersion » font.elfStyleSize » font.elfMatch
905     » font.elfReserved » bVendorId
906     » font.elfCulture » font.elfPanose;
907     return *this;
908 }
909 DATASTREAM& operator« ( const LOGPALETTE& palette )
910 {
911     // *** How big is this structure if the palette is empty? ***
912     *this « palette.palVersion « palette.palNumEntries;
913     return *this;
914 }
915 DATASTREAM& operator» ( LOGPALETTE& palette )
916 {
917     // *** How big is this structure if the palette is empty? ***
918     *this » palette.palVersion » palette.palNumEntries;
919     return *this;
920 }
921 private:
922 void fread ( void* ptr, size_t size, size_t nmemb, FILE* stream )
923 {
924     size_t res = ::fread( ptr, size, nmemb, stream );
925     if ( res < nmemb ) {
926         throw std::runtime_error( "Premature EOF on EMF stream" );
927     }
928 }
929 void fwrite ( const void* ptr, size_t size, size_t nmemb, FILE* stream )
930 {
931     size_t res = ::fwrite( ptr, size, nmemb, stream );
932     if ( res < nmemb ) {
933         throw std::runtime_error( "error writing EMF stream" );
934     }
935 }
936 };
937
938 class METAFILEDEVICECONTEXT;
939
940 class METARECORD {
941 public:
942     virtual void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const = 0;
943     virtual bool serialize ( DATASTREAM ds ) = 0;
944     virtual int size ( void ) const = 0;
945     virtual ~METARECORD ( ) { }
946 #ifdef ENABLE_EDITING
947     virtual void edit ( void ) const { }
948 #endif
949 };

```

```

1023 #endif
1024 };
1025
1026 #ifdef ENABLE_EDITING
1027 /* Miscellaneous editing routines */
1028 inline void edit_rectl ( const char* tag, const RECTL& rectl )
1029 {
1030     #if defined(__LP64__)
1031         const char* FMT = "\\t%s\\t: (%d, %d) - (%d, %d)\\n";
1032     #else
1033         const char* FMT = "\\t%s\\t: (%ld, %ld) - (%ld, %ld)\\n";
1034     #endif /* __x86_64__ */
1035     printf( FMT, tag, rectl.left, rectl.top, rectl.right, rectl.bottom );
1036 }
1037
1038 inline void edit_xform ( const char* tag, const XFORM& xform )
1039 {
1040     printf( "\\t%s.eM11\\t: %f\\n", tag, xform.eM11 );
1041     printf( "\\t%s.eM12\\t: %f\\n", tag, xform.eM12 );
1042     printf( "\\t%s.eM21\\t: %f\\n", tag, xform.eM21 );
1043     printf( "\\t%s.eM22\\t: %f\\n", tag, xform.eM22 );
1044     printf( "\\t%s.eDx\\t: %f\\n", tag, xform.eDx );
1045     printf( "\\t%s.eDy\\t: %f\\n", tag, xform.eDy );
1046 }
1047
1048 inline void edit_color ( const char* tag, const COLORREF& color )
1049 {
1050     #if defined(__LP64__)
1051         const char* FMT = "\\t%s\\t: R(0x%02x) G(0x%02x) B(0x%02x)\\n";
1052     #else
1053         const char* FMT = "\\t%s\\t: R(0x%02lx) G(0x%02lx) B(0x%02lx)\\n";
1054     #endif /* __x86_64__ */
1055     printf( FMT, tag,
1056         GetRValue( color ), GetGValue( color ), GetBValue( color ) );
1057 }
1058
1059 inline void edit_size ( const char* tag, const SIZE& size )
1060 {
1061     #if defined(__LP64__)
1062         const char* FMT = "\\t%s\\t: (%d, %d)\\n";
1063     #else
1064         const char* FMT = "\\t%s\\t: (%ld, %ld)\\n";
1065     #endif /* __x86_64__ */
1066     printf( FMT, tag, size.cx, size.cy );
1067 }
1068
1069 inline void edit_pointl ( const char* tag, const POINTL& point )
1070 {
1071     #if defined(__LP64__)
1072         const char* FMT = "\\t%s\\t: (%d, %d)\\n";
1073     #else
1074         const char* FMT = "\\t%s\\t: (%ld, %ld)\\n";
1075     #endif /* __x86_64__ */
1076     printf( FMT, tag, point.x, point.y );
1077 }
1078
1079 inline void edit_pointlarray ( const char* tag, const DWORD cptl,
1080     const POINTL* points )
1081 {
1082     #if defined(__LP64__)
1083         const char* FMT0 = "\\tcptl%s\\t: %d\\n";
1084         const char* FMT1 = "%d, %d\\n";
1085         const char* FMT2 = "\\t\\t%s %d, %d\\n";
1086     #else
1087         const char* FMT0 = "\\tcptl%s\\t: %ld\\n";
1088         const char* FMT1 = "%ld, %ld\\n";
1089         const char* FMT2 = "\\t\\t%s %ld, %ld\\n";
1090     #endif /* __x86_64__ */
1091     printf( FMT0, tag, cptl );
1092     printf( "\\taptl%s\\t: ", tag );
1093     if ( cptl > 0 )
1094         printf( FMT1, points[0].x, points[0].y );
1095     else
1096         puts( "" );
1097     for ( DWORD i = 1; i < cptl; i++ )
1098         printf( FMT2, tag, points[i].x, points[i].y );
1099 }
1100
1101 inline void edit_pointl6array ( const char* tag, const unsigned int cpts,
1102     const POINTL6* points )
1103 {
1104     printf( "\\tcpts%s\\t: %d\\n", tag, cpts );
1105     printf( "\\tapts%s\\t: ", tag );
1106     if ( cpts > 0 )
1107         printf( "%d, %d\\n", points[0].x, points[0].y );
1108     else
1109         puts( "" );

```

```

1110     for ( unsigned int i = 1; i < cpts; i++ )
1111         printf( "\t\t%s  %d, %d\n", tag, points[i].x, points[i].y );
1112 }
1113
1114 inline void edit_pen_style ( const char* tag, DWORD style )
1115 {
1116     printf( "\t%s\t:  ", tag );
1117     switch ( style & PS_STYLE_MASK ) {
1118     case PS_SOLID: printf( "PS_SOLID" ); break;
1119     case PS_DASH: printf( "PS_DASH" ); break;
1120     case PS_DOT: printf( "PS_DOT" ); break;
1121     case PS_DASHDOT: printf( "PS_DASHDOT" ); break;
1122     case PS_DASHDOTDOT: printf( "PS_DASHDOTDOT" ); break;
1123     case PS_NULL: printf( "PS_NULL" ); break;
1124     case PS_INSDFRAME: printf( "PS_INSDFRAME" ); break;
1125     case PS_USERSTYLE: printf( "PS_USERSTYLE" ); break;
1126     case PS_ALTERNATE: printf( "PS_ALTERNATE" ); break;
1127     }
1128     switch ( style & PS_ENDCAP_MASK ) {
1129     case PS_ENDCAP_ROUND: printf( " | PS_ENDCAP_ROUND" ); break;
1130     case PS_ENDCAP_SQUARE: printf( " | PS_ENDCAP_SQUARE" ); break;
1131     case PS_ENDCAP_FLAT: printf( " | PS_ENDCAP_FLAT" ); break;
1132     }
1133     switch ( style & PS_JOIN_MASK ) {
1134     case PS_JOIN_ROUND: printf( " | PS_JOIN_ROUND" ); break;
1135     case PS_JOIN_BEVEL: printf( " | PS_JOIN_BEVEL" ); break;
1136     case PS_JOIN_MITER: printf( " | PS_JOIN_MITER" ); break;
1137     }
1138     switch ( style & PS_TYPE_MASK ) {
1139     case PS_COSMETIC: printf( " | PS_COSMETIC" ); break;
1140     case PS_GEOMETRIC: printf( " | PS_GEOMETRIC" ); break;
1141     }
1142     printf( "\n" );
1143 }
1144
1145 inline void edit_brush_style ( const char* tag, DWORD style )
1146 {
1147     #if defined(__LP64__)
1148         const char* FMT = "unknown(%d)";
1149     #else
1150         const char* FMT = "unknown(%ld)";
1151     #endif /* __x86_64__ */
1152     printf( "\t%s\t:  ", tag );
1153     switch ( style ) {
1154     case BS_SOLID: printf( "BS_SOLID" ); break;
1155     case BS_NULL: printf( "BS_NULL" ); break;
1156     case BS_HATCHED: printf( "BS_HATCHED" ); break;
1157     case BS_PATTERN: printf( "BS_PATTERN" ); break;
1158     case BS_INDEXED: printf( "BS_INDEXED" ); break;
1159     case BS_DIBPATTERN: printf( "BS_DIBPATTERN" ); break;
1160     case BS_DIBPATTERNPT: printf( "BS_DIBPATTERNPT" ); break;
1161     case BS_PATTERN8X8: printf( "BS_PATTERN8X8" ); break;
1162     case BS_DIBPATTERN8X8: printf( "BS_DIBPATTERN8X8" ); break;
1163     case BS_MONOPATTERN: printf( "BS_DIBPATTERN8X8" ); break;
1164     default: printf( FMT, style );
1165     }
1166     printf( "\n" );
1167 }
1168
1169 inline void edit_brush_hatch ( const char* tag, DWORD hatch )
1170 {
1171     #if defined(__LP64__)
1172         const char* FMT = "unknown(%d)";
1173     #else
1174         const char* FMT = "unknown(%ld)";
1175     #endif /* __x86_64__ */
1176     printf( "\t%s\t:  ", tag );
1177     switch ( hatch ) {
1178     case HS_HORIZONTAL: printf( "HS_HORIZONTAL" ); break;
1179     case HS_VERTICAL: printf( "HS_VERTICAL" ); break;
1180     case HS_FDIAGONAL: printf( "HS_FDIAGONAL" ); break;
1181     case HS_BDIAGONAL: printf( "HS_BDIAGONAL" ); break;
1182     case HS_CROSS: printf( "HS_CROSS" ); break;
1183     case HS_DIAGCROSS: printf( "HS_DIAGCROSS" ); break;
1184     default: printf( FMT, hatch );
1185     }
1186     printf( "\n" );
1187 }
1188 #endif
1189 enum OBJECTTYPE { O_METAFILEDEVICECONTEXT = OBJ_METADC,
1190                 O_FONT = OBJ_FONT,
1191                 O_PEN = OBJ_PEN,
1192                 O_EXTPEN = OBJ_EXTPEN,
1193                 O_BRUSH = OBJ_BRUSH,
1194                 O_PALETTE = OBJ_PAL };
1195
1196 #if 0
1197 static char* typStr ( OBJECTTYPE type )

```

```

1207 {
1208     switch (type) {
1209         case O_METAFILEDEVICECONTEXT:
1210             return "metafile device context";
1211         case O_FONT:
1212             return "font";
1213         case O_PEN:
1214             return "pen";
1215         case O_EXTPEN:
1216             return "extended pen";
1217         case O_BRUSH:
1218             return "brush";
1219         case O_PALETTE:
1220             return "palette";
1221     }
1222     return "unknown object";
1223 }
1224 #endif
1225
1226 class OBJECT {
1227 public:
1228     HGDIOBJ handle;
1229     virtual ~OBJECT () {}
1230     OBJECT ( void ) : handle ( 0 ) {}
1231     virtual OBJECTTYPE getType ( void ) const = 0;
1232 };
1233
1234 class GRAPHICSOBJECT : public OBJECT {
1235 public:
1236     virtual ~GRAPHICSOBJECT () {}
1237     std::map< HDC, HGDIOBJ > contexts;
1238     virtual METARECORD* newEMR ( HDC dc, HGDIOBJ handle ) = 0;
1239 };
1240
1241 typedef METARECORD* (*METARECORDCTOR) (DATASTREAM&);
1242
1243 class GLOBALOBJECTS {
1244     std::vector<OBJECT*> objects;
1245
1246     std::map< DWORD, METARECORDCTOR > new_records;
1247
1248 public:
1249     GLOBALOBJECTS ( void );
1250     ~GLOBALOBJECTS ( void );
1251     HGDIOBJ add ( OBJECT* object );
1252     OBJECT* find ( const HGDIOBJ handle );
1253     void remove ( const OBJECT* object );
1254
1255     auto begin ( void )const { return objects.begin(); }
1256
1257     auto end ( void )const { return objects.end(); }
1258
1259     METARECORDCTOR newRecord ( DWORD iType ) const;
1260
1261     static EMF::METARECORD* new_eof ( DATASTREAM& ds );
1262     static EMF::METARECORD* new_setviewportorgex ( DATASTREAM& ds );
1263     static EMF::METARECORD* new_setwindoworgex ( DATASTREAM& ds );
1264     static EMF::METARECORD* new_setviewporttextex ( DATASTREAM& ds );
1265     static EMF::METARECORD* new_setwindowextex ( DATASTREAM& ds );
1266     static EMF::METARECORD* new_scaleviewporttextex ( DATASTREAM& ds );
1267     static EMF::METARECORD* new_scalewindowextex ( DATASTREAM& ds );
1268     static EMF::METARECORD* new_modifyworldtransform ( DATASTREAM& ds );
1269     static EMF::METARECORD* new_setworldtransform ( DATASTREAM& ds );
1270     static EMF::METARECORD* new_settextalign ( DATASTREAM& ds );
1271     static EMF::METARECORD* new_settextcolor ( DATASTREAM& ds );
1272     static EMF::METARECORD* new_setbkcolor ( DATASTREAM& ds );
1273     static EMF::METARECORD* new_setbkmode ( DATASTREAM& ds );
1274     static EMF::METARECORD* new_setpolyfillmode ( DATASTREAM& ds );
1275     static EMF::METARECORD* new_setmapmode ( DATASTREAM& ds );
1276     static EMF::METARECORD* new_selectobject ( DATASTREAM& ds );
1277     static EMF::METARECORD* new_deleteobject ( DATASTREAM& ds );
1278     static EMF::METARECORD* new_movetoex ( DATASTREAM& ds );
1279     static EMF::METARECORD* new_lineto ( DATASTREAM& ds );
1280     static EMF::METARECORD* new_arc ( DATASTREAM& ds );
1281     static EMF::METARECORD* new_arcto ( DATASTREAM& ds );
1282     static EMF::METARECORD* new_rectangle ( DATASTREAM& ds );
1283     static EMF::METARECORD* new_ellipse ( DATASTREAM& ds );
1284     static EMF::METARECORD* new_polyline ( DATASTREAM& ds );
1285     static EMF::METARECORD* new_polyline16 ( DATASTREAM& ds );
1286     static EMF::METARECORD* new_polygon ( DATASTREAM& ds );
1287     static EMF::METARECORD* new_polygon16 ( DATASTREAM& ds );
1288     static EMF::METARECORD* new_polypolygon ( DATASTREAM& ds );
1289     static EMF::METARECORD* new_polypolygon16 ( DATASTREAM& ds );
1290     static EMF::METARECORD* new_polybezier ( DATASTREAM& ds );
1291     static EMF::METARECORD* new_polybezier16 ( DATASTREAM& ds );
1292     static EMF::METARECORD* new_polybezierto ( DATASTREAM& ds );

```

```

1374 static EMF::METARECORD* new_polybeziertol6 ( DATASTREAM& ds );
1375 static EMF::METARECORD* new_polylineto ( DATASTREAM& ds );
1376 static EMF::METARECORD* new_polylinetol6 ( DATASTREAM& ds );
1377 static EMF::METARECORD* new_exttextouta ( DATASTREAM& ds );
1378 static EMF::METARECORD* new_exttextoutw ( DATASTREAM& ds );
1379 static EMF::METARECORD* new_setpixelv ( DATASTREAM& ds );
1380 static EMF::METARECORD* new_createpen ( DATASTREAM& ds );
1381 static EMF::METARECORD* new_extcreatepen ( DATASTREAM& ds );
1382 static EMF::METARECORD* new_createbrushindirect ( DATASTREAM& ds );
1383 static EMF::METARECORD* new_extcreatefontindirectw ( DATASTREAM& ds );
1384 static EMF::METARECORD* new_fillpath ( DATASTREAM& ds );
1385 static EMF::METARECORD* new_strokepath ( DATASTREAM& ds );
1386 static EMF::METARECORD* new_strokeandfillpath ( DATASTREAM& ds );
1387 static EMF::METARECORD* new_beginpath ( DATASTREAM& ds );
1388 static EMF::METARECORD* new_endpath ( DATASTREAM& ds );
1389 static EMF::METARECORD* new_closefigure ( DATASTREAM& ds );
1390 static EMF::METARECORD* new_savedc ( DATASTREAM& ds );
1391 static EMF::METARECORD* new_restoredc ( DATASTREAM& ds );
1392 static EMF::METARECORD* new_setmetargn ( DATASTREAM& ds );
1393 static EMF::METARECORD* new_setmiterlimit ( DATASTREAM& ds );
1394 };
1395
1396 extern GLOBALOBJECTS globalObjects;
1397
1398 class ENHMETAHEADER : public METARECORD, public ::ENHMETAHEADER {
1399 public:
1400     ENHMETAHEADER ( LPCWSTR description = 0 )
1401         : description_w( 0 ), description_size( 0 )
1402     {
1403         iType = EMR_HEADER;
1404         nSize = sizeof( ::ENHMETAHEADER );
1405
1406         // Compute the bounds
1407         RECT default_bounds = { 0, 0, 0, 0 };
1408         rc1Bounds = default_bounds;
1409         RECT default_frame = { 0, 0, 0, 0 };
1410         rc1Frame = default_frame;
1411         dSignature = ENHMETA_SIGNATURE;
1412         nVersion = 0x10000;
1413         nBytes = nSize;
1414         nRecords = 1;
1415         nHandles = 0;
1416         sReserved = 0;
1417         nDescription = 0;
1418         offDescription = 0;
1419         nPalEntries = 0;
1420         szlDevice.cx = XMAX_PIXELS;
1421         szlDevice.cy = YMAX_PIXELS;
1422         szlMillimeters.cx = XMAX_MM;
1423         szlMillimeters.cy = YMAX_MM;
1424         //
1425         cbPixelFormat = 0;
1426         offPixelFormat = 0;
1427         bOpenGL = FALSE;
1428         //
1429 #if 1
1430         szlMicrometers.cx = 1000 * szlMillimeters.cx;
1431         szlMicrometers.cy = 1000 * szlMillimeters.cy;
1432 #endif
1433         if ( description ) {
1434             // Count the number of characters in the description
1435             int description_count = 0, nulls = 0;
1436             LPCWSTR description_p = description;
1437             while ( nulls < 3 ) {
1438                 description_count++;
1439                 if ( (*description_p++) == 0 ) nulls++;
1440             }
1441
1442             // Make sure that the TOTAL record length will be a multiple of 4
1443
1444             int record_size = ROUND_TO_LONG( sizeof( ::ENHMETAHEADER ) +
1445                 sizeof( WCHAR ) * description_count );
1446             description_size =
1447                 (record_size - sizeof( ::ENHMETAHEADER )) / sizeof( WCHAR );
1448
1449             std::unique_ptr<WCHAR[]>
1450                 description_tmp( new WCHAR[ description_size ] );
1451
1452             description_w = description_tmp.release();
1453
1454             memset( description_w, 0, sizeof(WCHAR) * description_size );
1455         }
1456     }

```



```

1492     for ( int i=0; i<description_count; i++ )
1493         description_w[i] = *description++;
1494
1495     nSize = nBytes = record_size;
1496     nDescription = description_count;
1497     offDescription = sizeof( ::ENHMETAHEADER );
1498 }
1499 }
1500
1501 ~ENHMETAHEADER ( )
1502 {
1503     if ( description_w ) delete[] description_w;
1504 }
1505
1506 bool serialize ( DATASTREAM ds )
1507 {
1508     ds « iType « nSize
1509     « rclBounds « rclFrame
1510     « dSignature « nVersion « nBytes « nRecords « nHandles « sReserved
1511     « nDescription « offDescription « nPalEntries
1512     « szlDevice « szlMillimeters
1513     « cbPixelFormat « offPixelFormat « bOpenGL
1514     « szlMicrometers
1515     « WCHARSTR( description_w, description_size );
1516     return true;
1517 }
1518
1519 bool unserialize ( DATASTREAM ds )
1520 {
1521     ds » iType » nSize
1522     » rclBounds » rclFrame
1523     » dSignature » nVersion » nBytes » nRecords » nHandles » sReserved
1524     » nDescription » offDescription » nPalEntries
1525     » szlDevice » szlMillimeters;
1526
1527     // Some elements of the metafile header were added at later dates
1528
1529 #define OffsetOf( a, b ) ((unsigned int)((char*)&(((::ENHMETAHEADER*)a)->b)) - \
1530 (char*)&(((::ENHMETAHEADER*)a)))
1531     if ( OffsetOf( this, szlMicrometers ) <= offDescription )
1532         ds » cbPixelFormat » offPixelFormat » bOpenGL;
1533 #undef OffsetOf
1534     if ( sizeof(::ENHMETAHEADER) <= offDescription )
1535         ds » szlMicrometers;
1536
1537     // Should now probably check that the offset is correct...
1538
1539     int description_size_to_read = ( nSize - offDescription ) / sizeof(WCHAR);
1540
1541     if ( description_size_to_read < (int)nDescription ) {
1542         throw std::runtime_error( "record size inconsistent with description size" );
1543     }
1544
1545     description_size = max( 2, description_size_to_read );
1546
1547     std::unique_ptr<WCHAR[]> buffer( new WCHAR[description_size] );
1548
1549     WCHARSTR description( buffer.get(), description_size_to_read );
1550
1551     ds » description;
1552
1553     description_w = buffer.release();
1554
1555     // Make sure it's terminated properly.
1556     description_w[description_size-1] = 0;
1557     description_w[description_size-2] = 0;
1558
1559     return true;
1560 }
1561
1562 int size ( void )const { return nSize; }
1563
1564 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1565 {
1566     // Actually handled by the destination device context.
1567     EMF_UNUSED(source);
1568     EMF_UNUSED(dc);
1569 }
1570
1571 #ifdef ENABLE_EDITING
1572 void edit ( void )const
1573 {
1574     #if defined(__LP64__)
1575         const char* FMT0 = "\tiType\t\t\t: %d\n";
1576         const char* FMT1 = "\tnSize\t\t\t: %d\n";
1577         const char* FMT2 = "\tnBytes\t\t\t: %d\n";
1578         const char* FMT3 = "\tnRecords\t\t: %d\n";
1579         const char* FMT4 = "\tnDescription\t\t: %d\n";
1580         const char* FMT5 = "\toffDescription\t\t: %d\n";
1581         const char* FMT6 = "\tnPalEntries\t\t: %d\n";
1582         const char* FMT7 = "\tcbPixelFormat\t\t: %d\n";
1583         const char* FMT8 = "\toffPixelFormat\t\t: %d\n";
1584     #endif
1585 }
1586 #endif

```

```

1600     const char* FMT9 = "\tbOpenGL\t\t\t: %d\n";
1601 #else
1602     const char* FMT0 = "\tiType\t\t\t: %ld\n";
1603     const char* FMT1 = "\tnSize\t\t\t: %ld\n";
1604     const char* FMT2 = "\tnBytes\t\t\t: %ld\n";
1605     const char* FMT3 = "\tnRecords\t\t\t: %ld\n";
1606     const char* FMT4 = "\tnDescription\t\t\t: %ld\n";
1607     const char* FMT5 = "\toffDescription\t\t\t: %ld\n";
1608     const char* FMT6 = "\tnPalEntries\t\t\t: %ld\n";
1609     const char* FMT7 = "\tcbPixelFormat\t\t\t: %ld\n";
1610     const char* FMT8 = "\toffPixelFormat\t\t\t: %ld\n";
1611     const char* FMT9 = "\tbOpenGL\t\t\t: %ld\n";
1612 #endif
1613     printf( " *HEADER*\n" );
1614     printf( FMT0, iType );
1615     printf( FMT1, nSize );
1616     edit_rectl( "rclBounds\t", rclBounds );
1617     edit_rectl( "rclFrame\t", rclFrame );
1618     printf( "\tdSignature\t\t: %.4s\n", (const char*)&dSignature );
1619     printf( "\tnVersion\t\t: 0x%x\n", (unsigned int)nVersion );
1620     printf( FMT2, nBytes );
1621     printf( FMT3, nRecords );
1622     printf( "\tnHandles\t\t: %d\n", nHandles );
1623     printf( FMT4, nDescription );
1624     printf( FMT5, offDescription );
1625     printf( FMT6, nPalEntries );
1626     edit_size( "szlDevice\t", szlDevice );
1627     edit_size( "szlMillimeters\t", szlMillimeters );
1628
1629     /* Make a crude guess as to the age of this file */
1630 #define OffsetOf( a, b ) ((unsigned int)((const char*)&(((const ::ENHMETAHEADER*)a)->b)) - \
1631 (const char*)&(((const ::ENHMETAHEADER*)a)))
1632
1633     if ( OffsetOf( this, cbPixelFormat ) <= offDescription ) {
1634         printf( FMT7, cbPixelFormat );
1635         printf( FMT8, offPixelFormat );
1636         printf( FMT9, bOpenGL );
1637     }
1638     if ( sizeof(::ENHMETAHEADER) <= offDescription ) {
1639         edit_size( "szlMicrometers\t", szlMicrometers );
1640     }
1641 #endif
1642 }
1643
1644 #undef OffsetOf
1645
1646     if ( nDescription != 0 ) {
1647
1648         wchar_t last_w = 0;
1649         WCHAR* description = description_w;
1650
1651         printf( "\tDescription:" );
1652
1653         for ( DWORD i = 0; i < nDescription; i++ ) {
1654
1655             wchar_t w = *description++; /* This is not true, really. UNICODE is not
1656 * glibc's wide character representation */
1657
1658             if ( w != 0 ) {
1659                 if ( last_w == 0 ) printf( "\n\t\t" );
1660                 putchar( w );
1661             }
1662
1663             last_w = w;
1664         }
1665         printf( "\n" );
1666     }
1667 }
1668 #endif /* ENABLE_EDITING */
1669 };
1670
1671
1672
1673 class EMREOF : public METARECORD, ::EMREOF {
1674 public:
1675     EMREOF ( void )
1676     {
1677         emr.iType = EMR_EOF;
1678         emr.nSize = sizeof( ::EMREOF );
1679         nPalEntries = 0;
1680         offPalEntries = 0;
1681         nSizeLast = 0;
1682     }
1683
1684     EMREOF ( DATASTREAM& ds )
1685     {
1686         ds >> emr >> nPalEntries >> offPalEntries >> nSizeLast;
1687     }

```

```

1699
1703     bool serialize ( DATASTREAM ds )
1704     {
1705         ds « emr « nPalEntries « offPalEntries « nSizeLast;
1706         return true;
1707     }
1711     int size ( void )const { return emr.nSize; }
1717     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1718     {
1719         // Actually handled by the destination device context.
1720         EMF_UNUSED(source);
1721         EMF_UNUSED(dc);
1722     }
1723 #ifdef ENABLE_EDITING
1727     void edit ( void )const
1728     {
1729         printf( "EOF*\n" );
1730     }
1731 #endif /* ENABLE_EDITING */
1732 };
1733
1735
1740 class EMRSETVIEWPORTORGE : public METARECORD, ::EMRSETVIEWPORTORGE {
1741 public:
1746     EMRSETVIEWPORTORGE ( INT x, INT y )
1747     {
1748         emr.iType = EMR_SETVIEWPORTORGE;
1749         emr.nSize = sizeof( ::EMRSETVIEWPORTORGE );
1750         ptlOrigin.x = x;
1751         ptlOrigin.y = y;
1752     }
1757     EMRSETVIEWPORTORGE ( DATASTREAM& ds )
1758     {
1759         ds » emr » ptlOrigin;
1760     }
1764     bool serialize ( DATASTREAM ds )
1765     {
1766         ds « emr « ptlOrigin;
1767         return true;
1768     }
1772     int size ( void )const { return emr.nSize; }
1778     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1779     {
1780         EMF_UNUSED(source);
1781         SetViewportOrgEx( dc, ptlOrigin.x, ptlOrigin.y, 0 );
1782     }
1783 #ifdef ENABLE_EDITING
1787     void edit ( void )const
1788     {
1789         printf( "SETVIEWPORTORGE*\n" );
1790         edit_pointl( "ptlOrigin", ptlOrigin );
1791     }
1792 #endif /* ENABLE_EDITING */
1793 };
1794
1796
1803 class EMRSETWINDOWORGE : public METARECORD, ::EMRSETWINDOWORGE {
1804 public:
1809     EMRSETWINDOWORGE ( INT x, INT y )
1810     {
1811         emr.iType = EMR_SETWINDOWORGE;
1812         emr.nSize = sizeof( ::EMRSETWINDOWORGE );
1813         ptlOrigin.x = x;
1814         ptlOrigin.y = y;
1815     }
1820     EMRSETWINDOWORGE ( DATASTREAM& ds )
1821     {
1822         ds » emr » ptlOrigin;
1823     }
1827     bool serialize ( DATASTREAM ds )
1828     {
1829         ds « emr « ptlOrigin;
1830         return true;
1831     }
1835     int size ( void )const { return emr.nSize; }
1841     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1842     {
1843         EMF_UNUSED(source);
1844         SetWindowOrgEx( dc, ptlOrigin.x, ptlOrigin.y, 0 );
1845     }
1846 #ifdef ENABLE_EDITING
1850     void edit ( void )const
1851     {
1852         printf( "SETWINDOWORGE*\n" );
1853         edit_pointl( "ptlOrigin", ptlOrigin );
1854     }
1855 #endif /* ENABLE_EDITING */

```

```

1856     };
1857
1859
1864     class EMRSETVIEWPORTEXTX : public METARECORD, ::EMRSETVIEWPORTEXTX {
1865     public:
1870         EMRSETVIEWPORTEXTX ( INT cx, INT cy )
1871         {
1872             emr.iType = EMR_SETVIEWPORTEXTX;
1873             emr.nSize = sizeof( ::EMRSETVIEWPORTEXTX );
1874             szlExtent.cx = cx;
1875             szlExtent.cy = cy;
1876         }
1881         EMRSETVIEWPORTEXTX ( DATASTREAM& ds )
1882         {
1883             ds » emr » szlExtent;
1884         }
1888         bool serialize ( DATASTREAM ds )
1889         {
1890             ds « emr « szlExtent;
1891             return true;
1892         }
1896         int size ( void )const { return emr.nSize; }
1902         void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1903         {
1904             EMF_UNUSED(source);
1905             SetViewportExtEx( dc, szlExtent.cx, szlExtent.cy, 0 );
1906         }
1907 #ifdef ENABLE_EDITING
1911         void edit ( void )const
1912         {
1913             printf( "SETVIEWPORTEXTX*\n" );
1914             edit_sizel( "szlExtent", szlExtent );
1915         }
1916 #endif /* ENABLE_EDITING */
1917     };
1918
1920
1925     class EMRSCALEVIEWPORTEXTX : public METARECORD, ::EMRSCALEVIEWPORTEXTX {
1926     public:
1933         EMRSCALEVIEWPORTEXTX ( LONG x_num, LONG x_den, LONG y_num, LONG y_den )
1934         {
1935             emr.iType = EMR_SCALEVIEWPORTEXTX;
1936             emr.nSize = sizeof( ::EMRSCALEVIEWPORTEXTX );
1937             xNum = x_num;
1938             xDenom = x_den;
1939             yNum = y_num;
1940             yDenom = y_den;
1941         }
1946         EMRSCALEVIEWPORTEXTX ( DATASTREAM& ds )
1947         {
1948             ds » emr » xNum » xDenom » yNum » yDenom;
1949         }
1953         bool serialize ( DATASTREAM ds )
1954         {
1955             ds « emr « xNum « xDenom « yNum « yDenom;
1956             return true;
1957         }
1961         int size ( void )const { return emr.nSize; }
1967         void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
1968         {
1969             EMF_UNUSED(source);
1970             ScaleViewportExtEx( dc, xNum, xDenom, yNum, yDenom, 0 );
1971         }
1972 #ifdef ENABLE_EDITING
1976         void edit ( void )const
1977         {
1978             #if defined(__LP64__)
1979                 const char* FMT0 = "\txNum\t: %d\n";
1980                 const char* FMT1 = "\txDenom\t: %d\n";
1981                 const char* FMT2 = "\tyNum\t: %d\n";
1982                 const char* FMT3 = "\tyDenom\t: %d\n";
1983             #else
1984                 const char* FMT0 = "\txNum\t: %ld\n";
1985                 const char* FMT1 = "\txDenom\t: %ld\n";
1986                 const char* FMT2 = "\tyNum\t: %ld\n";
1987                 const char* FMT3 = "\tyDenom\t: %ld\n";
1988             #endif
1989             printf( "SCALEVIEWPORTEXTX*\n" );
1990             printf( FMT0, xNum );
1991             printf( FMT1, xDenom );
1992             printf( FMT2, yNum );
1993             printf( FMT3, yDenom );
1994         }
1995 #endif /* ENABLE_EDITING */
1996     };
1997
1999

```

```

2004 class EMRSETWINDOWEXTEx : public METARECORD, ::EMRSETWINDOWEXTEx {
2005 public:
2010     EMRSETWINDOWEXTEx ( INT cx, INT cy )
2011     {
2012         emr.iType = EMR_SETWINDOWEXTEx;
2013         emr.nSize = sizeof( ::EMRSETWINDOWEXTEx );
2014         szlExtent.cx = cx;
2015         szlExtent.cy = cy;
2016     }
2021     EMRSETWINDOWEXTEx ( DATASTREAM& ds )
2022     {
2023         ds » emr » szlExtent;
2024     }
2028     bool serialize ( DATASTREAM ds )
2029     {
2030         ds « emr « szlExtent;
2031         return true;
2032     }
2036     int size ( void )const { return emr.nSize; }
2042     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2043     {
2044         EMF_UNUSED(source);
2045         SetWindowExtEx( dc, szlExtent.cx, szlExtent.cy, 0 );
2046     }
2047 #ifdef ENABLE_EDITING
2051     void edit ( void )const
2052     {
2053         printf( "SETWINDOWEXTEx\n" );
2054         edit_sizel( "szlExtent", szlExtent );
2055     }
2056 #endif /* ENABLE_EDITING */
2057 };
2058
2060
2065 class EMRSCALEWINDOWEXTEx : public METARECORD, ::EMRSCALEWINDOWEXTEx {
2066 public:
2073     EMRSCALEWINDOWEXTEx ( LONG x_num, LONG x_den, LONG y_num, LONG y_den )
2074     {
2075         emr.iType = EMR_SCALEWINDOWEXTEx;
2076         emr.nSize = sizeof( ::EMRSCALEWINDOWEXTEx );
2077         xNum = x_num;
2078         xDenom = x_den;
2079         yNum = y_num;
2080         yDenom = y_den;
2081     }
2086     EMRSCALEWINDOWEXTEx ( DATASTREAM& ds )
2087     {
2088         ds » emr » xNum » xDenom » yNum » yDenom;
2089     }
2093     bool serialize ( DATASTREAM ds )
2094     {
2095         ds « emr « xNum « xDenom « yNum « yDenom;
2096         return true;
2097     }
2101     int size ( void )const { return emr.nSize; }
2107     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2108     {
2109         EMF_UNUSED(source);
2110         ScaleWindowExtEx( dc, xNum, xDenom, yNum, yDenom, 0 );
2111     }
2112 #ifdef ENABLE_EDITING
2116     void edit ( void )const
2117     {
2118         #if defined(__LP64__)
2119             const char* FMT0 = "txNum\t: %d\n";
2120             const char* FMT1 = "txDenom\t: %d\n";
2121             const char* FMT2 = "tyNum\t: %d\n";
2122             const char* FMT3 = "tyDenom\t: %d\n";
2123         #else
2124             const char* FMT0 = "txNum\t: %ld\n";
2125             const char* FMT1 = "txDenom\t: %ld\n";
2126             const char* FMT2 = "tyNum\t: %ld\n";
2127             const char* FMT3 = "tyDenom\t: %ld\n";
2128         #endif
2129         printf( "SCALEWINDOWEXTEx\n" );
2130         printf( FMT0, xNum );
2131         printf( FMT1, xDenom );
2132         printf( FMT2, yNum );
2133         printf( FMT3, yDenom );
2134     }
2135 #endif /* ENABLE_EDITING */
2136 };
2137
2139
2145 class EMRMODIFYWORLDTRANSFORM : public METARECORD, ::EMRMODIFYWORLDTRANSFORM {
2146 public:
2152     EMRMODIFYWORLDTRANSFORM ( const XFORM* transform, DWORD mode )

```

```

2153     {
2154         emr.iType = EMR_MODIFYWORLDTRANSFORM;
2155         emr.nSize = sizeof( ::EMRMODIFYWORLDTRANSFORM );
2156         xform = *transform;
2157         iMode = mode;
2158     }
2159     EMRMODIFYWORLDTRANSFORM ( DATASTREAM& ds )
2160     {
2161         ds » emr » xform » iMode;
2162     }
2163     bool serialize ( DATASTREAM ds )
2164     {
2165         ds « emr « xform « iMode;
2166         return true;
2167     }
2168     int size ( void )const { return emr.nSize; }
2169     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2170     {
2171         EMF_UNUSED(source);
2172         ModifyWorldTransform( dc, &xform, iMode );
2173     }
2174 #ifdef ENABLE_EDITING
2175     void edit ( void )const
2176     {
2177         #if defined(__LP64__)
2178             const char* FMT = "unknown(%d)\n";
2179         #else
2180             const char* FMT = "unknown(%ld)\n";
2181         #endif /* __x86_64__ */
2182         printf( "*MODIFYWORLDTRANSFORM*\n" );
2183         edit_xform( "xform", xform );
2184         printf( "\tiMode\t\t:  " );
2185         switch ( iMode ) {
2186             case MWT_IDENTITY: printf( "MWT_IDENTITY\n" ); break;
2187             case MWT_LEFTMULTIPLY: printf( "MWT_LEFTMULTIPLY\n" ); break;
2188             case MWT_RIGHTMULTIPLY: printf( "MWT_RIGHTMULTIPLY\n" ); break;
2189             default: printf( FMT, iMode );
2190         }
2191     }
2192 #endif /* ENABLE_EDITING */
2193 };
2194
2195 class EMRSETWORLDTRANSFORM : public METARECORD, ::EMRSETWORLDTRANSFORM {
2196 public:
2197     EMRSETWORLDTRANSFORM ( const XFORM* transform )
2198     {
2199         emr.iType = EMR_SETWORLDTRANSFORM;
2200         emr.nSize = sizeof( ::EMRSETWORLDTRANSFORM );
2201         xform = *transform;
2202     }
2203     EMRSETWORLDTRANSFORM ( DATASTREAM& ds )
2204     {
2205         ds » emr » xform;
2206     }
2207     bool serialize ( DATASTREAM ds )
2208     {
2209         ds « emr « xform;
2210         return true;
2211     }
2212     int size ( void )const { return emr.nSize; }
2213     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2214     {
2215         EMF_UNUSED(source);
2216         SetWorldTransform( dc, &xform );
2217     }
2218 #ifdef ENABLE_EDITING
2219     void edit ( void )const
2220     {
2221         printf( "*SETWORLDTRANSFORM*\n" );
2222         edit_xform( "xform", xform );
2223     }
2224 #endif /* ENABLE_EDITING */
2225 };
2226
2227 class EMRSETTEXTALIGN : public METARECORD, ::EMRSETTEXTALIGN {
2228 public:
2229     EMRSETTEXTALIGN ( UINT mode )
2230     {
2231         emr.iType = EMR_SETTEXTALIGN;
2232         emr.nSize = sizeof( ::EMRSETTEXTALIGN );
2233         iMode = mode;
2234     }
2235     EMRSETTEXTALIGN ( DATASTREAM& ds )
2236     {
2237         ds » emr » iMode;

```

```

2295     }
2299     bool serialize ( DATASTREAM ds )
2300     {
2301         ds « emr « iMode;
2302         return true;
2303     }
2307     int size ( void )const { return emr.nSize; }
2313     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2314     {
2315         EMF_UNUSED(source);
2316         SetTextAlign( dc, iMode );
2317     }
2318 #ifdef ENABLE_EDITING
2322     void edit ( void )const
2323     {
2324 #if defined(__LP64__)
2325         const char* FMT = "| unknown bits(0x%x)";
2326 #else
2327         const char* FMT = "| unknown bits(0x%lx)";
2328 #endif /* __x86_64__ */
2329         unsigned int known_bits = TA_BASELINE+TA_CENTER+TA_UPDATECP+TA_RTREADING;
2330         unsigned int unknown_bits = ~known_bits;
2331
2332         printf( "*SETTEXTALIGN*\n" );
2333         printf( "\tiMode\t:  " );
2334         if ( iMode & TA_UPDATECP )
2335             printf( "TA_UPDATECP" );
2336         else
2337             printf( "TA_NOUPDATECP" );
2338         if ( iMode & TA_CENTER )
2339             printf( " | TA_CENTER" );
2340         else if ( iMode & TA_RIGHT )
2341             printf( " | TA_RIGHT" );
2342         else
2343             printf( " | TA_LEFT" );
2344         if ( iMode & TA_BASELINE )
2345             printf( " | TA_BASELINE" );
2346         else if ( iMode & TA_BOTTOM )
2347             printf( " | TA_BOTTOM" );
2348         else
2349             printf( " | TA_TOP" );
2350         if ( iMode & TA_RTREADING )
2351             printf( " | TA_RTREADING" );
2352         if ( iMode & unknown_bits )
2353             printf( FMT, iMode & unknown_bits );
2354         printf( "\n" );
2355     }
2356 #endif /* ENABLE_EDITING */
2357 };
2358
2360
2363 class EMRSETTEXTCOLOR : public METARECORD, ::EMRSETTEXTCOLOR {
2364 public:
2368     EMRSETTEXTCOLOR ( COLORREF color )
2369     {
2370         emr.iType = EMR_SETTEXTCOLOR;
2371         emr.nSize = sizeof( ::EMRSETTEXTCOLOR );
2372         crColor = color;
2373     }
2378     EMRSETTEXTCOLOR ( DATASTREAM& ds )
2379     {
2380         ds » emr » crColor;
2381     }
2385     bool serialize ( DATASTREAM ds )
2386     {
2387         ds « emr « crColor;
2388         return true;
2389     }
2393     int size ( void )const { return emr.nSize; }
2399     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2400     {
2401         EMF_UNUSED(source);
2402         SetTextColor( dc, crColor );
2403     }
2404 #ifdef ENABLE_EDITING
2408     void edit ( void )const
2409     {
2410         printf( "*SETTEXTCOLOR*\n" );
2411         edit_color( "crColor", crColor );
2412     }
2413 #endif /* ENABLE_EDITING */
2414 };
2415
2417
2420 class EMRSETBKCOLOR : public METARECORD, ::EMRSETBKCOLOR {
2421 public:
2425     EMRSETBKCOLOR ( COLORREF color )

```

```

2426     {
2427         emr.iType = EMR_SETBKCOLOR;
2428         emr.nSize = sizeof( ::EMRSETBKCOLOR );
2429         crColor = color;
2430     }
2431     EMRSETBKCOLOR ( DATASTREAM& ds )
2432     {
2433         ds » emr » crColor;
2434     }
2435     bool serialize ( DATASTREAM ds )
2436     {
2437         ds « emr « crColor;
2438         return true;
2439     }
2440     int size ( void )const { return emr.nSize; }
2441     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2442     {
2443         EMF_UNUSED(source);
2444         SetBkColor( dc, crColor );
2445     }
2446 #ifdef ENABLE_EDITING
2447     void edit ( void )const
2448     {
2449         printf( "*SETBKCOLOR*\n" );
2450         edit_color( "crColor", crColor );
2451     }
2452 #endif /* ENABLE_EDITING */
2453 };
2454
2455 class EMRSETBKMODE : public METARECORD, ::EMRSETBKMODE {
2456 public:
2457     EMRSETBKMODE ( DWORD mode )
2458     {
2459         emr.iType = EMR_SETBKMODE;
2460         emr.nSize = sizeof( ::EMRSETBKMODE );
2461         iMode = mode;
2462     }
2463     EMRSETBKMODE ( DATASTREAM& ds )
2464     {
2465         ds » emr » iMode;
2466     }
2467     bool serialize ( DATASTREAM ds )
2468     {
2469         ds « emr « iMode;
2470         return true;
2471     }
2472     int size ( void )const { return emr.nSize; }
2473     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2474     {
2475         EMF_UNUSED(source);
2476         SetBkMode( dc, iMode );
2477     }
2478 #ifdef ENABLE_EDITING
2479     void edit ( void )const
2480     {
2481 #if defined(__LP64__)
2482         const char* FMT = "unknown(%d)\n";
2483 #else
2484         const char* FMT = "unknown(%ld)\n";
2485 #endif /* __x86_64__ */
2486         printf( "*SETBKMODE*\n" );
2487         printf( "\tiMode\t:  " );
2488         switch ( iMode ) {
2489             case TRANSPARENT: printf( "TRANSPARENT\n" ); break;
2490             case OPAQUE: printf( "OPAQUE\n" ); break;
2491             default: printf( FMT, iMode );
2492         }
2493     }
2494 #endif /* ENABLE_EDITING */
2495 };
2496
2497 class EMRSETPOLYFILLMODE : public METARECORD, ::EMRSETPOLYFILLMODE {
2498 public:
2499     EMRSETPOLYFILLMODE ( DWORD mode )
2500     {
2501         emr.iType = EMR_SETPOLYFILLMODE;
2502         emr.nSize = sizeof( ::EMRSETPOLYFILLMODE );
2503         iMode = mode;
2504     }
2505     EMRSETPOLYFILLMODE ( DATASTREAM& ds )
2506     {
2507         ds » emr » iMode;
2508     }
2509     bool serialize ( DATASTREAM ds )
2510     {

```



```

2569     ds « emr « iMode;
2570     return true;
2571 }
2575 int size ( void )const { return emr.nSize; }
2581 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2582 {
2583     EMF_UNUSED(source);
2584     SetPolyFillMode( dc, iMode );
2585 }
2586 #ifdef ENABLE_EDITING
2590 void edit ( void )const
2591 {
2592     #if defined(__LP64__)
2593         const char* FMT = "unknown(%d)\n";
2594     #else
2595         const char* FMT = "unknown(%ld)\n";
2596     #endif /* __x86_64__ */
2597     printf( "*SETPOLYFILLMODE*\n" );
2598     printf( "\tiMode:  " );
2599     switch ( iMode ) {
2600     case ALTERNATE: printf( "ALTERNATE\n" ); break;
2601     case WINDING: printf( "WINDING\n" ); break;
2602     default: printf( FMT, iMode );
2603     }
2604 }
2605 #endif /* ENABLE_EDITING */
2606 };
2607
2609
2613 class EMRSETMAPMODE : public METARECORD, ::EMRSETMAPMODE {
2614 public:
2618     EMRSETMAPMODE ( DWORD mode )
2619     {
2620         emr.iType = EMR_SETMAPMODE;
2621         emr.nSize = sizeof( ::EMRSETMAPMODE );
2622         iMode = mode;
2623     }
2628     EMRSETMAPMODE ( DATASTREAM& ds )
2629     {
2630         ds » emr » iMode;
2631     }
2635     bool serialize ( DATASTREAM ds )
2636     {
2637         ds « emr « iMode;
2638         return true;
2639     }
2643     int size ( void )const { return emr.nSize; }
2649     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2650     {
2651         EMF_UNUSED(source);
2652         SetMapMode( dc, iMode );
2653     }
2654     #ifdef ENABLE_EDITING
2658     void edit ( void )const
2659     {
2660         #if defined(__LP64__)
2661             const char* FMT = "unknown(%d)\n";
2662         #else
2663             const char* FMT = "unknown(%ld)\n";
2664         #endif /* __x86_64__ */
2665         printf( "*SETMAPMODE*\n" );
2666         printf( "\tiMode\t:  " );
2667         switch ( iMode ) {
2668         case MM_TEXT: printf( "MM_TEXT\n" ); break;
2669         case MM_LOMETRIC: printf( "MM_LOMETRIC\n" ); break;
2670         case MM_HIMETRIC: printf( "MM_HIMETRIC\n" ); break;
2671         case MM_LOENGLISH: printf( "MM_LOENGLISH\n" ); break;
2672         case MM_HIENGLISH: printf( "MM_HIENGLISH\n" ); break;
2673         case MM_TWIPS: printf( "MM_TWIPS\n" ); break;
2674         case MM_ISOTROPIC: printf( "MM_ISOTROPIC\n" ); break;
2675         case MM_ANISOTROPIC: printf( "MM_ANISOTROPIC\n" ); break;
2676         default: printf( FMT, iMode );
2677         }
2678     }
2679     #endif /* ENABLE_EDITING */
2680 };
2681
2683
2686 class EMRSELECTOBJECT : public METARECORD, ::EMRSELECTOBJECT {
2687 public:
2691     EMRSELECTOBJECT ( HGDIOBJ object )
2692     {
2693         emr.iType = EMR_SELECTOBJECT;
2694         emr.nSize = sizeof( ::EMRSELECTOBJECT );
2695         ihObject = object;
2696     }
2701     EMRSELECTOBJECT ( DATASTREAM& ds )

```

```

2702     {
2703         ds » emr » ihObject;
2704     }
2708     bool serialize ( DATASTREAM ds )
2709     {
2710         ds « emr « ihObject;
2711         return true;
2712     }
2716     int size ( void )const { return emr.nSize; }
2722     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
2723 #ifdef ENABLE_EDITING
2727     void edit ( void )const
2728     {
2729 #if defined(__LP64__)
2730         const char* FMT = "\tihObject\t: 0x%x\n";
2731 #else
2732         const char* FMT = "\tihObject\t: 0x%lx\n";
2733 #endif /* __x86_64__ */
2734         printf( "*SELECTOBJECT*\n" );
2735         printf( FMT, ihObject );
2736     }
2737 #endif /* ENABLE_EDITING */
2738 };
2739
2741
2744     class EMRDELETEOBJECT : public METARECORD, ::EMRDELETEOBJECT {
2745     public:
2749         EMRDELETEOBJECT ( HGDIOBJ object )
2750         {
2751             emr.iType = EMR_DELETEOBJECT;
2752             emr.nSize = sizeof( ::EMRDELETEOBJECT );
2753             ihObject = object;
2754         }
2759         EMRDELETEOBJECT ( DATASTREAM& ds )
2760         {
2761             ds » emr » ihObject;
2762         }
2766         bool serialize ( DATASTREAM ds )
2767         {
2768             ds « emr « ihObject;
2769             return true;
2770         }
2774         int size ( void )const { return emr.nSize; }
2780         void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
2781 #ifdef ENABLE_EDITING
2785         void edit ( void )const
2786         {
2787 #if defined(__LP64__)
2788             const char* FMT = "\tihObject\t: 0x%x\n";
2789 #else
2790             const char* FMT = "\tihObject\t: 0x%lx\n";
2791 #endif /* __x86_64__ */
2792             printf( "*DELETEOBJECT*\n" );
2793             printf( FMT, ihObject );
2794         }
2795 #endif /* ENABLE_EDITING */
2796 };
2797
2799
2802     class EMRMOVETOEX : public METARECORD, ::EMRMOVETOEX {
2803     public:
2808         EMRMOVETOEX ( INT x, INT y )
2809         {
2810             emr.iType = EMR_MOVETOEX;
2811             emr.nSize = sizeof( ::EMRMOVETOEX );
2812             ptl.x = x;
2813             ptl.y = y;
2814         }
2819         EMRMOVETOEX ( DATASTREAM& ds )
2820         {
2821             ds » emr » ptl;
2822         }
2826         bool serialize ( DATASTREAM ds )
2827         {
2828             ds « emr « ptl;
2829             return true;
2830         }
2834         int size ( void )const { return emr.nSize; }
2840         void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2841         {
2842             EMF_UNUSED(source);
2843             MoveToEx( dc, ptl.x, ptl.y, 0 );
2844         }
2845 #ifdef ENABLE_EDITING
2849         void edit ( void )const
2850         {
2851             printf( "*MOVETOEX*\n" );

```

```

2852     edit_pointl( "ptl", ptl );
2853 }
2854 #endif /* ENABLE_EDITING */
2855 };
2856
2857
2858
2859 class EMRLINETO : public METARECORD, ::EMRLINETO {
2860 public:
2861     EMRLINETO ( INT x, INT y )
2862     {
2863         emr.iType = EMR_LINETO;
2864         emr.nSize = sizeof( ::EMRLINETO );
2865         ptl.x = x;
2866         ptl.y = y;
2867     }
2868     EMRLINETO ( DATASTREAM& ds )
2869     {
2870         ds » emr » ptl;
2871     }
2872     bool serialize ( DATASTREAM ds )
2873     {
2874         ds « emr « ptl;
2875         return true;
2876     }
2877     int size ( void )const { return emr.nSize; }
2878     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2879     {
2880         EMF_UNUSED(source);
2881         LineTo( dc, ptl.x, ptl.y );
2882     }
2883 #ifdef ENABLE_EDITING
2884     void edit ( void )const
2885     {
2886         printf( "LINETO*\n" );
2887         edit_pointl( "ptl", ptl );
2888     }
2889 #endif /* ENABLE_EDITING */
2890 };
2891
2892
2893 class EMRARC : public METARECORD, ::EMRARC {
2894 public:
2895     EMRARC ( INT left, INT top, INT right, INT bottom, INT xstart,
2896             INT ystart, INT xend, INT yend )
2897     {
2898         emr.iType = EMR_ARC;
2899         emr.nSize = sizeof( ::EMRARC );
2900         rclBox.left = left;
2901         rclBox.right = right;
2902         rclBox.bottom = bottom;
2903         rclBox.top = top;
2904         ptlStart.x = xstart;
2905         ptlStart.y = ystart;
2906         ptlEnd.x = xend;
2907         ptlEnd.y = yend;
2908     }
2909     EMRARC ( DATASTREAM& ds )
2910     {
2911         ds » emr » rclBox » ptlStart » ptlEnd;
2912     }
2913     bool serialize ( DATASTREAM ds )
2914     {
2915         ds « emr « rclBox « ptlStart « ptlEnd;
2916         return true;
2917     }
2918     int size ( void )const { return emr.nSize; }
2919     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
2920     {
2921         EMF_UNUSED(source);
2922         Arc( dc, rclBox.left, rclBox.top, rclBox.right, rclBox.bottom,
2923             ptlStart.x, ptlStart.y, ptlEnd.x, ptlEnd.y );
2924     }
2925 #ifdef ENABLE_EDITING
2926     void edit ( void )const
2927     {
2928         printf( "ARC*\n" );
2929         edit_rectl( "rclBox\t", rclBox );
2930         edit_pointl( "ptlStart", ptlStart );
2931         edit_pointl( "ptlEnd\t", ptlEnd );
2932     }
2933 #endif /* ENABLE_EDITING */
2934 };
2935
2936
2937 class EMRARCTO : public METARECORD, ::EMRARCTO {
2938 public:
2939     EMRARCTO ( INT left, INT top, INT right, INT bottom, INT xstart,

```

```

3010         INT ystart, INT xend, INT yend )
3011     {
3012         emr.iType = EMR_ARCTO;
3013         emr.nSize = sizeof( ::EMRARCTO );
3014         rclBox.left = left;
3015         rclBox.right = right;
3016         rclBox.bottom = bottom;
3017         rclBox.top = top;
3018         ptlStart.x = xstart;
3019         ptlStart.y = ystart;
3020         ptlEnd.x = xend;
3021         ptlEnd.y = yend;
3022     }
3023     EMRARCTO ( DATASTREAM& ds )
3024     {
3025         ds » emr » rclBox » ptlStart » ptlEnd;
3026     }
3027     bool serialize ( DATASTREAM ds )
3028     {
3029         ds « emr « rclBox « ptlStart « ptlEnd;
3030         return true;
3031     }
3032     int size ( void )const { return emr.nSize; }
3033     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3034     {
3035         EMF_UNUSED(source);
3036         ArcTo( dc, rclBox.left, rclBox.top, rclBox.right, rclBox.bottom,
3037             ptlStart.x, ptlStart.y, ptlEnd.x, ptlEnd.y );
3038     }
3039     #ifdef ENABLE_EDITING
3040     void edit ( void )const
3041     {
3042         printf( "*ARCTO*\n" );
3043         edit_rectl( "rclBox\t", rclBox );
3044         edit_pointl( "ptlStart", ptlStart );
3045         edit_pointl( "ptlEnd\t", ptlEnd );
3046     }
3047     #endif /* ENABLE_EDITING */
3048 };
3049
3050 class EMRRECTANGLE : public METARECORD, ::EMRRECTANGLE {
3051 public:
3052     EMRRECTANGLE ( INT left, INT top, INT right, INT bottom )
3053     {
3054         emr.iType = EMR_RECTANGLE;
3055         emr.nSize = sizeof( ::EMRRECTANGLE );
3056         rclBox.left = left;
3057         rclBox.right = right;
3058         rclBox.bottom = bottom;
3059         rclBox.top = top;
3060     }
3061     EMRRECTANGLE ( DATASTREAM& ds )
3062     {
3063         ds » emr » rclBox;
3064     }
3065     bool serialize ( DATASTREAM ds )
3066     {
3067         ds « emr « rclBox;
3068         return true;
3069     }
3070     int size ( void )const { return emr.nSize; }
3071     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3072     {
3073         EMF_UNUSED(source);
3074         Rectangle( dc, rclBox.left, rclBox.top, rclBox.right, rclBox.bottom );
3075     }
3076     #ifdef ENABLE_EDITING
3077     void edit ( void )const
3078     {
3079         printf( "*RECTANGLE*\n" );
3080         edit_rectl( "rclBox", rclBox );
3081     }
3082     #endif /* ENABLE_EDITING */
3083 };
3084
3085 class EMRELLIPSE : public METARECORD, ::EMRELLIPSE {
3086 public:
3087     EMRELLIPSE ( INT left, INT top, INT right, INT bottom )
3088     {
3089         emr.iType = EMR_ELLIPSE;
3090         emr.nSize = sizeof( ::EMRELLIPSE );
3091         rclBox.left = left;
3092         rclBox.right = right;
3093         rclBox.bottom = bottom;
3094         rclBox.top = top;
3095     }

```

```

3152     }
3153     EMRELLIPSE ( DATASTREAM& ds )
3154     {
3155         ds » emr » rclBox;
3156     }
3157     bool serialize ( DATASTREAM ds )
3158     {
3159         ds « emr « rclBox;
3160         return true;
3161     }
3162     int size ( void )const { return emr.nSize; }
3163     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3164     {
3165         EMF_UNUSED(source);
3166         Ellipse( dc, rclBox.left, rclBox.top, rclBox.right, rclBox.bottom );
3167     }
3168     #ifdef ENABLE_EDITING
3169     void edit ( void )const
3170     {
3171         printf( "*ELLIPSE*\n" );
3172         edit_rectl( "rclBox", rclBox );
3173     }
3174     #endif /* ENABLE_EDITING */
3175 };
3176
3177 class EMPOLYLINE : public METARECORD, ::EMPOLYLINE {
3178     POINTL* lpoints{nullptr};
3179 public:
3180     EMPOLYLINE ( const RECTL* bounds, const POINT* points, INT n )
3181     {
3182         cptl = n;
3183         aptl[0].x = 0;          // Really unused
3184         aptl[0].y = 0;
3185
3186         emr.iType = EMR_POLYLINE;
3187         // The (cptl - 1) below is to account for aptl, which isn't written out
3188         emr.nSize = sizeof( ::EMPOLYLINE ) + sizeof( POINTL ) * ( cptl - 1 );
3189
3190         lpoints = new POINTL[cptl];
3191
3192         for (int i=0; i<n; i++) {
3193             lpoints[i].x = points[i].x;
3194             lpoints[i].y = points[i].y;
3195         }
3196
3197         rclBounds = *bounds;
3198     }
3199     ~EMPOLYLINE ( )
3200     {
3201         if ( lpoints ) delete[] lpoints;
3202     }
3203     EMPOLYLINE ( DATASTREAM& ds )
3204     {
3205         ds » emr » rclBounds » cptl;
3206
3207         if ( emr.nSize - (sizeof( ::EMPOLYLINE ) - sizeof( POINTL ) ) <
3208             sizeof( POINTL ) * cptl ) {
3209             throw std::runtime_error( "Invalid record size" );
3210         }
3211
3212         std::unique_ptr<POINTL[]> buffer( new POINTL[cptl] );
3213         POINTLARRAY points( buffer.get(), cptl );
3214
3215         ds » points;
3216
3217         lpoints = buffer.release();
3218     }
3219     bool serialize ( DATASTREAM ds )
3220     {
3221         ds « emr « rclBounds « cptl « POINTLARRAY( lpoints, cptl );
3222         return true;
3223     }
3224     int size ( void )const { return emr.nSize; }
3225     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3226     {
3227         EMF_UNUSED(source);
3228         // According to the wine windef.h header, POINT and POINTL are equivalent
3229         Polyline( dc, (POINT*)lpoints, cptl );
3230     }
3231     #ifdef ENABLE_EDITING
3232     void edit ( void )const
3233     {
3234         printf( "*POLYLINE*\n" );
3235         edit_rectl( "rclBounds", rclBounds );
3236         edit_pointlarray( "\t", cptl, lpoints );
3237     }
3238 }

```

```

3286 #endif /* ENABLE_EDITING */
3287 };
3288
3290
3293 class EMRPOLYLINE16 : public METARECORD, ::EMRPOLYLINE16 {
3294     POINT16* lpoints{ nullptr };
3295 public:
3301     EMRPOLYLINE16 ( const RECTL* bounds, const POINT16* points, INT n )
3302     {
3303         cpts = n;
3304         apts[0].x = 0;           // Really unused
3305         apts[0].y = 0;
3306
3307         emr.iType = EMR_POLYLINE16;
3308         // The (cptl - 1) below is to account for aptl, which isn't written out
3309         emr.nSize = sizeof( ::EMRPOLYLINE16 ) + sizeof( POINT16 ) * ( cpts - 1 );
3310
3311         lpoints = new POINT16[cpts];
3312
3313         for (int i=0; i<n; i++) {
3314             lpoints[i].x = points[i].x;
3315             lpoints[i].y = points[i].y;
3316         }
3317
3318         rclBounds = *bounds;
3319     }
3326     EMRPOLYLINE16 ( const RECTL* bounds, const POINT* points, INT n )
3327     {
3328         cpts = n;
3329         apts[0].x = 0;           // Really unused
3330         apts[0].y = 0;
3331
3332         emr.iType = EMR_POLYLINE16;
3333         // The (cptl - 1) below is to account for aptl, which isn't written out
3334         emr.nSize = sizeof( ::EMRPOLYLINE16 ) + sizeof( POINT16 ) * ( cpts - 1 );
3335
3336         lpoints = new POINT16[cpts];
3337
3338         for (int i=0; i<n; i++) {
3339             lpoints[i].x = points[i].x;
3340             lpoints[i].y = points[i].y;
3341         }
3342
3343         rclBounds = *bounds;
3344     }
3348     ~EMRPOLYLINE16 ( )
3349     {
3350         if ( lpoints ) delete[] lpoints;
3351     }
3356     EMRPOLYLINE16 ( DATASTREAM& ds )
3357     {
3358         ds » emr » rclBounds » cpts;
3359
3360         if ( emr.nSize - (sizeof(::EMRPOLYLINE16)-sizeof(POINT16)) <
3361             sizeof(POINT16) * cpts ) {
3362             throw std::runtime_error( "Invalid record size" );
3363         }
3364
3365         std::unique_ptr<POINT16[]> buffer( new POINT16[cpts] );
3366
3367         POINT16ARRAY points( buffer.get(), cpts );
3368
3369         ds » points;
3370
3371         lpoints = buffer.release();
3372     }
3376     bool serialize ( DATASTREAM ds )
3377     {
3378         ds « emr « rclBounds « cpts « POINT16ARRAY( lpoints, cpts );
3379         return true;
3380     }
3384     int size ( void )const { return emr.nSize; }
3390     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3391     {
3392         EMF_UNUSED(source);
3393         // According to the wine windef.h header, POINT and POINTL are equivalent
3394         Polyline16( dc, lpoints, cpts );
3395     }
3396 #ifdef ENABLE_EDITING
3400     void edit ( void )const
3401     {
3402         printf( "POLYLINE16*\n" );
3403         edit_rectl( "rclBounds", rclBounds );
3404         edit_point16array( "\t", cpts, lpoints );
3405     }
3406 #endif /* ENABLE_EDITING */
3407 };

```

```

3408
3409
3410
3413 class EMPOLYGON : public METARECORD, ::EMPOLYGON {
3414     POINTL* lpoints{ nullptr };
3415 public:
3421     EMPOLYGON ( const RECTL* bounds, const POINT* points, INT n )
3422     {
3423         cptl = n;
3424         aptl[0].x = 0;           // Really unused
3425         aptl[0].y = 0;
3426
3427         emr.iType = EMR_POLYGON;
3428         // The (cptl-1) below is to account for aptl, which isn't written out
3429         emr.nSize = sizeof( ::EMPOLYGON ) + sizeof( POINTL ) * (cptl-1);
3430
3431         lpoints = new POINTL[cptl];
3432
3433         for (int i=0; i<n; i++) {
3434             lpoints[i].x = points[i].x;
3435             lpoints[i].y = points[i].y;
3436         }
3437
3438         rclBounds = *bounds;
3439     }
3444     EMPOLYGON ( DATASTREAM& ds )
3445     {
3446         ds » emr » rclBounds » cptl;
3447
3448         if ( emr.nSize - (sizeof(::EMPOLYGON) - sizeof(POINTL)) <
3449             cptl * sizeof(POINTL) ) {
3450             throw std::runtime_error( "Invalid record size" );
3451         }
3452
3453         std::unique_ptr<POINTL[]> buffer( new POINTL[cptl] );
3454
3455         POINTLARRAY points( buffer.get(), cptl );
3456
3457         ds » points;
3458
3459         lpoints = buffer.release();
3460     }
3464     ~EMPOLYGON ( )
3465     {
3466         if ( lpoints ) delete[] lpoints;
3467     }
3471     bool serialize ( DATASTREAM ds )
3472     {
3473         ds « emr « rclBounds « cptl « POINTLARRAY( lpoints, cptl );
3474         return true;
3475     }
3479     int size ( void )const { return emr.nSize; }
3485     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3486     {
3487         EMF_UNUSED(source);
3488         // According to the wine windef.h header, POINT and POINTL are equivalent
3489         Polygon( dc, (POINT*)lpoints, cptl );
3490     }
3491 #ifdef ENABLE_EDITING
3495     void edit ( void )const
3496     {
3497         printf( "*POLYGON*\n" );
3498         edit_rectl( "rclBounds", rclBounds );
3499         edit_pointlarray( "\t", cptl, lpoints );
3500     }
3501 #endif /* ENABLE_EDITING */
3502 };
3503
3504
3505
3508 class EMPOLYGON16 : public METARECORD, ::EMPOLYGON16 {
3509     POINT16* lpoints{ nullptr };
3510 public:
3516     EMPOLYGON16 ( const RECTL* bounds, const POINT* points, INT16 n )
3517     {
3518         cpts = n;
3519         apts[0].x = 0;           // Really unused
3520         apts[0].y = 0;
3521
3522         emr.iType = EMR_POLYGON16;
3523         // The (cptl-1) below is to account for aptl, which isn't written out
3524         emr.nSize = sizeof( ::EMPOLYGON16 ) + sizeof( POINT16 ) * (cpts-1);
3525
3526         lpoints = new POINT16[cpts];
3527
3528         for (int i=0; i<n; i++) {
3529             lpoints[i].x = points[i].x;
3530             lpoints[i].y = points[i].y;
3531         }

```

```

3532
3533     rclBounds = *bounds;
3534 }
3541 EMRPOLYGON16 ( const RECTL* bounds, const POINT16* points, INT16 n )
3542 {
3543     cpts = n;
3544     apts[0].x = 0;           // Really unused
3545     apts[0].y = 0;
3546
3547     emr.iType = EMR_POLYGON16;
3548     // The (cptl-1) below is to account for aptl, which isn't written out
3549     emr.nSize = sizeof( ::EMRPOLYGON16 ) + sizeof( POINT16 ) * (cpts-1);
3550
3551     lpoints = new POINT16[cpts];
3552
3553     for (int i=0; i<n; i++) {
3554         lpoints[i].x = points[i].x;
3555         lpoints[i].y = points[i].y;
3556     }
3557
3558     rclBounds = *bounds;
3559 }
3564 EMRPOLYGON16 ( DATASTREAM& ds )
3565 {
3566     ds » emr » rclBounds » cpts;
3567
3568     if ( emr.nSize - (sizeof(::EMRPOLYGON16) - sizeof(POINT16)) <
3569         cpts * sizeof(POINT16) ) {
3570         throw std::runtime_error( "Invalid record size" );
3571     }
3572
3573     std::unique_ptr<POINT16[]> buffer( new POINT16[cpts] );
3574
3575     POINT16ARRAY points( buffer.get(), cpts );
3576
3577     ds » points;
3578
3579     lpoints = buffer.release();
3580 }
3584 ~EMRPOLYGON16 ( )
3585 {
3586     if ( lpoints ) delete[] lpoints;
3587 }
3591 bool serialize ( DATASTREAM ds )
3592 {
3593     ds « emr « rclBounds « cpts « POINT16ARRAY( lpoints, cpts );
3594     return true;
3595 }
3599 int size ( void )const { return emr.nSize; }
3605 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3606 {
3607     EMF_UNUSED(source);
3608     // According to the wine windef.h header, POINT and POINTL are equivalent
3609     Polygon16( dc, lpoints, cpts );
3610 }
3611 #ifdef ENABLE_EDITING
3615 void edit ( void )const
3616 {
3617     printf( "*POLYGON16*\n" );
3618     edit_rectl( "rclBounds", rclBounds );
3619     edit_point16array( "\t", cpts, lpoints );
3620 }
3621 #endif /* ENABLE_EDITING */
3622 };
3623
3625
3628 class EMRPOLYPOLYGON : public METARECORD, ::EMRPOLYPOLYGON {
3629     DWORD* lcounts{ nullptr };
3630     POINTL* lpoints{ nullptr };
3631 public:
3638     EMRPOLYPOLYGON ( const RECTL* bounds, const POINT* points, const INT* counts,
3639                     UINT polygons )
3640     {
3641         nPolys = polygons;
3642         // Count the number of points in points
3643         int n = 0;
3644         for ( unsigned int i = 0; i < nPolys; i++ )
3645             n += counts[i];
3646
3647         cptl = n;
3648         aPolyCounts[0] = 0; // Really unused
3649         aptl[0].x = 0;
3650         aptl[0].y = 0;
3651
3652         emr.iType = EMR_POLYPOLYGON;
3653         // The (#-1)'s below are to account for aPolyCounts[0] and aptl[0], which
3654         // aren't directly written out

```



```

3655     emr.nSize = sizeof( ::EMRPOLYPOLYGON ) + sizeof( POINTL ) * (cptl-1)
3656 + sizeof( DWORD ) * (nPolys-1);
3657
3658     lcounts = new DWORD[nPolys];
3659
3660     for ( unsigned int i = 0; i < nPolys; i++ )
3661     lcounts[i] = counts[i];
3662
3663     lpoints = new POINTL[cptl];
3664
3665     for (int i=0; i<n; i++) {
3666     lpoints[i].x = points[i].x;
3667     lpoints[i].y = points[i].y;
3668     }
3669
3670     rclBounds = *bounds;
3671 }
3672 ~EMRPOLYPOLYGON ( )
3673 {
3674     if ( lcounts ) delete[] lcounts;
3675     if ( lpoints ) delete[] lpoints;
3676 }
3677 EMRPOLYPOLYGON ( DATASTREAM& ds )
3678 {
3679     ds » emr » rclBounds » nPolys » cptl;
3680
3681     if ( emr.nSize - ( sizeof( ::EMRPOLYPOLYGON ) - sizeof(POINTL) - sizeof(DWORD) ) <
3682           sizeof( POINTL ) * cptl + sizeof( DWORD ) * nPolys ) {
3683         throw std::runtime_error( "Invalid record size" );
3684     }
3685
3686     std::unique_ptr<DWORD[]> cbuffer( new DWORD[nPolys] );
3687
3688     DWORDARRAY counts( cbuffer.get(), nPolys );
3689
3690     ds » counts;
3691
3692     // Counts have to add up to less than the number of points
3693     // we have.  DWORD is unsigned so we must care about overflow.
3694     DWORD n{0}, n_old{0};
3695     for ( DWORD c{0}; c < nPolys; ++c ) {
3696         n_old = n;
3697         n += cbuffer[c];
3698         if ( n < n_old ) {
3699             throw std::runtime_error( "Unsigned overflow" );
3700         }
3701     }
3702     if ( n > cptl ) {
3703         throw std::runtime_error( "Too few points" );
3704     }
3705
3706     std::unique_ptr<POINTL[]> pBuffer( new POINTL[cptl] );
3707
3708     POINTLARRAY points( pBuffer.get(), cptl );
3709
3710     ds » points;
3711
3712     // Don't do this until we won't have any more exceptions.
3713     lcounts = cbuffer.release();
3714     lpoints = pBuffer.release();
3715 }
3716 bool serialize ( DATASTREAM ds )
3717 {
3718     ds « emr « rclBounds « nPolys « cptl « DWORDARRAY( lcounts, nPolys )
3719     « POINTLARRAY( lpoints, cptl );
3720     return true;
3721 }
3722 int size ( void )const { return emr.nSize; }
3723 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3724 {
3725     EMF_UNUSED(source);
3726     // According to the wine windef.h header, POINT and POINTL are equivalent
3727     // (but DWORD and INT are not)
3728     std::vector<INT> countsv( lcounts, lcounts + nPolys );
3729
3730     PolyPolygon( dc, (POINT*)lpoints, countsv.data(), nPolys );
3731 }
3732 #ifdef ENABLE_EDITING
3733 void edit ( void )const
3734 {
3735     #if defined(__LP64__)
3736     const char* FMT0 = "\tnPolys\t\t: %d\n";
3737     const char* FMT1 = "\tcptl\t\t: %d\n";
3738     const char* FMT2 = "%d\n";
3739     const char* FMT3 = "\t\t\t %d\n";
3740     const char* FMT4 = "%d, %d\n";
3741     const char* FMT5 = "\t\t\t %d, %d\n";

```

```

3763 #else
3764     const char* FMT0 = "\tnPolys\t\t: %ld\n";
3765     const char* FMT1 = "\tcptl\t\t: %ld\n";
3766     const char* FMT2 = "%ld\n";
3767     const char* FMT3 = "\t\t\t\t %ld\n";
3768     const char* FMT4 = "%ld, %ld\n";
3769     const char* FMT5 = "\t\t\t\t %ld, %ld\n";
3770 #endif /* __x86_64__ */
3771     printf( " *POLYPOLYGON*\n" );
3772     edit_rectl( "rclBounds", rclBounds );
3773     printf( FMT0, nPolys );
3774     printf( FMT1, cptl );
3775     printf( "\taPolyCounts\t: " );
3776     if ( nPolys > 0 )
3777         printf( FMT2, lcounts[0] );
3778     else
3779         puts( "" );
3780     for ( unsigned int i = 1; i < nPolys; i++ )
3781         printf( FMT3, lcounts[i] );
3782     printf( "\tapt\t\t: " );
3783     if ( cptl > 0 )
3784         printf( FMT4, lpoints[0].x, lpoints[0].y );
3785     else
3786         puts( "" );
3787     for ( unsigned int i = 1; i < cptl; i++ )
3788         printf( FMT5, lpoints[i].x, lpoints[i].y );
3789 }
3790 #endif /* ENABLE_EDITING */
3791 };
3792
3793
3794
3795 class EMRPOLYPOLYGON16 : public METARECORD, ::EMRPOLYPOLYGON16 {
3796     DWORD* lcounts{ nullptr };
3797     POINT16* lpoints{ nullptr };
3798 public:
3799     EMRPOLYPOLYGON16 ( const RECTL* bounds, const POINT* points,
3800                       const INT* counts, UINT polygons )
3801     {
3802         nPolys = polygons;
3803         // Count the number of points in points
3804         int n = 0;
3805         for ( unsigned int i = 0; i < nPolys; i++ )
3806             n += counts[i];
3807
3808         cpts = n;
3809         aPolyCounts[0] = 0; // Really unused
3810         aptl[0].x = 0;
3811         aptl[0].y = 0;
3812
3813         emr.iType = EMR_POLYPOLYGON16;
3814         // The (#-1)'s below are to account for aPolyCounts[0] and aptl[0], which
3815         // aren't directly written out
3816         emr.nSize = sizeof( ::EMRPOLYPOLYGON16 ) + sizeof( POINT16 ) * (cpts-1)
3817         + sizeof( DWORD ) * (nPolys-1);
3818
3819         lcounts = new DWORD[nPolys];
3820
3821         for ( unsigned int i = 0; i < nPolys; i++ )
3822             lcounts[i] = counts[i];
3823
3824         lpoints = new POINT16[cpts];
3825
3826         for ( int i=0; i<n; i++ ) {
3827             lpoints[i].x = points[i].x;
3828             lpoints[i].y = points[i].y;
3829         }
3830
3831         rclBounds = *bounds;
3832     }
3833
3834     EMRPOLYPOLYGON16 ( const RECTL* bounds, const POINT16* points,
3835                       const INT* counts, UINT16 polygons )
3836     {
3837         nPolys = polygons;
3838         // Count the number of points in points
3839         int n = 0;
3840         for ( unsigned int i = 0; i < nPolys; i++ )
3841             n += counts[i];
3842
3843         cpts = n;
3844         aPolyCounts[0] = 0; // Really unused
3845         aptl[0].x = 0;
3846         aptl[0].y = 0;
3847
3848         emr.iType = EMR_POLYPOLYGON16;
3849         // The (#-1)'s below are to account for aPolyCounts[0] and aptl[0], which
3850         // aren't directly written out
3851         emr.nSize = sizeof( ::EMRPOLYPOLYGON16 ) + sizeof( POINT16 ) * (cpts-1)

```

```

3866     + sizeof( DWORD ) * (nPolys-1);
3867
3868     lcounts = new DWORD[nPolys];
3869
3870     for ( unsigned int i = 0; i < nPolys; i++ )
3871     lcounts[i] = counts[i];
3872
3873     lpoints = new POINT16[cpts];
3874
3875     for (int i=0; i<n; i++) {
3876     lpoints[i].x = points[i].x;
3877     lpoints[i].y = points[i].y;
3878     }
3879
3880     rclBounds = *bounds;
3881 }
3882 ~EMRPOLYPOLYGON16 ( )
3883 {
3884     if ( lcounts ) delete[] lcounts;
3885     if ( lpoints ) delete[] lpoints;
3886 }
3887 EMRPOLYPOLYGON16 ( DATASTREAM& ds )
3888 {
3889     ds » emr » rclBounds » nPolys » cpts;
3890
3891     if ( emr.nSize - ( sizeof( ::EMRPOLYPOLYGON16 ) - sizeof(POINT16) - sizeof(DWORD) ) <
3892           sizeof( POINT16 ) * cpts + sizeof( DWORD ) * nPolys ) {
3893         throw std::runtime_error( "Invalid record size" );
3894     }
3895
3896     std::unique_ptr<DWORD[]> cbuffer( new DWORD[nPolys] );
3897
3898     DWORDARRAY counts( cbuffer.get(), nPolys );
3899
3900     ds » counts;
3901
3902     // Counts have to add up to less than the number of points
3903     // we have.  DWORD is unsigned so we must care about overflow.
3904     DWORD n{0}, n_old{0};
3905     for ( DWORD c{0}; c < nPolys; ++c ) {
3906         n_old = n;
3907         n += cbuffer[c];
3908         if ( n < n_old ) {
3909             throw std::runtime_error( "Unsigned overflow" );
3910         }
3911     }
3912     if ( n > cpts ) {
3913         throw std::runtime_error( "Too few points" );
3914     }
3915
3916     std::unique_ptr<POINT16[]> pBuffer( new POINT16[cpts] );
3917
3918     POINT16ARRAY points( pBuffer.get(), cpts );
3919
3920     ds » points;
3921
3922     lcounts = cbuffer.release();
3923     lpoints = pBuffer.release();
3924 }
3925 bool serialize ( DATASTREAM ds )
3926 {
3927     ds « emr « rclBounds « nPolys « cpts « DWORDARRAY( lcounts, nPolys )
3928     « POINT16ARRAY( lpoints, cpts );
3929     return true;
3930 }
3931 int size ( void )const { return emr.nSize; }
3932 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
3933 {
3934     EMF_UNUSED(source);
3935     // According to the wine windef.h header, POINT and POINTL are equivalent
3936     // (but DWORD and INT are not)
3937     std::vector<INT> counts( lcounts, lcounts + nPolys );
3938
3939     PolyPolygon16( dc, lpoints, counts.data(), nPolys );
3940 }
3941 #ifdef ENABLE_EDITING
3942 void edit ( void )const
3943 {
3944     #if defined(__LP64__)
3945         const char* FMT0 = "\tnPolys\t\t: %d\n";
3946         const char* FMT1 = "\tcptl\t\t: %d\n";
3947         const char* FMT2 = "%d\n";
3948         const char* FMT3 = "\t\t\t %d\n";
3949     #else
3950         const char* FMT0 = "\tnPolys\t\t: %ld\n";
3951         const char* FMT1 = "\tcptl\t\t: %ld\n";
3952         const char* FMT2 = "%ld\n";
3953     #endif
3954 }

```

```

3974     const char* FMT3 = "\\t\\t\\t  %ld\\n";
3975 #endif /* __x86_64__ */
3976     printf( "*POLYPOLYGON16*\\n" );
3977     edit_rectl( "rclBounds", rclBounds );
3978     printf( FMT0, nPolys );
3979     printf( FMT1, cpts );
3980     printf( "\\taPolyCounts\\t:  " );
3981     if ( nPolys > 0 )
3982     printf( FMT2, lcounts[0] );
3983     else
3984     puts( "" );
3985     for ( unsigned int i = 1; i < nPolys; i++ )
3986     printf( FMT3, lcounts[i] );
3987     printf( "\\taps\\t\\t:  " );
3988     if ( cpts > 0 )
3989     printf( "%d, %d\\n", lpoints[0].x, lpoints[0].y );
3990     else
3991     puts( "" );
3992     for ( unsigned int i = 1; i < cpts; i++ )
3993     printf( "\\t\\t\\t  %d, %d\\n", lpoints[i].x, lpoints[i].y );
3994     }
3995 #endif /* ENABLE_EDITING */
3996 };
3997
3998
4002 class EMPOLYBEZIER : public METARECORD, ::EMPOLYBEZIER {
4003     POINTL* lpoints{ nullptr };
4004 public:
4005     EMPOLYBEZIER ( const RECTL* bounds, const POINT* points, INT n )
4006     {
4007         cptl = n;
4008         aptl[0].x = 0;           // Really unused
4009         aptl[0].y = 0;
4010
4011         emr.iType = EMR_POLYBEZIER;
4012         // The (cptl-1) below is to account for aptl, which isn't written out
4013         emr.nSize = sizeof( ::EMPOLYBEZIER ) + sizeof( POINTL ) * (cptl-1);
4014
4015         lpoints = new POINTL[cptl];
4016
4017         for (int i=0; i<n; i++) {
4018             lpoints[i].x = points[i].x;
4019             lpoints[i].y = points[i].y;
4020         }
4021
4022         rclBounds = *bounds;
4023     }
4024     EMPOLYBEZIER ( DATASTREAM& ds )
4025     {
4026         ds >> emr >> rclBounds >> cptl;
4027
4028         if ( emr.nSize - (sizeof( ::EMPOLYBEZIER ) - sizeof(POINTL)) <
4029             sizeof( POINTL ) * cptl ) {
4030             throw std::runtime_error( "Invalid record size" );
4031         }
4032
4033         std::unique_ptr<POINTL[]> buffer( new POINTL[cptl] );
4034
4035         POINTLARRAY points( buffer.get(), cptl );
4036
4037         ds >> points;
4038
4039         lpoints = buffer.release();
4040     }
4041     ~EMPOLYBEZIER ( )
4042     {
4043         if ( lpoints ) delete[] lpoints;
4044     }
4045     bool serialize ( DATASTREAM ds )
4046     {
4047         ds << emr << rclBounds << cptl << POINTLARRAY( lpoints, cptl );
4048         return true;
4049     }
4050     int size ( void )const { return emr.nSize; }
4051     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4052     {
4053         EMF_UNUSED(source);
4054         // According to the wine windef.h header, POINT and POINTL are equivalent
4055         PolyBezier( dc, (POINT*)lpoints, cptl );
4056     }
4057 #ifdef ENABLE_EDITING
4058     void edit ( void )const
4059     {
4060         printf( "*POLYBEZIER*\\n" );
4061         edit_rectl( "rclBounds", rclBounds );
4062         edit_pointlarray( "\\t", cptl, lpoints );
4063     }

```

```

4090 #endif /* ENABLE_EDITING */
4091 };
4092
4093
4094
4097 class EMRPOLYBEZIER16 : public METARECORD, ::EMRPOLYBEZIER16 {
4098     POINT16* lpoints{ nullptr };
4099 public:
4100     EMRPOLYBEZIER16 ( const RECTL* bounds, const POINT16* points, INT n )
4101     {
4102         cpts = n;
4103         apts[0].x = 0;           // Really unused
4104         apts[0].y = 0;
4105
4106         emr.iType = EMR_POLYBEZIER16;
4107         // The (cpts-1) below is to account for aptl, which isn't written out
4108         emr.nSize = sizeof( ::EMRPOLYBEZIER16 ) + sizeof( POINT16 ) * (cpts-1);
4109
4110         lpoints = new POINT16[cpts];
4111
4112         for (int i=0; i<n; i++) {
4113             lpoints[i].x = points[i].x;
4114             lpoints[i].y = points[i].y;
4115         }
4116
4117         rclBounds = *bounds;
4118     }
4119     EMRPOLYBEZIER16 ( const RECTL* bounds, const POINT* points, INT n )
4120     {
4121         cpts = n;
4122         apts[0].x = 0;           // Really unused
4123         apts[0].y = 0;
4124
4125         emr.iType = EMR_POLYBEZIER16;
4126         // The (cpts-1) below is to account for aptl, which isn't written out
4127         emr.nSize = sizeof( ::EMRPOLYBEZIER16 ) + sizeof( POINT16 ) * (cpts-1);
4128
4129         lpoints = new POINT16[cpts];
4130
4131         for (int i=0; i<n; i++) {
4132             lpoints[i].x = points[i].x;
4133             lpoints[i].y = points[i].y;
4134         }
4135
4136         rclBounds = *bounds;
4137     }
4138     EMRPOLYBEZIER16 ( DATASTREAM& ds )
4139     {
4140         ds >> emr >> rclBounds >> cpts;
4141
4142         if ( emr.nSize - (sizeof( ::EMRPOLYBEZIER16 ) - sizeof(POINT16)) <
4143             sizeof( POINT16 ) * cpts ) {
4144             throw std::runtime_error( "Invalid record size" );
4145         }
4146
4147         std::unique_ptr<POINT16[]> buffer( new POINT16[cpts] );
4148
4149         POINT16ARRAY points( buffer.get(), cpts );
4150
4151         ds >> points;
4152
4153         lpoints = buffer.release();
4154     }
4155     ~EMRPOLYBEZIER16 ( )
4156     {
4157         if ( lpoints ) delete[] lpoints;
4158     }
4159     bool serialize ( DATASTREAM ds )
4160     {
4161         ds << emr << rclBounds << cpts << POINT16ARRAY( lpoints, cpts );
4162         return true;
4163     }
4164     int size ( void )const { return emr.nSize; }
4165     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4166     {
4167         EMF_UNUSED(source);
4168         // According to the wine windef.h header, POINT and POINTL are equivalent
4169         PolyBezier16( dc, lpoints, cpts );
4170     }
4171 #ifdef ENABLE_EDITING
4172     void edit ( void )const
4173     {
4174         printf( "*POLYBEZIER16*\n" );
4175         edit_rectl( "rclBounds", rclBounds );
4176         edit_point16array( "\t", cpts, lpoints );
4177     }
4178 #endif /* ENABLE_EDITING */
4179 };

```

```

4212
4213
4214
4217 class EMPOLYBEZIERTO : public METARECORD, ::EMRPOLYBEZIER {
4218     POINTL* lpoints{ nullptr };
4219 public:
4225     EMPOLYBEZIERTO ( const RECTL* bounds, const POINT* points, INT n )
4226     {
4227         cptl = n;
4228         aptl[0].x = 0;           // Really unused
4229         aptl[0].y = 0;
4230
4231         emr.iType = EMR_POLYBEZIERTO;
4232         // The (cptl-1) below is to account for aptl, which isn't written out
4233         emr.nSize = sizeof( ::EMRPOLYBEZIERTO ) + sizeof( POINTL ) * (cptl-1);
4234
4235         lpoints = new POINTL[cptl];
4236
4237         for (int i=0; i<n; i++) {
4238             lpoints[i].x = points[i].x;
4239             lpoints[i].y = points[i].y;
4240         }
4241
4242         rclBounds = *bounds;
4243     }
4248     EMPOLYBEZIERTO ( DATASTREAM& ds )
4249     {
4250         ds » emr » rclBounds » cptl;
4251
4252         if ( emr.nSize - (sizeof( ::EMRPOLYBEZIERTO ) - sizeof(POINTL)) <
4253             sizeof( POINTL ) * cptl ) {
4254             throw std::runtime_error( "Invalid record size" );
4255         }
4256
4257         std::unique_ptr<POINTL[]> buffer( new POINTL[cptl] );
4258
4259         POINTLARRAY points( buffer.get(), cptl );
4260
4261         ds » points;
4262
4263         lpoints = buffer.release();
4264     }
4268     ~EMPOLYBEZIERTO ( )
4269     {
4270         if ( lpoints ) delete[] lpoints;
4271     }
4275     bool serialize ( DATASTREAM ds )
4276     {
4277         ds « emr « rclBounds « cptl « POINTLARRAY( lpoints, cptl );
4278         return true;
4279     }
4283     int size ( void )const { return emr.nSize; }
4289     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4290     {
4291         EMF_UNUSED(source);
4292         // According to the wine windef.h header, POINT and POINTL are equivalent
4293         PolyBezierTo( dc, (POINT*)lpoints, cptl );
4294     }
4295 #ifdef ENABLE_EDITING
4299     void edit ( void )const
4300     {
4301         printf( "*POLYBEZIERTO*\n" );
4302         edit_rectl( "rclBounds", rclBounds );
4303         edit_pointlarray( "\t", cptl, lpoints );
4304     }
4305 #endif /* ENABLE_EDITING */
4306 };
4307
4308
4309
4312 class EMPOLYBEZIERTO16 : public METARECORD, ::EMRPOLYBEZIER16 {
4313     POINT16* lpoints{ nullptr };
4314 public:
4320     EMPOLYBEZIERTO16 ( const RECTL* bounds, const POINT16* points, INT n )
4321     {
4322         cpts = n;
4323         apts[0].x = 0;           // Really unused
4324         apts[0].y = 0;
4325
4326         emr.iType = EMR_POLYBEZIERTO16;
4327         // The (cptl-1) below is to account for aptl, which isn't written out
4328         emr.nSize = sizeof( ::EMRPOLYBEZIERTO16 ) + sizeof( POINT16 ) * (cpts-1);
4329
4330         lpoints = new POINT16[cpts];
4331
4332         for (int i=0; i<n; i++) {
4333             lpoints[i].x = points[i].x;
4334             lpoints[i].y = points[i].y;
4335         }

```

```

4336     rclBounds = *bounds;
4337 }
4338 EMRPOLYBEZIERTO16 ( const RECTL* bounds, const POINT* points, INT n )
4339 {
4340     cpts = n;
4341     apts[0].x = 0;           // Really unused
4342     apts[0].y = 0;
4343
4344     emr.iType = EMR_POLYBEZIERTO16;
4345     // The (cptl-1) below is to account for aptl, which isn't written out
4346     emr.nSize = sizeof( ::EMRPOLYBEZIERTO16 ) + sizeof( POINT16 ) * (cpts-1);
4347
4348     lpoints = new POINT16[cpts];
4349
4350     for (int i=0; i<n; i++) {
4351         lpoints[i].x = points[i].x;
4352         lpoints[i].y = points[i].y;
4353     }
4354
4355     rclBounds = *bounds;
4356 }
4357 EMRPOLYBEZIERTO16 ( DATASTREAM& ds )
4358 {
4359     ds » emr » rclBounds » cpts;
4360
4361     if ( emr.nSize - (sizeof( ::EMRPOLYBEZIERTO16 ) - sizeof(POINT16)) <
4362         sizeof( POINT16 ) * cpts ) {
4363         throw std::runtime_error( "Invalid record size" );
4364     }
4365
4366     std::unique_ptr<POINT16[]> buffer( new POINT16[cpts] );
4367
4368     POINT16ARRAY points( buffer.get(), cpts );
4369
4370     ds » points;
4371
4372     lpoints = buffer.release();
4373 }
4374 ~EMRPOLYBEZIERTO16 ( )
4375 {
4376     if ( lpoints ) delete[] lpoints;
4377 }
4378 bool serialize ( DATASTREAM ds )
4379 {
4380     ds « emr « rclBounds « cpts « POINT16ARRAY( lpoints, cpts );
4381     return true;
4382 }
4383 int size ( void )const { return emr.nSize; }
4384 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4385 {
4386     EMF_UNUSED(source);
4387     // According to the wine windef.h header, POINT and POINTL are equivalent
4388     PolyBezierTo16( dc, lpoints, cpts );
4389 }
4390 #ifdef ENABLE_EDITING
4391 void edit ( void )const
4392 {
4393     printf( "*POLYBEZIERTO16*\n" );
4394     edit_rectl( "rclBounds", rclBounds );
4395     edit_point16array( "\t", cpts, lpoints );
4396 }
4397 #endif /* ENABLE_EDITING */
4398 };
4399
4400 class EMRPOLYLINETO : public METARECORD, ::EMRPOLYLINETO {
4401     POINTL* lpoints{ nullptr };
4402 public:
4403     EMRPOLYLINETO ( const RECTL* bounds, const POINT* points, INT n )
4404     {
4405         cptl = n;
4406         aptl[0].x = 0;
4407         aptl[0].y = 0;
4408
4409         emr.iType = EMR_POLYLINETO;
4410         // The (cptl-1) below is to account for aptl, which isn't written out
4411         emr.nSize = sizeof( ::EMRPOLYLINETO ) + sizeof( POINTL ) * (cptl-1);
4412
4413         lpoints = new POINTL[cptl];
4414
4415         for (int i=0; i<n; i++) {
4416             lpoints[i].x = points[i].x;
4417             lpoints[i].y = points[i].y;
4418         }
4419
4420         rclBounds = *bounds;

```

```

4458     }
4463     EMRPOLYLINETO ( DATASTREAM& ds )
4464     {
4465         ds » emr » rclBounds » cptl;
4466
4467         if ( emr.nSize - (sizeof( ::EMRPOLYLINETO ) - sizeof(POINTL)) <
4468             sizeof( POINTL ) * cptl ) {
4469             throw std::runtime_error( "Invalid record size" );
4470         }
4471
4472         std::unique_ptr<POINTL[]> buffer( new POINTL[cptl] );
4473
4474         POINTLARRAY points( buffer.get(), cptl );
4475
4476         ds » points;
4477
4478         lpoints = buffer.release();
4479     }
4483     ~EMRPOLYLINETO ( )
4484     {
4485         if ( lpoints ) delete[] lpoints;
4486     }
4490     bool serialize ( DATASTREAM ds )
4491     {
4492         ds « emr « rclBounds « cptl « POINTLARRAY( lpoints, cptl );
4493         return true;
4494     }
4498     int size ( void )const { return emr.nSize; }
4504     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4505     {
4506         EMF_UNUSED(source);
4507         // According to the wine windef.h header, POINT and POINTL are equivalent
4508         PolylineTo( dc, (POINT*)lpoints, cptl );
4509     }
4510     #ifdef ENABLE_EDITING
4514     void edit ( void )const
4515     {
4516         printf( "*POLYLINETO*\n" );
4517         edit_rectl( "rclBounds", rclBounds );
4518         edit_pointlarray( "\t", cptl, lpoints );
4519     }
4520     #endif /* ENABLE_EDITING */
4521 };
4522
4524
4527     class EMRPOLYLINETO16 : public METARECORD, ::EMRPOLYLINETO16 {
4528     POINT16* lpoints{ nullptr };
4529     public:
4535         EMRPOLYLINETO16 ( const RECTL* bounds, const POINT16* points, INT n )
4536         {
4537             cpts = n;
4538             apts[0].x = 0;
4539             apts[0].y = 0;
4540
4541             emr.iType = EMR_POLYLINETO16;
4542             // The (cptl-1) below is to account for aptl, which isn't written out
4543             emr.nSize = sizeof( ::EMRPOLYLINETO16 ) + sizeof( POINT16 ) * (cpts-1);
4544
4545             lpoints = new POINT16[cpts];
4546
4547             for (int i=0; i<n; i++) {
4548                 lpoints[i].x = points[i].x;
4549                 lpoints[i].y = points[i].y;
4550             }
4551
4552             rclBounds = *bounds;
4553         }
4560         EMRPOLYLINETO16 ( const RECTL* bounds, const POINT* points, INT n )
4561         {
4562             cpts = n;
4563             apts[0].x = 0;
4564             apts[0].y = 0;
4565
4566             emr.iType = EMR_POLYLINETO16;
4567             // The (cptl-1) below is to account for aptl, which isn't written out
4568             emr.nSize = sizeof( ::EMRPOLYLINETO16 ) + sizeof( POINT16 ) * (cpts-1);
4569
4570             lpoints = new POINT16[cpts];
4571
4572             for (int i=0; i<n; i++) {
4573                 lpoints[i].x = points[i].x;
4574                 lpoints[i].y = points[i].y;
4575             }
4576
4577             rclBounds = *bounds;
4578         }
4583         EMRPOLYLINETO16 ( DATASTREAM& ds )

```



```

4584     {
4585         ds » emr » rclBounds » cpts;
4586
4587         if ( emr.nSize - (sizeof( ::EMRPOLYLINE16 ) - sizeof(POINT16)) <
4588             sizeof( POINT16 ) * cpts ) {
4589             throw std::runtime_error( "Invalid record size" );
4590         }
4591
4592         std::unique_ptr<POINT16[]> buffer( new POINT16[cpts] );
4593
4594         POINT16ARRAY points( buffer.get(), cpts );
4595
4596         ds » points;
4597
4598         lpoints = buffer.release();
4599     }
4600     ~EMRPOLYLINE16 ( )
4601     {
4602         if ( lpoints ) delete[] lpoints;
4603     }
4604     bool serialize ( DATASTREAM ds )
4605     {
4606         ds « emr « rclBounds « cpts « POINT16ARRAY( lpoints, cpts );
4607         return true;
4608     }
4609     int size ( void )const { return emr.nSize; }
4610     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4611     {
4612         EMF_UNUSED(source);
4613         // According to the wine windef.h header, POINT and POINTL are equivalent
4614         PolylineTo16( dc, lpoints, cpts );
4615     }
4616 #ifdef ENABLE_EDITING
4617     void edit ( void )const
4618     {
4619         printf( "POLYLINE16*\n" );
4620         edit_rectl( "rclBounds", rclBounds );
4621         edit_point16array( "\t", cpts, lpoints );
4622     }
4623 #endif /* ENABLE_EDITING */
4624 };
4625
4626 class EMREXTTEXTOUTA : public METARECORD, ::EMREXTTEXTOUTA {
4627     PSTR string_a{ nullptr };
4628     int string_size;
4629
4630     INT* dx_i{ nullptr };
4631 public:
4632     EMREXTTEXTOUTA ( const RECTL* bounds, DWORD graphicsMode, FLOAT xScale,
4633                     FLOAT yScale, const PEMRTEXT text, LPCSTR string,
4634                     const INT* dx )
4635     {
4636         emr.iType = EMR_EXTTEXTOUTA;
4637         emr.nSize = sizeof( ::EMREXTTEXTOUTA );
4638
4639         rclBounds = *bounds;
4640
4641         iGraphicsMode = graphicsMode;
4642         exScale = xScale;
4643         eyScale = yScale;
4644
4645         emrtext = *text;
4646
4647         string_size = ROUND_TO_LONG( emrtext.nChars );
4648
4649         string_a = new CHAR[ string_size ];
4650
4651         memset( string_a, 0, sizeof(CHAR) * string_size );
4652
4653         for ( unsigned int i=0; i<emrtext.nChars; i++ )
4654             string_a[i] = *string++;
4655
4656         emrtext.offString = emr.nSize;
4657         emr.nSize += string_size * sizeof(CHAR);
4658 #if 0
4659         /*
4660         Test only - Problem: Windows requires this dx to be set - at least from 2K on
4661         but to calculate real dx values is hard
4662         For pstoeedit - this is "fixed" now by estimating dx in pstoeedit
4663         */
4664         if ( !dx ) {
4665             int * dxn = new int [string_size];
4666             for (unsigned int i=0; i < string_size; i++) dxn[i] = 10;
4667             dx = dxn;
4668         }
4669 #endif
4670     }
4671 };

```

```

4702
4703     if ( dx ) {
4704
4705         dx_i = new INT[ emrtext.nChars ];
4706
4707         for ( unsigned int i=0; i<emrtext.nChars; i++ )
4708             dx_i[i] = *dx++;
4709
4710         emrtext.offDx = emr.nSize;
4711         emr.nSize += emrtext.nChars * sizeof(INT);
4712     }
4713     else {
4714         emrtext.offDx = 0;
4715         dx_i = 0;
4716     }
4717 }
4722 EMREXTTEXTOUTA ( DATASTREAM& ds )
4723 {
4724     ds » emr » rclBounds » iGraphicsMode » exScale » eyScale » emrtext;
4725
4726     if ( emrtext.nChars > 0 and emrtext.offString == 0 ) {
4727         throw std::runtime_error( "Invalid text specification" );
4728     }
4729
4730     if ( emrtext.nChars > emr.nSize - emrtext.offString ) {
4731         throw std::runtime_error( "Invalid text specification" );
4732     }
4733
4734     std::unique_ptr<char[]> cbuffer;
4735     std::unique_ptr<INT[]> ibuffer;
4736
4737     if ( emrtext.offString != 0 ) {
4738         string_size = ROUND_TO_LONG( emrtext.nChars );
4739
4740         cbuffer.reset( new char[string_size] );
4741
4742         memset( cbuffer.get(), 0, sizeof(CHAR) * string_size );
4743
4744         CHARSTR string( cbuffer.get(), string_size );
4745
4746         ds » string;
4747     }
4748
4749     if ( emrtext.offDx ) {
4750         ibuffer.reset( new INT[emrtext.nChars] );
4751
4752         INTARRAY dx_is( ibuffer.get(), emrtext.nChars );
4753
4754         ds » dx_is;
4755     }
4756
4757     string_a = cbuffer.release();
4758     dx_i = ibuffer.release();
4759 }
4764 ~EMREXTTEXTOUTA ( )
4765 {
4766     if ( string_a ) delete[] string_a;
4767     if ( dx_i ) delete[] dx_i;
4768 }
4772 bool serialize ( DATASTREAM ds )
4773 {
4774     ds « emr « rclBounds « iGraphicsMode « exScale « eyScale
4775     « emrtext « CHARSTR( string_a, string_size );
4776     if ( dx_i )
4777         ds « INTARRAY( dx_i, emrtext.nChars );
4778     return true;
4779 }
4783 int size ( void )const { return emr.nSize; }
4789 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
4790 {
4791     EMF_UNUSED(source);
4792     RECT rect;
4793     rect.left = emrtext.rcl.left;
4794     rect.top = emrtext.rcl.top;
4795     rect.right = emrtext.rcl.right;
4796     rect.bottom = emrtext.rcl.bottom;
4797
4798     ExtTextOutA( dc, emrtext.ptlReference.x, emrtext.ptlReference.y,
4799         emrtext.fOptions, &rect, string_a, emrtext.nChars,
4800         dx_i );
4801 }
4802 #ifdef ENABLE_EDITING
4806     void edit ( void )const
4807 {
4808     #if defined(__LP64__)
4809         const char* FMT0 = "unknown(%d)\n";
4810         const char* FMT1 = "\tptlReference\t:  (%d,%d)\n";

```

```

4811     const char* FMT2 = "\tnChars\t\t:  %d\n";
4812     const char* FMT3 = "\toffString\t:  %d\n";
4813     const char* FMT4 = "\toffDx\t\t:  %d\n";
4814 #else
4815     const char* FMT0 = "unknown(%ld)\n";
4816     const char* FMT1 = "\tptlReference\t:  (%ld,%ld)\n";
4817     const char* FMT2 = "\tnChars\t\t:  %ld\n";
4818     const char* FMT3 = "\toffString\t:  %ld\n";
4819     const char* FMT4 = "\toffDx\t\t:  %ld\n";
4820 #endif /* __x86_64__ */
4821     printf( " *EXTTEXTOUTA*\n" );
4822     edit_rectl( "rclBounds", rclBounds );
4823     printf( "\tiGraphicsMode\t:  " );
4824     switch ( iGraphicsMode ) {
4825     case GM_COMPATIBLE: printf( "GM_COMPATIBLE\n" ); break;
4826     case GM_ADVANCED: printf( "GM_ADVANCED\n" ); break;
4827     default: printf( FMT0, iGraphicsMode );
4828     }
4829     printf( "\texScale\t\t:  %f\n", exScale );
4830     printf( "\teyScale\t\t:  %f\n", eyScale );
4831     printf( FMT1, emrtext.ptlReference.x, emrtext.ptlReference.y );
4832     printf( FMT2, emrtext.nChars );
4833     printf( FMT3, emrtext.offString );
4834     printf( "\tfOptions\t:  " );
4835     if ( emrtext.fOptions == 0 )
4836     printf( "None" );
4837     else {
4838     if ( emrtext.fOptions & ETO_GRAYED ) {
4839     printf( "ETO_GRAYED" );
4840     if ( emrtext.fOptions & ~ETO_GRAYED )
4841     printf( " | " );
4842     }
4843     if ( emrtext.fOptions & ETO_OPAQUE ) {
4844     printf( "ETO_OPAQUE" );
4845     if ( emrtext.fOptions & ~(ETO_GRAYED | ETO_OPAQUE) )
4846     printf( " | " );
4847     }
4848     if ( emrtext.fOptions & ETO_CLIPPED ) {
4849     printf( "ETO_CLIPPED" );
4850     if ( emrtext.fOptions & ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED) )
4851     printf( " | " );
4852     }
4853     if ( emrtext.fOptions & ETO_GLYPH_INDEX ) {
4854     printf( "ETO_GLYPH_INDEX" );
4855     if ( emrtext.fOptions &
4856     ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED | ETO_GLYPH_INDEX) )
4857     printf( " | " );
4858     }
4859     if ( emrtext.fOptions & ETO_RTLREADING ) {
4860     printf( "ETO_RTLREADING" );
4861     if ( emrtext.fOptions &
4862     ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED | ETO_GLYPH_INDEX |
4863     ETO_RTLREADING) )
4864     printf( " | " );
4865     }
4866     if ( emrtext.fOptions & ETO_IGNORELANGUAGE )
4867     printf( "ETO_IGNORELANGUAGE" );
4868     }
4869     printf( "\n" );
4870     edit_rectl( "rcl\t", emrtext.rcl );
4871     printf( FMT4, emrtext.offDx );
4872     printf( "\tString:\n\t\t" );
4873     if ( emrtext.nChars > 0 ) {
4874     for ( DWORD i = 0; i < emrtext.nChars; ++i ) {
4875     putchar( string_a[i] );
4876     }
4877     }
4878     else {
4879     printf( "<empty>" );
4880     }
4881     putchar( '\n' );
4882     if ( emrtext.offDx != 0 ) {
4883     printf( "\tOffsets:\n\t\t" );
4884     for ( unsigned int i = 0; i < emrtext.nChars; i++ )
4885     printf( "%d ", dx_i[i] );
4886     printf( "\n" );
4887     }
4888     }
4889 #endif /* ENABLE_EDITING */
4890 };
4891
4892
4893 class EMREXTTEXTOUTW : public METARECORD, ::EMREXTTEXTOUTW {
4894     PWSTR string_a{ nullptr };
4895     int string_size;
4896
4897     INT* dx_i{ nullptr };
4898 public:

```

```

4912     EMREXTTEXTOUTW ( const RECTL* bounds, DWORD graphicsMode, FLOAT xScale,
4913                     FLOAT yScale, const PEMRTEXT text, LPCWSTR string,
4914                     const INT* dx )
4915     {
4916         emr.iType = EMR_EXTTEXTOUTW;
4917         emr.nSize = sizeof( :EMREXTTEXTOUTW );
4918
4919         rclBounds = *bounds;
4920
4921         iGraphicsMode = graphicsMode;
4922         exScale = xScale;
4923         eyScale = yScale;
4924
4925         emrtext = *text;
4926
4927         string_size = ROUND_TO_LONG( emrtext.nChars );
4928
4929         string_a = new WCHAR[ string_size ];
4930
4931         memset( string_a, 0, sizeof(WCHAR) * string_size );
4932
4933         for ( unsigned int i=0; i<emrtext.nChars; i++ )
4934             string_a[i] = *string++;
4935
4936         emrtext.offString = emr.nSize;
4937         emr.nSize += string_size * sizeof(WCHAR);
4938     #if 0
4939     /*
4940     Test only - Problem: Windows requires this dx to be set - at least from 2K on
4941     but to calculate real dx values is hard
4942     For pstoeedit - this is "fixed" now by estimating dx in pstoeedit
4943     */
4944         if ( !dx ) {
4945             int * dxn = new int [string_size];
4946             for (unsigned int i=0; i < string_size; i++) dxn[i] = 10;
4947             dx = dxn;
4948         }
4949     #endif
4950
4951         if ( dx ) {
4952             dx_i = new INT[ emrtext.nChars ];
4953
4954             for ( unsigned int i=0; i<emrtext.nChars; i++ )
4955                 dx_i[i] = *dx++;
4956
4957             emrtext.offDx = emr.nSize;
4958             emr.nSize += emrtext.nChars * sizeof(INT);
4959         }
4960         else {
4961             emrtext.offDx = 0;
4962             dx_i = 0;
4963         }
4964     }
4965
4966     EMREXTTEXTOUTW ( DATASTREAM& ds )
4967     {
4968         ds » emr » rclBounds » iGraphicsMode » exScale » eyScale » emrtext;
4969
4970         if ( emrtext.nChars > 0 and emrtext.offString == 0 ) {
4971             throw std::runtime_error( "Invalid text specification" );
4972         }
4973
4974         if ( emrtext.nChars > emr.nSize - emrtext.offString ) {
4975             throw std::runtime_error( "Invalid text specification" );
4976         }
4977
4978         std::unique_ptr<WCHAR[]> cbuffer;
4979         std::unique_ptr<INT[]> ibuffer;
4980
4981         if ( emrtext.offString != 0 ) { // So, what is the point of this check?
4982             string_size = ROUND_TO_LONG( emrtext.nChars );
4983
4984             cbuffer.reset( new WCHAR[string_size] );
4985
4986             memset( cbuffer.get(), 0, sizeof(WCHAR) * string_size );
4987
4988             WCHARSTR string( cbuffer.get(), string_size );
4989
4990             ds » string;
4991         }
4992
4993         if ( emrtext.offDx ) {
4994             ibuffer.reset( new INT[ emrtext.nChars ] );
4995
4996             INTARRAY dx_is( ibuffer.get(), emrtext.nChars );
4997
4998             ds » dx_is;

```

```

5003     }
5004
5005     string_a = cbuffer.release();
5006     dx_i = ibuffer.release();
5007 }
5012 ~EMREXTTEXTOUTW ( )
5013 {
5014     if ( string_a ) delete[] string_a;
5015     if ( dx_i ) delete[] dx_i;
5016 }
5020 bool serialize ( DATASTREAM ds )
5021 {
5022     ds « emr « rclBounds « iGraphicsMode « exScale « eyScale
5023     « emrtext « WCHARSTR( string_a, string_size );
5024     if ( dx_i )
5025     ds « INTARRAY( dx_i, emrtext.nChars );
5026     return true;
5027 }
5031 int size ( void )const { return emr.nSize; }
5037 void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5038 {
5039     EMF_UNUSED(source);
5040     RECT rect;
5041     rect.left = emrtext.rcl.left;
5042     rect.top = emrtext.rcl.top;
5043     rect.right = emrtext.rcl.right;
5044     rect.bottom = emrtext.rcl.bottom;
5045
5046     ExtTextOutW( dc, emrtext.ptlReference.x, emrtext.ptlReference.y,
5047                 emrtext.fOptions, &rect, string_a, emrtext.nChars,
5048                 dx_i );
5049 }
5050 #ifdef ENABLE_EDITING
5054     void edit ( void )const
5055     {
5056         #if defined(__LP64__)
5057             const char* FMT0 = "unknown(%d)\n";
5058             const char* FMT1 = "\tptlReference\t: (%d,%d)\n";
5059             const char* FMT2 = "\tnChars\t\t: %d\n";
5060             const char* FMT3 = "\toffString\t: %d\n";
5061             const char* FMT4 = "\toffDx\t\t: %d\n";
5062         #else
5063             const char* FMT0 = "unknown(%ld)\n";
5064             const char* FMT1 = "\tptlReference\t: (%ld,%ld)\n";
5065             const char* FMT2 = "\tnChars\t\t: %ld\n";
5066             const char* FMT3 = "\toffString\t: %ld\n";
5067             const char* FMT4 = "\toffDx\t\t: %ld\n";
5068         #endif /* __x86_64__ */
5069         printf( "*EXTTEXTOUTW*\n" );
5070         edit_rectl( "rclBounds", rclBounds );
5071         printf( "\tiGraphicsMode\t: " );
5072         switch ( iGraphicsMode ) {
5073             case GM_COMPATIBLE: printf( "GM_COMPATIBLE\n" ); break;
5074             case GM_ADVANCED: printf( "GM_ADVANCED\n" ); break;
5075             default: printf( FMT0, iGraphicsMode );
5076         }
5077         printf( "\texScale\t\t: %f\n", exScale );
5078         printf( "\teyScale\t\t: %f\n", eyScale );
5079         printf( FMT1, emrtext.ptlReference.x, emrtext.ptlReference.y );
5080         printf( FMT2, emrtext.nChars );
5081         printf( FMT3, emrtext.offString );
5082         printf( "\tfOptions\t: " );
5083         if ( emrtext.fOptions == 0 )
5084             printf( "None" );
5085         else {
5086             if ( emrtext.fOptions & ETO_GRAYED ) {
5087                 printf( "ETO_GRAYED" );
5088                 if ( emrtext.fOptions & ~ETO_GRAYED )
5089                     printf( " | " );
5090             }
5091             if ( emrtext.fOptions & ETO_OPAQUE ) {
5092                 printf( "ETO_OPAQUE" );
5093                 if ( emrtext.fOptions & ~(ETO_GRAYED | ETO_OPAQUE) )
5094                     printf( " | " );
5095             }
5096             if ( emrtext.fOptions & ETO_CLIPPED ) {
5097                 printf( "ETO_CLIPPED" );
5098                 if ( emrtext.fOptions & ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED) )
5099                     printf( " | " );
5100             }
5101             if ( emrtext.fOptions & ETO_GLYPH_INDEX ) {
5102                 printf( "ETO_GLYPH_INDEX" );
5103                 if ( emrtext.fOptions &
5104                     ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED | ETO_GLYPH_INDEX) )
5105                     printf( " | " );
5106             }
5107             if ( emrtext.fOptions & ETO_RTREADING ) {

```

```

5108     printf( "ETO_RTLREADING" );
5109     if ( emrtext.fOptions &
5110         ~(ETO_GRAYED | ETO_OPAQUE | ETO_CLIPPED | ETO_GLYPH_INDEX |
5111           ETO_RTLREADING) )
5112         printf( " | " );
5113 }
5114 if ( emrtext.fOptions & ETO_IGNORELANGUAGE )
5115     printf( "ETO_IGNORELANGUAGE" );
5116 }
5117 printf( "\n" );
5118 edit_rect1( "rc1\t", emrtext.rc1 );
5119 printf( FMT4, emrtext.offDx );
5120
5121 if ( emrtext.nChars > 0 ) {
5122     // iconv_open arguments are TO, FROM (not the other way around).
5123     iconv_t cvt = iconv_open( "UTF-8", "UTF-16LE" );
5124     std::vector<char> utf8_buffer( emrtext.nChars );
5125     // Cannot predict the space necessary to hold the converted
5126     // string. So, we loop until conversion is complete.
5127     size_t size = emrtext.nChars;
5128     size_t in_bytes_left = emrtext.nChars * sizeof(*string_a);
5129     size_t converted = 0;
5130     char* in_buffer = (char*)string_a;
5131     while ( 1 ) {
5132         char* out_buffer = &utf8_buffer[converted];
5133         size_t out_bytes_left = size - converted;
5134
5135         size_t n = iconv( cvt, &in_buffer, &in_bytes_left,
5136                         &out_buffer, &out_bytes_left );
5137
5138         converted = size - out_bytes_left;
5139
5140         if ( n == (size_t)-1 ) {
5141             if ( errno == E2BIG ) {
5142                 size_t new_size = 2 * utf8_buffer.size();
5143                 utf8_buffer.resize( new_size );
5144                 size = utf8_buffer.size();
5145             }
5146             else {
5147                 // Real conversion error.
5148                 break;
5149             }
5150         }
5151         else {
5152             break;
5153         }
5154     }
5155
5156     iconv_close( cvt );
5157
5158     if ( converted == utf8_buffer.size() )
5159         utf8_buffer.push_back( '\0' );
5160     else
5161         utf8_buffer[converted] = '\0';
5162
5163     printf( "\tString:\n\t\t%s\n", utf8_buffer.data() );
5164 }
5165 else {
5166     puts( "\tString:\n\t\t<empty>\n" );
5167 }
5168
5169 if ( emrtext.offDx != 0 and emrtext.nChars > 0 ) {
5170     printf( "\tOffsets:\n\t\t" );
5171     for ( unsigned int i = 0; i < emrtext.nChars; i++ )
5172         printf( "%d ", dx_i[i] );
5173     printf( "\n" );
5174 }
5175 }
5176 #endif /* ENABLE_EDITING */
5177 };
5178
5180
5183 class EMRSETPIXELV : public METARECORD, ::EMRSETPIXELV {
5184 public:
5185     EMRSETPIXELV ( INT x, INT y, COLORREF color )
5186     {
5187         emr.iType = EMR_SETPIXELV;
5188         emr.nSize = sizeof( ::EMRSETPIXELV );
5189         ptlPixel.x = x;
5190         ptlPixel.y = y;
5191         crColor = color;
5192     }
5193     EMRSETPIXELV ( DATASTREAM& ds )
5194     {
5195         ds » emr » ptlPixel » crColor;
5196     }
5197     bool serialize ( DATASTREAM ds )

```

```

5210     {
5211         ds << emr << ptlPixel << crColor;
5212         return true;
5213     }
5217     int size ( void )const { return emr.nSize; }
5223     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5224 {
5225     EMF_UNUSED(source);
5226     SetPixel( dc, ptlPixel.x, ptlPixel.y, crColor );
5227 }
5228 #ifdef ENABLE_EDITING
5232     void edit ( void )const
5233 {
5234     printf( "*SETPIXELV*\n" );
5235     edit_pointl( "ptlPixel", ptlPixel );
5236     edit_color( "crColor\t", crColor );
5237 }
5238 #endif /* ENABLE_EDITING */
5239 };
5240
5241 class PEN;
5242 class EXTPEN;
5243 class BRUSH;
5244 class FONT;
5245 class PALETTE;
5246
5248
5251 class EMRCREATEPEN : public METARECORD, public ::EMRCREATEPEN
5252 {
5253 public:
5258     EMRCREATEPEN ( PEN* pen, HGDI OBJ handle );
5263     EMRCREATEPEN ( DATASTREAM& ds );
5267     bool serialize ( DATASTREAM ds )
5268     {
5269         ds << emr << ihPen << lopn;
5270         return true;
5271     }
5275     int size ( void )const { return emr.nSize; }
5281     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
5282 #ifdef ENABLE_EDITING
5286     void edit ( void )const
5287 {
5288     #if defined(__LP64__)
5289         const char* FMT0 = "\tihPen\t\t: 0x%x\n";
5290         const char* FMT1 = "\tlopn.lopnWidth\t: %d, %d\n";
5291     #else
5292         const char* FMT0 = "\tihPen\t\t: 0x%x\n";
5293         const char* FMT1 = "\tlopn.lopnWidth\t: %ld, %ld\n";
5294     #endif /* __x86_64__ */
5295     printf( "CREATEPEN*\n" );
5296     printf( FMT0, ihPen );
5297     edit_pen_style( "lopn.lopnStyle", lopn.lopnStyle );
5298     printf( FMT1, lopn.lopnWidth.x, lopn.lopnWidth.y );
5299     edit_color( "lopn.lopnColor", lopn.lopnColor );
5300 }
5301 #endif /* ENABLE_EDITING */
5302 };
5303
5305
5309 class EMREXTCREATEPEN : public METARECORD, public ::EMREXTCREATEPEN
5310 {
5311 public:
5316     EMREXTCREATEPEN ( EXTPEN* pen, HGDI OBJ handle );
5321     EMREXTCREATEPEN ( DATASTREAM& ds );
5325     bool serialize ( DATASTREAM ds )
5326     {
5327         ds << emr << ihPen << offBmi << cbBmi << offBits << cbBits << elp;
5328         return true;
5329     }
5333     int size ( void )const { return emr.nSize; }
5339     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
5340 #ifdef ENABLE_EDITING
5344     void edit ( void )const
5345 {
5346     #if defined(__LP64__)
5347         const char* FMT0 = "\tihPen\t\t\t: 0x%x\n";
5348         const char* FMT1 = "\toffBmi\t\t\t: %d\n";
5349         const char* FMT2 = "\tcbBmi\t\t\t: %d\n";
5350         const char* FMT3 = "\toffBits\t\t\t: %d\n";
5351         const char* FMT4 = "\tcbBits\t\t\t: %d\n";
5352         const char* FMT5 = "\telp.elpWidth\t\t: %d\n";
5353         const char* FMT6 = "\telp.elpNumEntries\t: %d\n";
5354     #else
5355         const char* FMT0 = "\tihPen\t\t\t: 0x%x\n";
5356         const char* FMT1 = "\toffBmi\t\t\t: %ld\n";
5357         const char* FMT2 = "\tcbBmi\t\t\t: %ld\n";
5358         const char* FMT3 = "\toffBits\t\t\t: %ld\n";

```

```

5359     const char* FMT4 = "\tcbBits\t\t\t: %ld\n";
5360     const char* FMT5 = "\telp.elpWidth\t\t: %ld\n";
5361     const char* FMT6 = "\telp.elpNumEntries\t: %ld\n";
5362 #endif /* __x86_64__ */
5363     printf( "EMR_CREATEPEN*\n" );
5364     printf( FMT0, ihPen );
5365     printf( FMT1, offBmi );
5366     printf( FMT2, cbBmi );
5367     printf( FMT3, offBits );
5368     printf( FMT4, cbBits );
5369     edit_pen_style( "elp.elpPenStyle\t", elp.elpPenStyle );
5370     printf( FMT5, elp.elpWidth );
5371     edit_brush_style( "elp.elpBrushStyle", elp.elpBrushStyle );
5372     edit_color( "elp.elpColor\t", elp.elpColor );
5373     edit_brush_hatch( "elp.elpHatch\t", elp.elpHatch );
5374     printf( FMT6, elp.elpNumEntries );
5375 }
5376 #endif /* ENABLE_EDITING */
5377 };
5378
5380
5383 class EMR_CREATEBRUSHINDIRECT : public METARECORD, public ::EMR_CREATEBRUSHINDIRECT
5384 {
5385 public:
5386     EMR_CREATEBRUSHINDIRECT ( BRUSH* brush, HGDI OBJ handle );
5387     EMR_CREATEBRUSHINDIRECT ( DATASTREAM& ds );
5388     bool serialize ( DATASTREAM ds )
5389     {
5390         ds << emr << ihBrush << lb;
5391         return true;
5392     }
5393     int size ( void )const { return emr.nSize; }
5394     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
5395 #ifdef ENABLE_EDITING
5396     void edit ( void )const
5397     {
5398     #if defined(__LP64__)
5399         const char* FMT = "\tihBrush\t\t\t: 0x%x\n";
5400     #else
5401         const char* FMT = "\tihBrush\t\t\t: 0x%lx\n";
5402     #endif /* __x86_64__ */
5403         printf( "EMR_CREATEBRUSHINDIRECT*\n" );
5404         printf( FMT, ihBrush );
5405         edit_brush_style( "lb.lbStyle", lb.lbStyle );
5406         edit_color( "lb.lbColor", lb.lbColor );
5407         edit_brush_hatch( "lb.lbHatch", lb.lbHatch );
5408     }
5409 #endif /* ENABLE_EDITING */
5410 };
5411
5413
5416 class EMR_EXT_CREATEFONTINDIRECTW : public METARECORD, public ::EMR_EXT_CREATEFONTINDIRECTW
5417 {
5418 public:
5419     EMR_EXT_CREATEFONTINDIRECTW ( FONT* font, HGDI OBJ handle );
5420     EMR_EXT_CREATEFONTINDIRECTW ( DATASTREAM& ds );
5421     bool serialize ( DATASTREAM ds )
5422     {
5423         // Since EMF records have to be multiples of 4 bytes, this
5424         // should perhaps be a general thing, but we know it's currently
5425         // only a problem for this structure.
5426
5427         ds << emr << ihFont << elfw << PADDING( 2 );
5428         return true;
5429     }
5430     int size ( void )const { return emr.nSize; }
5431     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
5432 #ifdef ENABLE_EDITING
5433     void edit ( void )const
5434     {
5435     #if defined(__LP64__)
5436         const char* FMT0 = "\tihFont\t\t\t\t: %d\n";
5437         const char* FMT1 = "\tlfHeight\t\t\t: %d\n";
5438         const char* FMT2 = "\tlfWidth\t\t\t\t: %d\n";
5439         const char* FMT3 = "\tlfEscapement\t\t: %d\n";
5440         const char* FMT4 = "\tlfOrientation\t\t: %d\n";
5441         const char* FMT5 = "\tlfVersion\t\t\t: %d\n";
5442         const char* FMT6 = "\tlfStyleSize\t\t: %d\n";
5443         const char* FMT7 = "\tlfMatch\t\t\t\t: %d\n";
5444         const char* FMT8 = "\tlfCulture\t\t\t: %d\n";
5445     #else
5446         const char* FMT0 = "\tihFont\t\t\t\t: %ld\n";
5447         const char* FMT1 = "\tlfHeight\t\t\t: %ld\n";
5448         const char* FMT2 = "\tlfWidth\t\t\t\t: %ld\n";
5449         const char* FMT3 = "\tlfEscapement\t\t: %ld\n";
5450         const char* FMT4 = "\tlfOrientation\t\t: %ld\n";

```



```

5496     const char* FMT5 = "\\telfVersion\\t\\t: %ld\\n";
5497     const char* FMT6 = "\\telfStyleSize\\t\\t: %ld\\n";
5498     const char* FMT7 = "\\telfMatch\\t\\t: %ld\\n";
5499     const char* FMT8 = "\\telfCulture\\t\\t: %ld\\n";
5500 #endif /* __x86_64__ */
5501     printf( "*EXTCREATEFONTINDIRECTW\\n" );
5502     printf( FMT0, ihFont );
5503     printf( FMT1, elfw.elfLogFont.lfHeight );
5504     printf( FMT2, elfw.elfLogFont.lfWidth );
5505     printf( FMT3, elfw.elfLogFont.lfEscapement );
5506     printf( FMT4, elfw.elfLogFont.lfOrientation );
5507     printf( "\\tlfWeight\\t\\t: " );
5508     switch ( elfw.elfLogFont.lfWeight ) {
5509     case FW_DONTCARE: printf( "FW_DONTCARE\\n" ); break;
5510     case FW_THIN: printf( "FW_THIN\\n" ); break;
5511     case FW_EXTRALIGHT: printf( "FW_EXTRALIGHT\\n" ); break;
5512     case FW_LIGHT: printf( "FW_LIGHT\\n" ); break;
5513     case FW_NORMAL: printf( "FW_NORMAL\\n" ); break;
5514     case FW_MEDIUM: printf( "FW_MEDIUM\\n" ); break;
5515     case FW_SEMIBOLD: printf( "FW_SEMIBOLD\\n" ); break;
5516     case FW_BOLD: printf( "FW_BOLD\\n" ); break;
5517     case FW_EXTRABOLD: printf( "FW_EXTRABOLD\\n" ); break;
5518     case FW_BLACK: printf( "FW_BLACK\\n" ); break;
5519     }
5520     printf( "\\tlfItalic\\t\\t: %d\\n", elfw.elfLogFont.lfItalic );
5521     printf( "\\tlfUnderline\\t\\t: %d\\n", elfw.elfLogFont.lfUnderline );
5522     printf( "\\tlfStrikeOut\\t\\t: %d\\n", elfw.elfLogFont.lfStrikeOut );
5523     printf( "\\tlfCharSet\\t\\t: %d\\n", elfw.elfLogFont.lfCharSet );
5524     printf( "\\tlfOutPrecision\\t\\t: %d\\n", elfw.elfLogFont.lfOutPrecision );
5525     printf( "\\tlfClipPrecision\\t\\t: %d\\n", elfw.elfLogFont.lfClipPrecision );
5526     printf( "\\tlfQuality\\t\\t: %d\\n", elfw.elfLogFont.lfQuality );
5527     printf( "\\tlfPitchAndFamily\\t\\t: %d\\n", elfw.elfLogFont.lfPitchAndFamily );
5528     int i = 0;
5529     printf( "\\tlfFaceName\\t\\t: ' " );
5530     while ( elfw.elfLogFont.lfFaceName[i] != 0 && i < LF_FACESIZE ) {
5531     putchar( elfw.elfLogFont.lfFaceName[i] );
5532     i++;
5533     }
5534     puts( "' " );
5535
5536     i = 0;
5537     printf( "\\telfFullName\\t\\t: ' " );
5538     while ( elfw.elfFullName[i] != 0 && i < LF_FULLFACESIZE ) {
5539     putchar( elfw.elfFullName[i] );
5540     i++;
5541     }
5542     puts( "' " );
5543
5544     i = 0;
5545     printf( "\\telfStyle\\t\\t: ' " );
5546     while ( elfw.elfStyle[i] != 0 && i < LF_FACESIZE ) {
5547     putchar( elfw.elfStyle[i] );
5548     i++;
5549     }
5550     puts( "' " );
5551
5552     printf( FMT5, elfw.elfVersion );
5553     printf( FMT6, elfw.elfStyleSize );
5554     printf( FMT7, elfw.elfMatch );
5555     printf( "\\telfVendorId\\t\\t: '%s'\\n", elfw.elfVendorId );
5556     printf( FMT8, elfw.elfCulture );
5557     printf( "\\telfPanose\\t\\t:\\n" );
5558     printf( "\\t\\tbFamilyType\\t\\t: %d\\n", elfw.elfPanose.bFamilyType );
5559     printf( "\\t\\tbSerifStyle\\t\\t: %d\\n", elfw.elfPanose.bSerifStyle );
5560     printf( "\\t\\tbWeight\\t\\t: %d\\n", elfw.elfPanose.bWeight );
5561     printf( "\\t\\tbProportion\\t\\t: %d\\n", elfw.elfPanose.bProportion );
5562     printf( "\\t\\tbContrast\\t\\t: %d\\n", elfw.elfPanose.bContrast );
5563     printf( "\\t\\tbStrokeVariation\\t\\t: %d\\n", elfw.elfPanose.bStrokeVariation );
5564     printf( "\\t\\tbArmStyle\\t\\t: %d\\n", elfw.elfPanose.bArmStyle );
5565     printf( "\\t\\tbLetterform\\t\\t: %d\\n", elfw.elfPanose.bLetterform );
5566     printf( "\\t\\tbMidline\\t\\t: %d\\n", elfw.elfPanose.bMidline );
5567     printf( "\\t\\tbXHeight\\t\\t: %d\\n", elfw.elfPanose.bXHeight );
5568     }
5569 #endif /* ENABLE_EDITING */
5570 };
5571
5572
5573
5574 class EMRCREATEPALETTE : public METARECORD, public ::EMRCREATEPALETTE
5575 {
5576 public:
5577     EMRCREATEPALETTE ( PALETTE* palette, HGDIOBJ handle );
5578     EMRCREATEPALETTE ( DATASTREAM& ds );
5579     bool serialize ( DATASTREAM ds )
5580     {
5581         ds << emr << ihPal << lgpl;
5582         return true;
5583     }
5584 };

```

```

5600     int size ( void )const { return emr.nSize; }
5606     void execute ( METAFILEDEVICECONTEXT* source, HDC dc ) const;
5607 #ifdef ENABLE_EDITING
5611     void edit ( void )const
5612 {
5613     printf( "*CREATEPALETTE* (not really handled by libEMF)\n" );
5614 }
5615 #endif /* ENABLE_EDITING */
5616 };
5617
5619
5622 class EMRFILLPATH : public METARECORD, ::EMRFILLPATH {
5623 public:
5627     EMRFILLPATH ( const RECTL* bounds )
5628     {
5629         emr.iType = EMR_FILLPATH;
5630         emr.nSize = sizeof( ::EMRFILLPATH );
5631         rclBounds = *bounds;
5632     }
5637     EMRFILLPATH ( DATASTREAM& ds )
5638     {
5639         ds » emr » rclBounds;
5640     }
5644     bool serialize ( DATASTREAM ds )
5645     {
5646         ds « emr « rclBounds;
5647         return true;
5648     }
5652     int size ( void )const { return emr.nSize; }
5658     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5659 {
5660     EMF_UNUSED(source);
5661     FillPath( dc );
5662 }
5663 #ifdef ENABLE_EDITING
5667     void edit ( void )const
5668 {
5669     printf( "*FILLPATH*\n" );
5670     edit_rectl( "rclBounds", rclBounds );
5671 }
5672 #endif /* ENABLE_EDITING */
5673 };
5675
5678 class EMRSTROKEPATH : public METARECORD, ::EMRSTROKEPATH {
5679 public:
5683     EMRSTROKEPATH ( const RECTL* bounds )
5684     {
5685         emr.iType = EMR_STROKEPATH;
5686         emr.nSize = sizeof( ::EMRSTROKEPATH );
5687         rclBounds = *bounds;
5688     }
5693     EMRSTROKEPATH ( DATASTREAM& ds )
5694     {
5695         ds » emr » rclBounds;
5696     }
5700     bool serialize ( DATASTREAM ds )
5701     {
5702         ds « emr « rclBounds;
5703         return true;
5704     }
5708     int size ( void )const { return emr.nSize; }
5714     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5715 {
5716     EMF_UNUSED(source);
5717     StrokePath( dc );
5718 }
5719 #ifdef ENABLE_EDITING
5723     void edit ( void )const
5724 {
5725     printf( "*STROKEPATH*\n" );
5726     edit_rectl( "rclBounds", rclBounds );
5727 }
5728 #endif /* ENABLE_EDITING */
5729 };
5731
5734 class EMRSTROKEANDFILLPATH : public METARECORD, ::EMRSTROKEANDFILLPATH {
5735 public:
5739     EMRSTROKEANDFILLPATH ( const RECTL* bounds )
5740     {
5741         emr.iType = EMR_STROKEANDFILLPATH;
5742         emr.nSize = sizeof( ::EMRSTROKEANDFILLPATH );
5743         rclBounds = *bounds;
5744     }
5749     EMRSTROKEANDFILLPATH ( DATASTREAM& ds )
5750     {
5751         ds » emr » rclBounds;
5752     }

```

```

5756     bool serialize ( DATASTREAM ds )
5757     {
5758         ds « emr « rclBounds;
5759         return true;
5760     }
5761     int size ( void )const { return emr.nSize; }
5770     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5771     {
5772         EMF_UNUSED(source);
5773         StrokeAndFillPath( dc );
5774     }
5775 #ifdef ENABLE_EDITING
5779     void edit ( void )const
5780     {
5781         printf( " *STROKEANDFILLPATH*\n" );
5782         edit_rectl( "rclBounds", rclBounds );
5783     }
5784 #endif /* ENABLE_EDITING */
5785 };
5787
5790 class EMRBEGINPATH : public METARECORD, ::EMRBEGINPATH {
5791 public:
5795     EMRBEGINPATH ( void )
5796     {
5797         emr.iType = EMR_BEGINPATH;
5798         emr.nSize = sizeof( ::EMRBEGINPATH );
5799     }
5804     EMRBEGINPATH ( DATASTREAM& ds )
5805     {
5806         ds » emr;
5807     }
5811     bool serialize ( DATASTREAM ds )
5812     {
5813         ds « emr;
5814         return true;
5815     }
5819     int size ( void )const { return emr.nSize; }
5825     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5826     {
5827         EMF_UNUSED(source);
5828         BeginPath( dc );
5829     }
5830 #ifdef ENABLE_EDITING
5834     void edit ( void )const
5835     {
5836         printf( " *BEGINPATH*\n" );
5837     }
5838 #endif /* ENABLE_EDITING */
5839 };
5841
5844 class EMRENDPATH : public METARECORD, ::EMRENDPATH {
5845 public:
5849     EMRENDPATH ( void )
5850     {
5851         emr.iType = EMR_ENDPATH;
5852         emr.nSize = sizeof( ::EMRENDPATH );
5853     }
5858     EMRENDPATH ( DATASTREAM& ds )
5859     {
5860         ds » emr;
5861     }
5865     bool serialize ( DATASTREAM ds )
5866     {
5867         ds « emr;
5868         return true;
5869     }
5873     int size ( void )const { return emr.nSize; }
5879     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5880     {
5881         EMF_UNUSED(source);
5882         EndPath( dc );
5883     }
5884 #ifdef ENABLE_EDITING
5888     void edit ( void )const
5889     {
5890         printf( " *ENDPATH*\n" );
5891     }
5892 #endif /* ENABLE_EDITING */
5893 };
5895
5898 class EMRCLOSEFIGURE : public METARECORD, ::EMRCLOSEFIGURE {
5899 public:
5903     EMRCLOSEFIGURE ( void )
5904     {
5905         emr.iType = EMR_CLOSEFIGURE;
5906         emr.nSize = sizeof( ::EMRCLOSEFIGURE );
5907     }

```

```

5912     EMRCLOSEFIGURE ( DATASTREAM& ds )
5913     {
5914         ds » emr;
5915     }
5919     bool serialize ( DATASTREAM ds )
5920     {
5921         ds « emr;
5922         return true;
5923     }
5927     int size ( void )const { return emr.nSize; }
5933     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5934     {
5935         EMF_UNUSED(source);
5936         CloseFigure( dc );
5937     }
5938     #ifdef ENABLE_EDITING
5942         void edit ( void )const
5943         {
5944             printf( " *CLOSEFIGURE*\n" );
5945         }
5946     #endif /* ENABLE_EDITING */
5947     };
5949
5953     class EMRSAVEDC : public METARECORD, ::EMRSAVEDC {
5954     public:
5958         EMRSAVEDC ( void )
5959         {
5960             emr.iType = EMR_SAVEDC;
5961             emr.nSize = sizeof( ::EMRSAVEDC );
5962         }
5967         EMRSAVEDC ( DATASTREAM& ds )
5968         {
5969             ds » emr;
5970         }
5974         bool serialize ( DATASTREAM ds )
5975         {
5976             ds « emr;
5977             return true;
5978         }
5982         int size ( void )const { return emr.nSize; }
5988         void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
5989         {
5990             EMF_UNUSED(source);
5991             SaveDC( dc );
5992         }
5993         #ifdef ENABLE_EDITING
5997             void edit ( void )const
5998             {
5999                 printf( " *SAVEDC*\n" );
6000             }
6001         #endif /* ENABLE_EDITING */
6002     };
6004
6007     class EMRRESTOREDC : public METARECORD, ::EMRRESTOREDC {
6008     public:
6012         EMRRESTOREDC ( INT n )
6013         {
6014             emr.iType = EMR_RESTOREDC;
6015             emr.nSize = sizeof( ::EMRRESTOREDC );
6016             iRelative = n;
6017         }
6022         EMRRESTOREDC ( DATASTREAM& ds )
6023         {
6024             ds » emr » iRelative;
6025         }
6029         bool serialize ( DATASTREAM ds )
6030         {
6031             ds « emr « iRelative;
6032             return true;
6033         }
6037         int size ( void )const { return emr.nSize; }
6043         void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
6044         {
6045             EMF_UNUSED(source);
6046             RestoreDC( dc, iRelative );
6047         }
6048         #ifdef ENABLE_EDITING
6052             void edit ( void )const
6053             {
6054                 #if defined(__LP64__)
6055                     const char* FMT = "\tiRelative:  %d\n";
6056                 #else
6057                     const char* FMT = "\tiRelative:  %ld\n";
6058                 #endif /* __x86_64__ */
6059                 printf( " *RESTOREDC*\n" );
6060                 printf( FMT, iRelative );
6061             }

```

```

6062 #endif /* ENABLE_EDITING */
6063 };
6064
6065
6066 class EMRSETMETARGN : public METARECORD, ::EMRSETMETARGN {
6067 public:
6073     EMRSETMETARGN ( void )
6074     {
6075         emr.iType = EMR_SETMETARGN;
6076         emr.nSize = sizeof( ::EMRSETMETARGN );
6077     }
6082     EMRSETMETARGN ( DATASTREAM& ds )
6083     {
6084         ds » emr;
6085     }
6089     bool serialize ( DATASTREAM ds )
6090     {
6091         ds « emr;
6092         return true;
6093     }
6097     int size ( void )const { return emr.nSize; }
6103     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
6104     {
6105         EMF_UNUSED(source);
6106         SetMetaRgn( dc );
6107     }
6108 #ifdef ENABLE_EDITING
6112     void edit ( void )const
6113     {
6114         printf( "SETMETARGN*\n" );
6115     }
6116 #endif /* ENABLE_EDITING */
6117 };
6118
6120
6123 class PEN : public GRAPHICSOBJECT, public LOGPEN {
6124 public:
6128     PEN ( const LOGPEN* lpen )
6129     {
6130         lopnStyle = lpen->lopnStyle;
6131         lopnWidth = lpen->lopnWidth;
6132         lopnColor = lpen->lopnColor;
6133     }
6137     OBJECTTYPE getType ( void )const { return O_PEN; }
6144     METARECORD* newEMR ( HDC dc, HGDIOBJ emf_handle )
6145     {
6146         contexts[dc] = emf_handle;
6147         return new EMRCREATEPEN( this, emf_handle );
6148     }
6149 };
6150
6152
6155 class EXTPEN : public GRAPHICSOBJECT, public EXTLOGPEN {
6156 public:
6160     EXTPEN ( const EXTLOGPEN* lpen )
6161     {
6162         elpPenStyle = lpen->elpPenStyle;
6163         elpWidth = lpen->elpWidth;
6164         elpBrushStyle = lpen->elpBrushStyle;
6165         elpColor = lpen->elpColor;
6166         elpHatch = lpen->elpHatch;
6167         elpNumEntries = 0;
6168         elpStyleEntry[0] = 0;
6169     }
6173     OBJECTTYPE getType ( void )const { return O_EXTPEN; }
6180     METARECORD* newEMR ( HDC dc, HGDIOBJ emf_handle )
6181     {
6182         contexts[dc] = emf_handle;
6183         return new EMREXTCREATEPEN( this, emf_handle );
6184     }
6185 };
6186
6188
6191 class BRUSH : public GRAPHICSOBJECT, public LOGBRUSH {
6192 public:
6196     BRUSH ( const LOGBRUSH* lbrush )
6197     {
6198         lbStyle = lbrush->lbStyle;
6199         lbColor = lbrush->lbColor;
6200         lbHatch = lbrush->lbHatch;
6201     }
6205     OBJECTTYPE getType ( void )const { return O_BRUSH; }
6212     METARECORD* newEMR ( HDC dc, HGDIOBJ emf_handle )
6213     {
6214         contexts[dc] = emf_handle;
6215         return new EMRCREATEBRUSHINDIRECT( this, emf_handle );
6216     }
6217 };

```

```

6218
6220
6223 class FONT : public GRAPHICSOBJECT, public EXTLOGFONTW {
6224 public:
6228     FONT ( const LOGFONTW* lfont )
6229     {
6230         this->elfLogFont = *lfont;
6231         // There are a lot more entries in the EXTLOGFONTW structure than
6232         // the API has values for, so we invent them here
6233         memset( &elfFullName, 0, sizeof elfFullName );
6234         memset( &elfStyle, 0, sizeof elfStyle );
6235         elfVersion = ELF_VERSION;
6236         elfStyleSize = 0;
6237         elfMatch = 0;
6238         elfReserved = 0;
6239         memset( &elfVendorId, 0, sizeof elfVendorId );
6240         elfCulture = ELF_CULTURE_LATIN;
6241         memset( &elfPanose, 1, sizeof(PANOSE) );
6242     }
6243     OBJECTTYPE getType ( void )const { return O_FONT; }
6253     METARECORD* newEMR ( HDC dc, HGDIOBJ emf_handle )
6254     {
6255         contexts[dc] = emf_handle;
6256         return new EMREXTCREATEFONTINDIRECTW( this, emf_handle );
6257     }
6258 };
6259
6261
6264 class PALETTE : public GRAPHICSOBJECT, public LOGPALETTE {
6265 public:
6269     PALETTE ( const LOGPALETTE* lpalette )
6270     {
6271         EMF_UNUSED(lpalette);
6272         palVersion = 0;
6273         palNumEntries = 0;
6274         PALETTEENTRY zero_entry = { 0, 0, 0, 0 };
6275         palPalEntry[0] = zero_entry;
6276     }
6280     OBJECTTYPE getType ( void )const { return O_PALETTE; }
6287     METARECORD* newEMR ( HDC dc, HGDIOBJ emf_handle )
6288     {
6289         contexts[dc] = emf_handle;
6290         return new EMRCREATEPALETTE( this, emf_handle );
6291     }
6292 };
6293
6295
6298 class EMRSETMITERLIMIT : public METARECORD, ::EMRSETMITERLIMIT {
6299 public:
6303     EMRSETMITERLIMIT ( FLOAT limit )
6304     {
6305         emr.iType = EMR_SETMITERLIMIT;
6306         emr.nSize = sizeof( ::EMRSETMITERLIMIT );
6307         eMiterLimit = limit;
6308     }
6313     EMRSETMITERLIMIT ( DATASTREAM& ds )
6314     {
6315         int miter_limit;
6316         ds » emr » miter_limit;
6317         eMiterLimit = float(miter_limit);
6318     }
6322     bool serialize ( DATASTREAM ds )
6323     {
6324         ds « emr « (int)eMiterLimit;
6325         return true;
6326     }
6330     int size ( void )const { return emr.nSize; }
6336     void execute ( METAFILEDEVICECONTEXT* source, HDC dc )const
6337     {
6338         EMF_UNUSED(source);
6339         SetMiterLimit( dc, eMiterLimit, 0 );
6340     }
6341 #ifdef ENABLE_EDITING
6345     void edit ( void )const
6346     {
6347         printf( "*SETMITERLIMIT*\n" );
6348         printf( "\teMiterLimit\t:  %f\n", eMiterLimit );
6349     }
6350 #endif /* ENABLE_EDITING */
6351 };
6352
6354
6360 class METAFILEDEVICECONTEXT : public OBJECT {
6368     void init ( const RECT* size, LPCWSTR description_w ) {
6369
6370         // Evidently, metafile handles are numbered from 1, so don't
6371         // ever use 0.

```

```

6372
6373     handles.push_back( true );
6374
6375     // Keep some of our graphics state in a header record
6376
6377     header = new ENHMETAHEADER ( description_w );
6378     records.push_back( header );
6379
6380     // Compute the size and position of the metafile on the "page"
6381
6382     if ( size ) {
6383         update_frame = false;
6384
6385         header->rclFrame.left = size->left;
6386         header->rclFrame.top = size->top;
6387         header->rclFrame.right = size->right;
6388         header->rclFrame.bottom = size->bottom;
6389
6390         header->rclBounds.left =
6391             size->left * header->szlDevice.cx / ( header->szlMillimeters.cx * 100 );
6392         header->rclBounds.top =
6393             size->top * header->szlDevice.cy / ( header->szlMillimeters.cy * 100 );
6394         header->rclBounds.right =
6395             size->right * header->szlDevice.cx / ( header->szlMillimeters.cx * 100 );
6396         header->rclBounds.bottom =
6397             size->bottom * header->szlDevice.cy / ( header->szlMillimeters.cy * 100 );
6398     }
6399     else {
6400         update_frame = true;
6401
6402         header->rclBounds.left = -10;
6403         header->rclBounds.top = -10;
6404         header->rclBounds.right = 10;
6405         header->rclBounds.bottom = 10;
6406
6407         header->rclFrame.left = (LONG)floor( (float)header->rclBounds.left *
6408             header->szlMillimeters.cx * 100 / header->szlDevice.cx );
6409         header->rclFrame.top = (LONG)floor( (float)header->rclBounds.top *
6410             header->szlMillimeters.cy * 100 / header->szlDevice.cy );
6411         header->rclFrame.right = (LONG)ceil( (float)header->rclBounds.right *
6412             header->szlMillimeters.cx * 100 / header->szlDevice.cx );
6413         header->rclFrame.bottom = (LONG)ceil( (float)header->rclBounds.bottom *
6414             header->szlMillimeters.cy * 100 / header->szlDevice.cy );
6415     }
6416
6417     // Some default graphics state (are they really, though?)
6418
6419     SIZEL default_resolution = { RESOLUTION, RESOLUTION };
6420     resolution = default_resolution;
6421     SIZEL default_viewport_ext = { 1, 1 };
6422     viewport_ext = default_viewport_ext;
6423     POINT default_viewport_org = { 0, 0 };
6424     viewport_org = default_viewport_org;
6425     SIZEL default_window_ext = { 1, 1 };
6426     window_ext = default_window_ext;
6427     POINT default_window_org = { 0, 0 };
6428     window_org = default_window_org;
6429
6430     min_device_point = viewport_org;
6431     max_device_point = viewport_org;
6432
6433     pen = (PEN*)globalObjects.find( BLACK_PEN | ENHMETA_STOCK_OBJECT );
6434     brush = (BRUSH*)globalObjects.find( BLACK_BRUSH | ENHMETA_STOCK_OBJECT );
6435     font = (FONT*)globalObjects.find( DEVICE_DEFAULT_FONT | ENHMETA_STOCK_OBJECT );
6436     palette = (PALETTE*)globalObjects.find( DEFAULT_PALETTE | ENHMETA_STOCK_OBJECT );
6437
6438     text_alignment = TA_BASELINE;
6439     text_color = RGB(0,0,0);
6440     bk_color = RGB(0xff,0xff,0xff);
6441     bk_mode = OPAQUE;
6442     polyfill_mode = ALTERNATE;
6443     map_mode = MM_TEXT;
6444     miter_limit = 10.f;
6445
6446     handle = globalObjects.add( this );
6447 }
6448
6449 public:
6450     ::FILE* fp;
6451     DATASTREAM ds;
6452     ENHMETAHEADER* header;
6453     std::vector< EMF::METARECORD* > records;
6454
6455     // Keep a small set of graphics state information
6456     SIZEL resolution;
6457     SIZEL viewport_ext;
6458     POINT viewport_org;

```

```

6472     SIZEL window_ext;
6473     POINT window_org;
6474     bool update_frame;
6475     POINT min_device_point;
6476     POINT max_device_point;
6477     POINT point;
6478     PEN* pen;
6479     BRUSH* brush;
6480     FONT* font;
6481     PALETTE* palette;
6482     UINT text_alignment;
6483     COLORREF text_color;
6484     COLORREF bk_color;
6485     INT bk_mode;
6486     INT polyfill_mode;
6487     INT map_mode;
6488     FLOAT miter_limit;
6489
6495     std::vector< bool > handles;
6496
6502     std::map< HGDIOBJ, HGDIOBJ > emf_handles;
6503
6514     METAFILEDEVICECONTEXT ( FILE* fp_, const RECT* size,
6515                             LPCWSTR description_w )
6516 :   fp(fp_), ds( fp_ )
6517 {
6518     init( size, description_w );
6519 }
6524 virtual ~METAFILEDEVICECONTEXT ( )
6525 {
6526     // Purge all the metarecords (if there are any) {this include the
6527     // header record, too}
6528     if ( records.size() > 0 )
6529 deleteMetafile();
6530 }
6534 OBJECTTYPE GetType ( void )const { return O_METAFILEDEVICECONTEXT; }
6539 DWORD nextHandle ( void )
6540 {
6541     for ( unsigned int i = 1; i < handles.size(); i++ ) {
6542 if ( !handles[i] ) {
6543     handles[i] = true;
6544     return i;
6545 }
6546     }
6547     handles.push_back( true );
6548     // Well, it appears that even StockObject handles count for something.
6549     // Not sure what the right value here is, then.
6550     header->nHandles = handles.size();
6551     return handles.size()-1;
6552 }
6556 void clearHandle ( DWORD handle )
6557 {
6558     if ( handle < handles.size() ) {
6559         handles[handle] = false;
6560     }
6561 }
6567 void appendRecord ( METARECORD* record )
6568 {
6569     records.push_back( record );
6570
6571     header->nBytes += record->size();
6572     header->nRecords++;
6573 }
6579 void appendHandle ( METARECORD* record )
6580 {
6581     records.push_back( record );
6582
6583     header->nBytes += record->size();
6584     header->nRecords++;
6585 }
6590 void deleteMetafile ( void )
6591 {
6592     for ( auto r = records.begin(); r != records.end(); r++ ) {
6593         delete *r;
6594     }
6595     records.clear();
6596 }
6601 void mergePoint ( const LONG& x, const LONG& y )
6602 {
6603     POINT p;
6604     p.x = x;
6605     p.y = y;
6606     mergePoint( p );
6607 }
6612 void mergePoint( const POINT& p )
6613 {
6614     POINT device_point;

```



```

6615
6616 // *** Note, it's possible for the global transformation matrix to
6617 // affect this too. ***
6618
6619 int window_width = window_ext.cx <= 0 ? 1 : window_ext.cx;
6620 int window_height = window_ext.cy <= 0 ? 1 : window_ext.cy;
6621
6622 device_point.x = (LONG)( (float)( p.x - window_org.x ) / window_width *
6623 viewport_ext.cx + viewport_org.x );
6624
6625 device_point.y = (LONG)( (float)( p.y - window_org.y ) / window_height *
6626 viewport_ext.cy + viewport_org.y );
6627
6628 // If the user didn't specify a bounding rectangle in the constructor,
6629 // compute one from this data, too.
6630 if ( device_point.x < min_device_point.x ) {
6631 min_device_point.x = device_point.x;
6632 if ( update_frame ) {
6633 header->rclBounds.left = min_device_point.x - 10;
6634 int device_width = header->szlDevice.cx <= 0 ? 1 : header->szlDevice.cx;
6635 header->rclFrame.left = (LONG)floor( (float)header->rclBounds.left *
6636 header->szlMillimeters.cx * 100 / device_width );
6637 }
6638 }
6639 else if ( device_point.x > max_device_point.x ) {
6640 max_device_point.x = device_point.x;
6641 if ( update_frame ) {
6642 header->rclBounds.right = max_device_point.x + 10;
6643 int device_width = header->szlDevice.cx <= 0 ? 1 : header->szlDevice.cx;
6644 header->rclFrame.right = (LONG)ceil( (float)header->rclBounds.right *
6645 header->szlMillimeters.cx * 100 / device_width );
6646 }
6647 }
6648
6649 if ( device_point.y < min_device_point.y ) {
6650 min_device_point.y = device_point.y;
6651 if ( update_frame ) {
6652 header->rclBounds.top = min_device_point.y - 10;
6653 int device_height = header->szlDevice.cy <= 0 ? 1 : header->szlDevice.cy;
6654 header->rclFrame.top = (LONG)floor( (float)header->rclBounds.top *
6655 header->szlMillimeters.cy * 100 / device_height );
6656 }
6657 }
6658 else if ( device_point.y > max_device_point.y ) {
6659 max_device_point.y = device_point.y;
6660 if ( update_frame ) {
6661 header->rclBounds.bottom = max_device_point.y + 10;
6662 int device_height = header->szlDevice.cy <= 0 ? 1 : header->szlDevice.cy;
6663 header->rclFrame.bottom = (LONG)ceil( (float)header->rclBounds.bottom *
6664 header->szlMillimeters.cy * 100 / device_height );
6665 }
6666 }
6667 }
6668 };
6669
6670 } // close EMF namespace
6671
6672 #undef EMF_UNUSED
6673 #endif /* _LIBEMF_H */

```


Index

- ~EMREXTTEXTOUTA
 - EMF::EMREXTTEXTOUTA, [57](#)
- ~EMREXTTEXTOUTW
 - EMF::EMREXTTEXTOUTW, [59](#)
- ~EMRPOLYBEZIER
 - EMF::EMRPOLYBEZIER, [69](#)
- ~EMRPOLYBEZIER16
 - EMF::EMRPOLYBEZIER16, [72](#)
- ~EMRPOLYBEZIERTO
 - EMF::EMRPOLYBEZIERTO, [74](#)
- ~EMRPOLYBEZIERTO16
 - EMF::EMRPOLYBEZIERTO16, [77](#)
- ~EMRPOLYGON
 - EMF::EMRPOLYGON, [79](#)
- ~EMRPOLYGON16
 - EMF::EMRPOLYGON16, [82](#)
- ~EMRPOLYLINE
 - EMF::EMRPOLYLINE, [84](#)
- ~EMRPOLYLINE16
 - EMF::EMRPOLYLINE16, [87](#)
- ~EMRPOLYLINETO
 - EMF::EMRPOLYLINETO, [89](#)
- ~EMRPOLYLINETO16
 - EMF::EMRPOLYLINETO16, [92](#)
- ~EMRPOLYPOLYGON
 - EMF::EMRPOLYPOLYGON, [94](#)
- ~EMRPOLYPOLYGON16
 - EMF::EMRPOLYPOLYGON16, [97](#)
- ~ENHMETAHEADER
 - EMF::ENHMETAHEADER, [144](#)
- ~METAFILEDEVICECONTEXT
 - EMF::METAFILEDEVICECONTEXT, [158](#)
- ~METARECORD
 - EMF::METARECORD, [162](#)
- add
 - EMF::GLOBALOBJECTS, [151](#)
- appendHandle
 - EMF::METAFILEDEVICECONTEXT, [158](#)
- appendRecord
 - EMF::METAFILEDEVICECONTEXT, [158](#)
- basetsd.h, [172](#)
- begin
 - EMF::GLOBALOBJECTS, [151](#)
- BRUSH
 - EMF::BRUSH, [8](#)
- BYTEARRAY
 - EMF::BYTEARRAY, [9](#)
- CHARSTR
 - EMF::CHARSTR, [10](#)
- clearHandle
 - EMF::METAFILEDEVICECONTEXT, [158](#)
- contexts
 - EMF::GRAPHICSOBJECT, [155](#)
- DATASTREAM
 - EMF::DATASTREAM, [12](#)
- deleteMetafile
 - EMF::METAFILEDEVICECONTEXT, [159](#)
- ds
 - EMF::METAFILEDEVICECONTEXT, [160](#)
- DWORDARRAY
 - EMF::DWORDARRAY, [28](#)
- emf.h, [172](#)
- EMF::BRUSH, [7](#)
 - BRUSH, [8](#)
 - getType, [8](#)
 - newEMR, [8](#)
- EMF::BYTEARRAY, [9](#)
 - BYTEARRAY, [9](#)
- EMF::CHARSTR, [10](#)
 - CHARSTR, [10](#)
- EMF::DATASTREAM, [11](#)
 - DATASTREAM, [12](#)
 - operator<<, [12–20](#)
 - operator>>, [20–27](#)
 - setStream, [27](#)
- EMF::DWORDARRAY, [28](#)
 - DWORDARRAY, [28](#)
- EMF::EMRARC, [29](#)
 - EMRARC, [29, 30](#)
 - execute, [30](#)
 - serialize, [30](#)
 - size, [31](#)
- EMF::EMRARCTO, [31](#)
 - EMRARCTO, [32](#)
 - execute, [32](#)
 - serialize, [33](#)
 - size, [33](#)
- EMF::EMRBEGINPATH, [33](#)
 - EMRBEGINPATH, [34](#)
 - execute, [34](#)
 - serialize, [35](#)
 - size, [35](#)
- EMF::EMRCLOSEFIGURE, [35](#)
 - EMRCLOSEFIGURE, [36](#)
 - execute, [36](#)
 - serialize, [36](#)
 - size, [37](#)
- EMF::EMRCREATEBRUSHINDIRECT, [37](#)
 - EMRCREATEBRUSHINDIRECT, [37, 38](#)
 - execute, [38](#)
 - serialize, [38](#)
 - size, [39](#)
- EMF::EMRCREATEPALETTE, [39](#)
 - EMRCREATEPALETTE, [39, 40](#)
 - execute, [40](#)
 - serialize, [40](#)
 - size, [41](#)

- EMF::EMRCREATEPEN, 41
 - EMRCREATEPEN, 41, 42
 - execute, 42
 - serialize, 42
 - size, 43
- EMF::EMRDELETEOBJECT, 43
 - EMRDELETEOBJECT, 43, 44
 - execute, 44
 - serialize, 44
 - size, 45
- EMF::EMRELLIPSE, 45
 - EMRELLIPSE, 45, 46
 - execute, 46
 - serialize, 46
 - size, 47
- EMF::EMRENDPATH, 47
 - EMRENDPATH, 47, 48
 - execute, 49
 - serialize, 49
 - size, 49
- EMF::EMREOF, 50
 - EMREOF, 50
 - execute, 51
 - serialize, 51
 - size, 51
- EMF::EMREXTCREATEFONTINDIRECTW, 52
 - EMREXTCREATEFONTINDIRECTW, 52
 - execute, 53
 - serialize, 53
 - size, 53
- EMF::EMREXTCREATEPEN, 54
 - EMREXTCREATEPEN, 54, 55
 - execute, 55
 - serialize, 55
 - size, 56
- EMF::EMREXTTEXTOUTA, 56
 - ~EMREXTTEXTOUTA, 57
 - EMREXTTEXTOUTA, 56, 57
 - execute, 57
 - serialize, 58
 - size, 58
- EMF::EMREXTTEXTOUTW, 58
 - ~EMREXTTEXTOUTW, 59
 - EMREXTTEXTOUTW, 59
 - execute, 60
 - serialize, 60
 - size, 60
- EMF::EMRFILLPATH, 60
 - EMRFILLPATH, 61
 - execute, 61
 - serialize, 62
 - size, 62
- EMF::EMRLINETO, 62
 - EMRLINETO, 63
 - execute, 63
 - serialize, 64
 - size, 64
- EMF::EMRMODIFYWORLDTRANSFORM, 64
 - EMRMODIFYWORLDTRANSFORM, 65
 - execute, 66
 - serialize, 66
 - size, 66
- EMF::EMRMOVETOEX, 66
 - EMRMOVETOEX, 67
 - execute, 67
 - serialize, 68
 - size, 68
- EMF::EMRPOLYBEZIER, 68
 - ~EMRPOLYBEZIER, 69
 - EMRPOLYBEZIER, 69
 - execute, 70
 - serialize, 70
 - size, 70
- EMF::EMRPOLYBEZIER16, 71
 - ~EMRPOLYBEZIER16, 72
 - EMRPOLYBEZIER16, 71, 72
 - execute, 72
 - serialize, 73
 - size, 73
- EMF::EMRPOLYBEZIERTO, 73
 - ~EMRPOLYBEZIERTO, 74
 - EMRPOLYBEZIERTO, 74
 - execute, 75
 - serialize, 75
 - size, 75
- EMF::EMRPOLYBEZIERTO16, 76
 - ~EMRPOLYBEZIERTO16, 77
 - EMRPOLYBEZIERTO16, 76, 77
 - execute, 77
 - serialize, 78
 - size, 78
- EMF::EMRPOLYGON, 78
 - ~EMRPOLYGON, 79
 - EMRPOLYGON, 79
 - execute, 79
 - serialize, 80
 - size, 80
- EMF::EMRPOLYGON16, 80
 - ~EMRPOLYGON16, 82
 - EMRPOLYGON16, 81, 82
 - execute, 82
 - serialize, 82
 - size, 82
- EMF::EMRPOLYLINE, 83
 - ~EMRPOLYLINE, 84
 - EMRPOLYLINE, 83, 84
 - execute, 84
 - serialize, 84
 - size, 86
- EMF::EMRPOLYLINE16, 86
 - ~EMRPOLYLINE16, 87
 - EMRPOLYLINE16, 86, 87
 - execute, 87
 - serialize, 88
 - size, 88
- EMF::EMRPOLYLINETO, 88

- ~EMRPOLYLINETO, 89
- EMRPOLYLINETO, 89
- execute, 90
- serialize, 90
- size, 90
- EMF::EMRPOLYLINETO16, 91
 - ~EMRPOLYLINETO16, 92
 - EMRPOLYLINETO16, 91, 92
 - execute, 92
 - serialize, 93
 - size, 93
- EMF::EMRPOLYPOLYGON, 93
 - ~EMRPOLYPOLYGON, 94
 - EMRPOLYPOLYGON, 94
 - execute, 94
 - serialize, 95
 - size, 95
- EMF::EMRPOLYPOLYGON16, 95
 - ~EMRPOLYPOLYGON16, 97
 - EMRPOLYPOLYGON16, 96, 97
 - execute, 97
 - serialize, 97
 - size, 98
- EMF::EMRRECTANGLE, 98
 - EMRRECTANGLE, 98, 99
 - execute, 99
 - serialize, 99
 - size, 100
- EMF::EMRRESTOREDC, 100
 - EMRRESTOREDC, 100, 101
 - execute, 102
 - serialize, 102
 - size, 102
- EMF::EMRSAVEDC, 103
 - EMRSAVEDC, 103
 - execute, 104
 - serialize, 104
 - size, 104
- EMF::EMRSCALEVIEWPORTEXT, 105
 - EMRSCALEVIEWPORTEXT, 105, 106
 - execute, 106
 - serialize, 106
 - size, 106
- EMF::EMRSCALEWINDOWEXT, 107
 - EMRSCALEWINDOWEXT, 107, 108
 - execute, 108
 - serialize, 108
 - size, 108
- EMF::EMRSELECTOBJECT, 109
 - EMRSELECTOBJECT, 109
 - execute, 110
 - serialize, 110
 - size, 110
- EMF::EMRSETBKCOLOR, 111
 - EMRSETBKCOLOR, 111
 - execute, 112
 - serialize, 112
 - size, 112
- EMF::EMRSETBKMODE, 113
 - EMRSETBKMODE, 113
 - execute, 114
 - serialize, 114
 - size, 114
- EMF::EMRSETMAPMODE, 115
 - EMRSETMAPMODE, 115
 - execute, 116
 - serialize, 116
 - size, 116
- EMF::EMRSETMETARGN, 117
 - EMRSETMETARGN, 117
 - execute, 118
 - serialize, 118
 - size, 118
- EMF::EMRSETMITERLIMIT, 119
 - EMRSETMITERLIMIT, 119
 - execute, 120
 - serialize, 120
 - size, 120
- EMF::EMRSETPIXELV, 121
 - EMRSETPIXELV, 121
 - execute, 122
 - serialize, 122
 - size, 122
- EMF::EMRSETPOLYFILLMODE, 123
 - EMRSETPOLYFILLMODE, 123
 - execute, 124
 - serialize, 124
 - size, 124
- EMF::EMRSETTEXTALIGN, 125
 - EMRSETTEXTALIGN, 125
 - execute, 126
 - serialize, 126
 - size, 126
- EMF::EMRSETTEXTCOLOR, 127
 - EMRSETTEXTCOLOR, 127
 - execute, 128
 - serialize, 128
 - size, 128
- EMF::EMRSETVIEWPORTEXT, 129
 - EMRSETVIEWPORTEXT, 129
 - execute, 130
 - serialize, 130
 - size, 130
- EMF::EMRSETVIEWPORTORGEX, 131
 - EMRSETVIEWPORTORGEX, 131
 - execute, 132
 - serialize, 132
 - size, 132
- EMF::EMRSETWINDOWEXT, 133
 - EMRSETWINDOWEXT, 133
 - execute, 134
 - serialize, 134
 - size, 134
- EMF::EMRSETWINDOWORGEX, 135
 - EMRSETWINDOWORGEX, 135
 - execute, 136

- serialize, 136
- size, 136
- EMF::EMRSETWORLDTRANSFORM, 137
 - EMRSETWORLDTRANSFORM, 137
 - execute, 139
 - serialize, 139
 - size, 139
- EMF::EMRSTROKEANDFILLPATH, 140
 - EMRSTROKEANDFILLPATH, 140
 - execute, 141
 - serialize, 141
 - size, 141
- EMF::EMRSTROKEPATH, 142
 - EMRSTROKEPATH, 142
 - execute, 143
 - serialize, 143
 - size, 143
- EMF::ENHMETAHEADER, 144
 - ~ENHMETAHEADER, 144
 - ENHMETAHEADER, 144
 - execute, 145
 - serialize, 145
 - size, 145
 - unserialize, 145
- EMF::EXTPEN, 146
 - EXTPEN, 146
 - getType, 146
 - newEMR, 147
- EMF::FONT, 147
 - FONT, 148
 - getType, 148
 - newEMR, 148
- EMF::GLOBALOBJECTS, 149
 - add, 151
 - begin, 151
 - end, 152
 - find, 152
 - newRecord, 152
 - remove, 152
- EMF::GRAPHICSOBJECT, 154
 - contexts, 155
 - newEMR, 154
- EMF::INTARRAY, 155
 - INTARRAY, 155
- EMF::METAFILEDEVICECONTEXT, 156
 - ~METAFILEDEVICECONTEXT, 158
 - appendHandle, 158
 - appendRecord, 158
 - clearHandle, 158
 - deleteMetafile, 159
 - ds, 160
 - emf_handles, 160
 - fp, 160
 - getType, 159
 - handles, 160
 - header, 160
 - mergePoint, 159
 - METAFILEDEVICECONTEXT, 157
 - nextHandle, 159
 - records, 160
- EMF::METARECORD, 161
 - ~METARECORD, 162
 - execute, 162
 - serialize, 162
 - size, 163
- EMF::OBJECT, 164
 - getType, 164
 - handle, 165
 - OBJECT, 164
- EMF::PADDING, 165
 - PADDING, 165
- EMF::PALETTE, 166
 - getType, 167
 - newEMR, 167
 - PALETTE, 166
- EMF::PEN, 167
 - getType, 168
 - newEMR, 168
 - PEN, 168
- EMF::POINT16ARRAY, 169
 - POINT16ARRAY, 169
- EMF::POINTLARRAY, 170
 - POINTLARRAY, 170
- EMF::WCHARSTR, 171
 - WCHARSTR, 171
- emf_handles
 - EMF::METAFILEDEVICECONTEXT, 160
- EMRARC
 - EMF::EMRARC, 29, 30
- EMRARCTO
 - EMF::EMRARCTO, 32
- EMRBEGINPATH
 - EMF::EMRBEGINPATH, 34
- EMRCLOSEFIGURE
 - EMF::EMRCLOSEFIGURE, 36
- EMRCREATEBRUSHINDIRECT
 - EMF::EMRCREATEBRUSHINDIRECT, 37, 38
- EMRCREATEPALETTE
 - EMF::EMRCREATEPALETTE, 39, 40
- EMRCREATEPEN
 - EMF::EMRCREATEPEN, 41, 42
- EMRDELETEOBJECT
 - EMF::EMRDELETEOBJECT, 43, 44
- EMRELLIPSE
 - EMF::EMRELLIPSE, 45, 46
- EMRENDPATH
 - EMF::EMRENDPATH, 47, 48
- EMREOF
 - EMF::EMREOF, 50
- EMREXTCREATEFONTINDIRECTW
 - EMF::EMREXTCREATEFONTINDIRECTW, 52
- EMREXTCREATEPEN
 - EMF::EMREXTCREATEPEN, 54, 55
- EMREXTTEXTOUTA
 - EMF::EMREXTTEXTOUTA, 56, 57
- EMREXTTEXTOUTW

EMF::EMREXTTEXTOUTW, 59
EMRFILLPATH
EMF::EMRFILLPATH, 61
EMRLINETO
EMF::EMRLINETO, 63
EMRMODIFYWORLDTRANSFORM
EMF::EMRMODIFYWORLDTRANSFORM, 65
EMRMOVETOEX
EMF::EMRMOVETOEX, 67
EMRPOLYBEZIER
EMF::EMRPOLYBEZIER, 69
EMRPOLYBEZIER16
EMF::EMRPOLYBEZIER16, 71, 72
EMRPOLYBEZIERTO
EMF::EMRPOLYBEZIERTO, 74
EMRPOLYBEZIERTO16
EMF::EMRPOLYBEZIERTO16, 76, 77
EMRPOLYGON
EMF::EMRPOLYGON, 79
EMRPOLYGON16
EMF::EMRPOLYGON16, 81, 82
EMRPOLYLINE
EMF::EMRPOLYLINE, 83, 84
EMRPOLYLINE16
EMF::EMRPOLYLINE16, 86, 87
EMRPOLYLINETO
EMF::EMRPOLYLINETO, 89
EMRPOLYLINETO16
EMF::EMRPOLYLINETO16, 91, 92
EMRPOLYPOLYGON
EMF::EMRPOLYPOLYGON, 94
EMRPOLYPOLYGON16
EMF::EMRPOLYPOLYGON16, 96, 97
EMRRECTANGLE
EMF::EMRRECTANGLE, 98, 99
EMRRESTOREDC
EMF::EMRRESTOREDC, 100, 101
EMRSAVEDC
EMF::EMRSAVEDC, 103
EMRSCALEVIEWPORTEXT
EMF::EMRSCALEVIEWPORTEXT, 105, 106
EMRSCALEWINDOWEXT
EMF::EMRSCALEWINDOWEXT, 107, 108
EMRSELECTOBJECT
EMF::EMRSELECTOBJECT, 109
EMRSETBKCOLOR
EMF::EMRSETBKCOLOR, 111
EMRSETBKMODE
EMF::EMRSETBKMODE, 113
EMRSETMAPMODE
EMF::EMRSETMAPMODE, 115
EMRSETMETARGN
EMF::EMRSETMETARGN, 117
EMRSETMITERLIMIT
EMF::EMRSETMITERLIMIT, 119
EMRSETPIXELV
EMF::EMRSETPIXELV, 121
EMRSETPOLYFILLMODE
EMF::EMRSETPOLYFILLMODE, 123
EMRSETTEXTALIGN
EMF::EMRSETTEXTALIGN, 125
EMRSETTEXTCOLOR
EMF::EMRSETTEXTCOLOR, 127
EMRSETVIEWPORTEXT
EMF::EMRSETVIEWPORTEXT, 129
EMRSETVIEWPORTORGE
EMF::EMRSETVIEWPORTORGE, 131
EMRSETWINDOWEXT
EMF::EMRSETWINDOWEXT, 133
EMRSETWINDOWORGE
EMF::EMRSETWINDOWORGE, 135
EMRSETWORLDTRANSFORM
EMF::EMRSETWORLDTRANSFORM, 137
EMRSTROKEANDFILLPATH
EMF::EMRSTROKEANDFILLPATH, 140
EMRSTROKEPATH
EMF::EMRSTROKEPATH, 142
end
EMF::GLOBALOBJECTS, 152
ENHMETAHEADER
EMF::ENHMETAHEADER, 144
execute
EMF::EMRARC, 30
EMF::EMRARCTO, 32
EMF::EMRBEGINPATH, 34
EMF::EMRCLOSEFIGURE, 36
EMF::EMRCREATEBRUSHINDIRECT, 38
EMF::EMRCREATEPALETTE, 40
EMF::EMRCREATEPEN, 42
EMF::EMRDELETEOBJECT, 44
EMF::EMRELLIPSE, 46
EMF::EMRENDPATH, 49
EMF::EMREOF, 51
EMF::EMREXTCREATEFONTINDIRECTW, 53
EMF::EMREXTCREATEPEN, 55
EMF::EMREXTTEXTOUTA, 57
EMF::EMREXTTEXTOUTW, 60
EMF::EMRFILLPATH, 61
EMF::EMRLINETO, 63
EMF::EMRMODIFYWORLDTRANSFORM, 66
EMF::EMRMOVETOEX, 67
EMF::EMRPOLYBEZIER, 70
EMF::EMRPOLYBEZIER16, 72
EMF::EMRPOLYBEZIERTO, 75
EMF::EMRPOLYBEZIERTO16, 77
EMF::EMRPOLYGON, 79
EMF::EMRPOLYGON16, 82
EMF::EMRPOLYLINE, 84
EMF::EMRPOLYLINE16, 87
EMF::EMRPOLYLINETO, 90
EMF::EMRPOLYLINETO16, 92
EMF::EMRPOLYPOLYGON, 94
EMF::EMRPOLYPOLYGON16, 97
EMF::EMRRECTANGLE, 99
EMF::EMRRESTOREDC, 102
EMF::EMRSAVEDC, 104

- EMF::EMRSCALEVIEWPORTEXTEX, 106
- EMF::EMRSCALEWINDOWEXTTEX, 108
- EMF::EMRSELECTOBJECT, 110
- EMF::EMRSETBKCOLOR, 112
- EMF::EMRSETBKMODE, 114
- EMF::EMRSETMAPMODE, 116
- EMF::EMRSETMETARGN, 118
- EMF::EMRSETMITERLIMIT, 120
- EMF::EMRSETPIXELV, 122
- EMF::EMRSETPOLYFILLMODE, 124
- EMF::EMRSETTEXTALIGN, 126
- EMF::EMRSETTEXTCOLOR, 128
- EMF::EMRSETVIEWPORTEXTEX, 130
- EMF::EMRSETVIEWPORTORGEX, 132
- EMF::EMRSETWINDOWEXTTEX, 134
- EMF::EMRSETWINDOWORGEX, 136
- EMF::EMRSETWORLDTRANSFORM, 139
- EMF::EMRSTROKEANDFILLPATH, 141
- EMF::EMRSTROKEPATH, 143
- EMF::ENHMETAHEADER, 145
- EMF::METARECORD, 162
- EXTPEN
 - EMF::EXTPEN, 146
- find
 - EMF::GLOBALOBJECTS, 152
- FONT
 - EMF::FONT, 148
- fp
 - EMF::METAFILEDEVICECONTEXT, 160
- getType
 - EMF::BRUSH, 8
 - EMF::EXTPEN, 146
 - EMF::FONT, 148
 - EMF::METAFILEDEVICECONTEXT, 159
 - EMF::OBJECT, 164
 - EMF::PALETTE, 167
 - EMF::PEN, 168
- guiddef.h, 174
- handle
 - EMF::OBJECT, 165
- handles
 - EMF::METAFILEDEVICECONTEXT, 160
- header
 - EMF::METAFILEDEVICECONTEXT, 160
- INTARRAY
 - EMF::INTARRAY, 155
- libemf.h, 364
- mergePoint
 - EMF::METAFILEDEVICECONTEXT, 159
- METAFILEDEVICECONTEXT
 - EMF::METAFILEDEVICECONTEXT, 157
- newEMR
 - EMF::BRUSH, 8
- EMF::EXTPEN, 147
- EMF::FONT, 148
- EMF::GRAPHICSOBJECT, 154
- EMF::PALETTE, 167
- EMF::PEN, 168
- newRecord
 - EMF::GLOBALOBJECTS, 152
- nextHandle
 - EMF::METAFILEDEVICECONTEXT, 159
- OBJECT
 - EMF::OBJECT, 164
- operator<<
 - EMF::DATASTREAM, 12–20
- operator>>
 - EMF::DATASTREAM, 20–27
- PADDING
 - EMF::PADDING, 165
- PALETTE
 - EMF::PALETTE, 166
- PEN
 - EMF::PEN, 168
- POINT16ARRAY
 - EMF::POINT16ARRAY, 169
- POINTLARRAY
 - EMF::POINTLARRAY, 170
- poppack.h, 175
- pshpack2.h, 176
- pshpack4.h, 176
- records
 - EMF::METAFILEDEVICECONTEXT, 160
- remove
 - EMF::GLOBALOBJECTS, 152
- serialize
 - EMF::EMRARC, 30
 - EMF::EMRARCTO, 33
 - EMF::EMRBEGINPATH, 35
 - EMF::EMRCLOSEFIGURE, 36
 - EMF::EMRCREATEBRUSHINDIRECT, 38
 - EMF::EMRCREATEPALETTE, 40
 - EMF::EMRCREATEPEN, 42
 - EMF::EMRDELETEOBJECT, 44
 - EMF::EMRELLIPSE, 46
 - EMF::EMRENDPATH, 49
 - EMF::EMREOF, 51
 - EMF::EMREXTCREATEFONTINDIRECTW, 53
 - EMF::EMREXTCREATEPEN, 55
 - EMF::EMREXTTEXTOUTA, 58
 - EMF::EMREXTTEXTOUTW, 60
 - EMF::EMRFILLPATH, 62
 - EMF::EMRLINETO, 64
 - EMF::EMRMODIFYWORLDTRANSFORM, 66
 - EMF::EMRMOVETOEX, 68
 - EMF::EMRPOLYBEZIER, 70
 - EMF::EMRPOLYBEZIER16, 73
 - EMF::EMRPOLYBEZIERTO, 75

- EMF::EMRPOLYBEZIERTO16, [78](#)
- EMF::EMRPOLYGON, [80](#)
- EMF::EMRPOLYGON16, [82](#)
- EMF::EMRPOLYLINE, [84](#)
- EMF::EMRPOLYLINE16, [88](#)
- EMF::EMRPOLYLINETO, [90](#)
- EMF::EMRPOLYLINETO16, [93](#)
- EMF::EMRPOLYPOLYGON, [95](#)
- EMF::EMRPOLYPOLYGON16, [97](#)
- EMF::EMRRECTANGLE, [99](#)
- EMF::EMRRESTOREDC, [102](#)
- EMF::EMRSAVEDC, [104](#)
- EMF::EMRSCALEVIEWPORTEXTEX, [106](#)
- EMF::EMRSCALEWINDOWEXTTEX, [108](#)
- EMF::EMRSELECTOBJECT, [110](#)
- EMF::EMRSETBKCOLOR, [112](#)
- EMF::EMRSETBKMODE, [114](#)
- EMF::EMRSETMAPMODE, [116](#)
- EMF::EMRSETMETARGN, [118](#)
- EMF::EMRSETMITERLIMIT, [120](#)
- EMF::EMRSETPIXELV, [122](#)
- EMF::EMRSETPOLYFILLMODE, [124](#)
- EMF::EMRSETTEXTALIGN, [126](#)
- EMF::EMRSETTEXTCOLOR, [128](#)
- EMF::EMRSETVIEWPORTEXTEX, [130](#)
- EMF::EMRSETVIEWPORTORGEX, [132](#)
- EMF::EMRSETWINDOWEXTTEX, [134](#)
- EMF::EMRSETWINDOWORGEX, [136](#)
- EMF::EMRSETWORLDTRANSFORM, [139](#)
- EMF::EMRSTROKEANDFILLPATH, [141](#)
- EMF::EMRSTROKEPATH, [143](#)
- EMF::ENHMETAHEADER, [145](#)
- EMF::METARECORD, [162](#)
- setStream
 - EMF::DATASTREAM, [27](#)
- size
 - EMF::EMRARC, [31](#)
 - EMF::EMRARCTO, [33](#)
 - EMF::EMRBEGINPATH, [35](#)
 - EMF::EMRCLOSEFIGURE, [37](#)
 - EMF::EMRCREATEBRUSHINDIRECT, [39](#)
 - EMF::EMRCREATEPALETTE, [41](#)
 - EMF::EMRCREATEPEN, [43](#)
 - EMF::EMRDELETEOBJECT, [45](#)
 - EMF::EMRELLIPSE, [47](#)
 - EMF::EMRENDPATH, [49](#)
 - EMF::EMREOF, [51](#)
 - EMF::EMREXTCREATEFONTINDIRECTW, [53](#)
 - EMF::EMREXTCREATEPEN, [56](#)
 - EMF::EMREXTTEXTOUTA, [58](#)
 - EMF::EMREXTTEXTOUTW, [60](#)
 - EMF::EMRFILLPATH, [62](#)
 - EMF::EMRLINETO, [64](#)
 - EMF::EMRMODIFYWORLDTRANSFORM, [66](#)
 - EMF::EMRMOVETOEX, [68](#)
 - EMF::EMRPOLYBEZIER, [70](#)
 - EMF::EMRPOLYBEZIER16, [73](#)
 - EMF::EMRPOLYBEZIERTO, [75](#)
 - EMF::EMRPOLYBEZIERTO16, [78](#)
 - EMF::EMRPOLYGON, [80](#)
 - EMF::EMRPOLYGON16, [82](#)
 - EMF::EMRPOLYLINE, [86](#)
 - EMF::EMRPOLYLINE16, [88](#)
 - EMF::EMRPOLYLINETO, [90](#)
 - EMF::EMRPOLYLINETO16, [93](#)
 - EMF::EMRPOLYPOLYGON, [95](#)
 - EMF::EMRPOLYPOLYGON16, [98](#)
 - EMF::EMRRECTANGLE, [100](#)
 - EMF::EMRRESTOREDC, [102](#)
 - EMF::EMRSAVEDC, [104](#)
 - EMF::EMRSCALEVIEWPORTEXTEX, [106](#)
 - EMF::EMRSCALEWINDOWEXTTEX, [108](#)
 - EMF::EMRSELECTOBJECT, [110](#)
 - EMF::EMRSETBKCOLOR, [112](#)
 - EMF::EMRSETBKMODE, [114](#)
 - EMF::EMRSETMAPMODE, [116](#)
 - EMF::EMRSETMETARGN, [118](#)
 - EMF::EMRSETMITERLIMIT, [120](#)
 - EMF::EMRSETPIXELV, [122](#)
 - EMF::EMRSETPOLYFILLMODE, [124](#)
 - EMF::EMRSETTEXTALIGN, [126](#)
 - EMF::EMRSETTEXTCOLOR, [128](#)
 - EMF::EMRSETVIEWPORTEXTEX, [130](#)
 - EMF::EMRSETVIEWPORTORGEX, [132](#)
 - EMF::EMRSETWINDOWEXTTEX, [134](#)
 - EMF::EMRSETWINDOWORGEX, [136](#)
 - EMF::EMRSETWORLDTRANSFORM, [139](#)
 - EMF::EMRSTROKEANDFILLPATH, [141](#)
 - EMF::EMRSTROKEPATH, [143](#)
 - EMF::ENHMETAHEADER, [145](#)
 - EMF::METARECORD, [163](#)
- unserialize
 - EMF::ENHMETAHEADER, [145](#)
- w16.h, [176](#)
- WCHARSTR
 - EMF::WCHARSTR, [171](#)
- winbase.h, [177](#)
- windef.h, [198](#)
- winerror.h, [201](#)
- wingdi.h, [223](#)
- winnt.h, [261](#)
- winuser.h, [316](#)