

```
1  typedef struct _IMAGE_DOS_HEADER {          // DOS .EXE header
2      WORD    e_magic;                        // Magic number
3      WORD    e_cblp;                        // Bytes on last page of file
4      WORD    e_cp;                          // Pages in file
5      WORD    e_crlc;                        // Relocations
6      WORD    e_cparhdr;                     // Size of header in paragraphs
7      WORD    e_minalloc;                     // Minimum extra paragraphs needed
8      WORD    e_maxalloc;                     // Maximum extra paragraphs needed
9      WORD    e_ss;                          // Initial (relative) SS value
10     WORD    e_sp;                           // Initial SP value
11     WORD    e_csum;                         // Checksum
12     WORD    e_ip;                           // Initial IP value
13     WORD    e_cs;                           // Initial (relative) CS value
14     WORD    e_lfarlc;                       // File address of relocation table
15     WORD    e_ovno;                         // Overlay number
16     WORD    e_res[4];                       // Reserved words
17     WORD    e_oemid;                        // OEM identifier (for e_oeminfo)
18     WORD    e_oeminfo;                      // OEM information; e_oemid specific
19     WORD    e_res2[10];                     // Reserved words
20     LONG    e_lfanew;                       // File address of new exe header
21 } IMAGE_DOS_HEADER, *PIMAGE_DOS_HEADER;
22 //
23 // File header format.
24 //
25
26 typedef struct _IMAGE_FILE_HEADER {
27     WORD    Machine;
28     WORD    NumberOfSections;
29     DWORD   TimeDateStamp;
30     DWORD   PointerToSymbolTable;
31     DWORD   NumberOfSymbols;
32     WORD    SizeOfOptionalHeader;
33     WORD    Characteristics;
34 } IMAGE_FILE_HEADER, *PIMAGE_FILE_HEADER;
35
```

```
36 #define IMAGE_SIZEOF_FILE_HEADER          20
37 //
38 // Directory format.
39 //
40
41 typedef struct _IMAGE_DATA_DIRECTORY {
42     DWORD   VirtualAddress;
43     DWORD   Size;
44 } IMAGE_DATA_DIRECTORY, *PIMAGE_DATA_DIRECTORY;
45
46 #define IMAGE_NUMBEROF_DIRECTORY_ENTRIES    16
47
48 //
49 // Optional header format.
50 //
51
52 typedef struct _IMAGE_OPTIONAL_HEADER {
53     //
54     // Standard fields.
55     //
56
57     WORD    Magic;
58     BYTE    MajorLinkerVersion;
59     BYTE    MinorLinkerVersion;
60     DWORD   SizeOfCode;
61     DWORD   SizeOfInitializedData;
62     DWORD   SizeOfUninitializedData;
63     DWORD   AddressOfEntryPoint;
64     DWORD   BaseOfCode;
65     DWORD   BaseOfData;
66
67     //
68     // NT additional fields.
69     //
70
```

```
71     DWORD    ImageBase;
72     DWORD    SectionAlignment;
73     DWORD    FileAlignment;
74     WORD     MajorOperatingSystemVersion;
75     WORD     MinorOperatingSystemVersion;
76     WORD     MajorImageVersion;
77     WORD     MinorImageVersion;
78     WORD     MajorSubsystemVersion;
79     WORD     MinorSubsystemVersion;
80     DWORD    Win32VersionValue;
81     DWORD    SizeOfImage;
82     DWORD    SizeOfHeaders;
83     DWORD    CheckSum;
84     WORD     Subsystem;
85     WORD     DllCharacteristics;
86     DWORD    SizeOfStackReserve;
87     DWORD    SizeOfStackCommit;
88     DWORD    SizeOfHeapReserve;
89     DWORD    SizeOfHeapCommit;
90     DWORD    LoaderFlags;
91     DWORD    NumberOfRvaAndSizes;
92     IMAGE_DATA_DIRECTORY DataDirectory[IMAGE_NUMBEROF_DIRECTORY_ENTRIES];
93 } IMAGE_OPTIONAL_HEADER32, *PIMAGE_OPTIONAL_HEADER32;
94
95 #define IMAGE_NT_OPTIONAL_HDR32_MAGIC    0x10b
96 #define IMAGE_NT_OPTIONAL_HDR64_MAGIC    0x20b
97 #define IMAGE_ROM_OPTIONAL_HDR_MAGIC     0x107
98
99 typedef IMAGE_OPTIONAL_HEADER32          IMAGE_OPTIONAL_HEADER;
100 typedef PIMAGE_OPTIONAL_HEADER32         PIMAGE_OPTIONAL_HEADER;
101 #define IMAGE_NT_OPTIONAL_HDR_MAGIC      IMAGE_NT_OPTIONAL_HDR32_MAGIC
102
103 typedef struct _IMAGE_NT_HEADERS {
104     DWORD Signature;
105     IMAGE_FILE_HEADER FileHeader;
```

```
106     IMAGE_OPTIONAL_HEADER32 OptionalHeader;
107 } IMAGE_NT_HEADERS32, *PIMAGE_NT_HEADERS32;
108
109 typedef IMAGE_NT_HEADERS32          IMAGE_NT_HEADERS;
110 typedef PIMAGE_NT_HEADERS32        PIMAGE_NT_HEADERS;
111
112 // Directory Entries
113
114 #define IMAGE_DIRECTORY_ENTRY_EXPORT      0 // Export Directory
115 #define IMAGE_DIRECTORY_ENTRY_IMPORT      1 // Import Directory
116 #define IMAGE_DIRECTORY_ENTRY_RESOURCE    2 // Resource Directory
117 #define IMAGE_DIRECTORY_ENTRY_EXCEPTION   3 // Exception Directory
118 #define IMAGE_DIRECTORY_ENTRY_SECURITY    4 // Security Directory
119 #define IMAGE_DIRECTORY_ENTRY_BASERELOC   5 // Base Relocation Table
120 #define IMAGE_DIRECTORY_ENTRY_DEBUG       6 // Debug Directory
121 //      IMAGE_DIRECTORY_ENTRY_COPYRIGHT    7 // (X86 usage)
122 #define IMAGE_DIRECTORY_ENTRY_ARCHITECTURE 7 // Architecture Specific Data
123 #define IMAGE_DIRECTORY_ENTRY_GLOBALPTR    8 // RVA of GP
124 #define IMAGE_DIRECTORY_ENTRY_TLS          9 // TLS Directory
125 #define IMAGE_DIRECTORY_ENTRY_LOAD_CONFIG 10 // Load Configuration Directory
126 #define IMAGE_DIRECTORY_ENTRY_BOUND_IMPORT 11 // Bound Import Directory in headers
127 #define IMAGE_DIRECTORY_ENTRY_IAT         12 // Import Address Table
128 #define IMAGE_DIRECTORY_ENTRY_DELAY_IMPORT 13 // Delay Load Import Descriptors
129 #define IMAGE_DIRECTORY_ENTRY_COM_DESCRIPTOR 14 // COM Runtime descriptor
130
131 //
132 // Section header format.
133 //
134
135 #define IMAGE_SIZEOF_SHORT_NAME          8
136
137 typedef struct _IMAGE_SECTION_HEADER {
138     BYTE    Name[IMAGE_SIZEOF_SHORT_NAME];
139     union {
140         DWORD    PhysicalAddress;
```

```
141         DWORD    VirtualSize;
142     } Misc;
143     DWORD    VirtualAddress;
144     DWORD    SizeOfRawData;
145     DWORD    PointerToRawData;
146     DWORD    PointerToRelocations;
147     DWORD    PointerToLinenumbers;
148     WORD     NumberOfRelocations;
149     WORD     NumberOfLinenumbers;
150     DWORD    Characteristics;
151 } IMAGE_SECTION_HEADER, *PIMAGE_SECTION_HEADER;
152
153 #define IMAGE_SIZEOF_SECTION_HEADER    40
154
```