

WinHex Template Tutorial

Paul Mullen

Winhex has a powerful but often overlooked way of viewing many different data file formats, the template.

A template describes the structure of each record in a binary data file, so enabling you to see and edit the actual values within each record. You can use a pre-defined template or very simply create your own each time you need to examine a data file.

To illustrate the use of templates, I will look at data in the popular “dbf”, or “xbase”, format, which originated with Ashton-Tate’s dBase program and has since been adopted by many applications. I have provided a sample file called states.dbf containing information about the 50 States of the USA.

Information about many different file formats can be found at <http://www.wotsit.org>, where I obtained the format of a dbf file.

A dbf file contains three portions:

- Header
- Field Descriptions
- Data Records

The Header is 32 bytes long, unused fields at end being zero filled.

BYTES	DESCRIPTION
00	Version (see below)
01-03	Last update, format YYMMDD
04-07	Number of records in file (32-bit number)
08-09	Length of header (16-bit number)
10-11	Number of bytes in record (16-bit number)
12-13	Reserved, fill with 0x00
14	Incomplete transaction flag (dBase IV) Begin Transaction sets it to 0x01 End Transaction or RollBack reset it to 0x00
15	Encryption flag, encrypted 0x01 else 0x00
16-27	dBaseIV multi-user environment use
28	Production index exists - 0x01 else 0x00
	dBaseIV language driver ID
30-31	Reserved, fill with 0x00

Version: usually 03 if there is no memo field, 0x83 with memo
FoxPro with memo - 0xF5
dBaseIV with memo - 0x8B
dBaseIV with SQL Table - 0x8E

To start creating a template, open the data file in Winhex, position the cursor at the start of a record, and then select the Template Manager from the Tools menu (shortcut, use Alt-F12). It is usually easier to start with an existing template and modify it, rather than create a new template from scratch. You can start use the dbf header.txt and dbf Fields.txt templates as they stand and then build your own data template based on dbf records.txt.

A template file always starts with the word template followed by a title, then there will be the keyword description followed by a longer description – both title and description will be shown by template manager. You can also add comments – everything on a line after // will be ignored.

WinHex Template Tutorial

Paul Mullen

```
template "dbf header"

// Sample template by Paul Mullen, pcguru@the-answer.com

description "First 32 bytes of a dBase/xBase .dbf file"
```

Next we can define how the template is to be used: a good idea to guard against user errors. For now let's just use

```
appliedto file

requires 30 "00 00"
```

The first of these tells us to apply the template to a file, rather than a disk structure. The second is a check for validity of the file format, and just checks that the header ends with two zero bytes.

Now the file structure – which will begin with begin, and end with end!

Each line in between will consist of a type and a description. Depending on your preferred programming language, you can use alternative terms to describe each type of field. Common types (with alternatives) are:

```
hex
uint8 (byte)int8      - signed byte size integer
int (int16)           - 16 bit signed integer
uint (uint16, word)  - 16 bit unsigned integer
int32 (long)         - 32 bit unsigned integer
uint32 (dword)       - 32 bit unsigned integer
int64 (longlong)    - 64 bit signed integer
float (single)      - 32 bit (single precision) floating point
real                - 48 bit floating point
double              - 64 bit (double precision) floating point
longdouble (extended) -128bit floating point
char (string)       - ASCII text
char16 (string16)  - Unicode
```

So our dBase header is going to look like this:

```
begin
hex      "Version"
byte 3   "Last update, format YYMMDD"
uint32   "Number of records in file"
uint16   "Length of header"
uint16   "Data Record length"
hex 2    "Reserved, fill with 0"
byte     "Incomplete transaction"
byte     "Encryption flag"
hex 12   "dBaseIV multi-user"
byte     "Production index exists"
byte     "dBaseIV language"
hex 2    "always 0x00"
end
```

As a final touch, let's mark the two reserved fields with the keyword read-only. This causes those line to appear greyed out, so they can't be edited, and also forms convenient visual dividers in the record.

WinHex Template Tutorial

Paul Mullen

Ok, so when you look at the start of the states.dbf file, you should see that it was Version 3, last updated on 6/11/96 (11/6/96 for our European readers), contains 51 data records, which follow 257 bytes of header information, and each data record is 69 bytes long.

The rest of those 257 bytes of header consist of a number of field descriptions, each 32 bytes long, terminated by a single 0 byte. Arithmetic tells us that there must be seven fields in the file ($32 + 7 * 32 + 1 = 257$).

Each field in the data record is defined by a 32 byte entry as follows:

```
BYTES  DESCRIPTION
0-10   Field Name ASCII padded with 0x00
11     Field Type Identifier (see below)
12-15  Offset of field in record
16     Field length in bytes
17     Field decimal places
18-19  Reserved
20     dBaseIV work area ID
21-30  Reserved
31     Field is part of production index - 0x01 else 0x00
```

xbase fields types:

```
Code  Description
C     Character
D     Date, format YYMMDD
F     Floating Point
G     General - FoxPro addition
L     Logical, T:t,F:f,Y:y,N:n,?-not initialized
M     Memo (stored as 10 digits representing the dbt block number)
N     Numeric
P     Picture - FoxPro addition
```

A peculiarity of dBase files is that numbers (type N) are not stored in binary form but in characters (like Cobol, if there are any Cobol programmers still around). So every field will be of type char.

So this gives us the definition of dbf fields.txt. Since there will be more than one field description, we add the keyword multiple. This will add a new field to the data editing window, showing the current record number, and previous and next record buttons.

```
template "dbf field"

// Sample template by Paul Mullen, pcguru@the-answer.com

description "32 byte definition for each data field"

appliesto file

requires 10 00
multiple

begin
    char[10]          "Field Name (zero terminated)"
    read-only hex 1  "(zero terminator)"
    char[1]           "Field Type"
    int32             "offset from start of record"
```

WinHex Template Tutorial

Paul Mullen

```
byte          "Field length (bytes)"
byte          "Decimal places"
read-only hex 2 "(Reserved)"
byte          "Work area ID"
read-only hex 10 "(Reserved)"
byte          "Used in production index"
end
```

From browsing the field descriptions we see the following fields:

Type	Length	Dec	Name
N	12	3	Area
C	25		State Name
C	2		State FIPS
C	7		Region
C	2		Abbreviation
N	10		Pop 1990
N	10		Pop 1996

In addition every dbase record starts with a single character “deleted” marker – space if the record is valid, * if deleted.

So we get the following template

```
template "dbf sample data record"

// Sample template by Paul Mullen, pcguru@the-answer.com

description "applies to states.dbf"

appliesto file

multiple

begin
    char[1]          "*=deleted"
    char[12]         "Area"
    char[25]         "State Name"
    char[2]          "FIPS"
    char[7]          "Region"
    char[2]          "Abbreviation"
    char[10]         "Pop 1990"
    char[10]         "Pop 1996"
end
```

Finally, you may recall that when we inspected the header, it showed 51 records. Yet there are only 50 states. The additional “record” is not really a record at all but a dBase “end of file marker”, consisting of the single byte 0x1A. This value was used by the old CP/M operating system (and DEC operating systems before it) to mark the end of text files and was carried over into early versions of MsDos.

Now you should be ready to start tackling some more advanced templates: look at the other sample templates and just experiment!

```
template "dbf file header"
```

WinHex Template Tutorial

Paul Mullen

```
description "First 32 bytes of a dBase/xBase .dbf file"

appliesto file

requires 30 "00 00"

begin
    hex 1                                "Version"
    byte [3]                             "Last update, format YYYYMMDD"
    uint32                                "Number of records in file"
    uint16                                "Length of header"
    uint16                                "Data Record length"
    read-only hex 2  "(Reserved, fill with 0)"
    byte                                  "Incomplete transaction"
    byte                                  "Encryption flag"
    hex 12                                "dBaseIV multi-user"
    byte                                  "Production index exists"
    byte                                  "dBaseIV language option"
    read-only hex 2  "(always 0x00)"
end

template "dbf field"

// Sample template by Paul Mullen, pcguru@the-answer.com

description "32 byte definition for each data field"

appliesto file

requires 10 00
multiple

begin
    char[10]                             "Field Name (zero terminated)"
    read-only hex 1  "(zero terminator)"
    char[1]                               "Field Type"
    int32                                "offset from start of record"
    byte                                  "Field length (bytes)"
    byte                                  "Decimal places"
    read-only hex 2  "(Reserved)"
    byte                                  "Work area ID"
    read-only hex 10 "(Reserved)"
    byte                                  "Used in production index"
end

template "dbf sample data record"

// Sample template by Paul Mullen, pcguru@the-answer.com

description "applies to states.dbf"

appliesto file

multiple

begin
    char[1]                               "*=deleted"
    char[12]                              "Area"
```

WinHex Template Tutorial

Paul Mullen

```
char[25]      "State Name"  
char[2]      "FIPS"  
char[7]      "Region"  
char[2]      "Abbreviation"  
char[10]     "Pop 1990"  
char[10]     "Pop 1996"  
end
```