WORD.FMT

PSIONICS FILE - WORD.FMT

==========

Format of Word files

Last modified 1996-01-18

==========

Word files can be encrypted with a password. Encryption affects only theheader and data record type 8 (see below).

A word file begins with a header of the following form (U indicates valuesfor unencrypted files, and E values for encrypted files):

  Offset 0 (cstr): "PSIONWPDATAFILE"

  Offset 16 (word): format version number (U: 1, E: 256) Offset 18 (word): encryption algorithm version (U: 0, E: 1)

  Offset 20 to 28: encryption key check value (all $EA if unencrypted) Offset 29 to 35: copy of offset 20 to 26

  Offset 36 (word): U: $EAEA, E: 0

  Offset 38 (word): unused

The rest of the file consists of records. All records have the form: Offset 0 (word): record type

  Offset 2 (word): size of data portion (L)

  Offset 4 to L+3: data portion

Word files have record types 1 to 9; the word processor application createsthem in numerical order of type. Exactly one record of each type is used,except that there may be more than one record of types 6 and 7.

All distances and font sizes are in units of 0.05 points (i.e. a value of1440 represents one inch). All font names are represented by standard code numbers:

    -1 = Inherited (where permitted)   0 = Courier   17 = Emperor   40 = Greek

     1 = Pica   18 = Madeleine   41 = Kana   2 = Elite   19 = Zapf Humanist   42 = Hebrew   3 = Prestige   20 = Classic   44 = Russian

     4 = Letter Gothic   24 = Times Roman   48 = Narrator   5 =

Gothic 25 = Century 49 = Emphasis 6 = Cubic 26 = Palatino 50 = Zapf Chancery 7 = Lineprinter 27 = Souvenir 52 = Old English

8 = Helvetica 28 = Garamond 55 = Cooper Black 9 = Avant Garde 29 = Caledonia 56 = Symbol 10 = Spartan 30 = Bodoni 57 = Line Draw 11 = Metro 31 = University 58 = Math 7

12 = Presentation 32 = Script 59 = Math 8 13 = APL 33 = Script PS 60 = Dingbats 14 = OCR A 36 = Commercial Script 61 = EAN

15 = OCR B 37 = Park Avenue 62 = PC Line 16 = Standard Roman 38 = Coronet

Record type 1 holds information about the file. It is always 10 bytes: Offset 0 (word): cursor position within text record (type 8) Offset 2 (byte): each set bit indicates a character type should be shown

as symbols:

Bit 0: tabs

Bit 1: spaces

Bit 2: carriage returns

Bit 3: soft hyphens

Bit 4: forced line breaks

Offset 3 (byte): (Series 3a only)

Bits 0 to 1: status window: 0=none, 1=narrow, 2=wide Bits 4 to 5: zoom state: 0=smallest, ... 3=largest

Offset 4 (byte): 0=style bar off, 1=style bar on Offset 5 (byte): 0=file type is paragraph, 1=file type is line

Offset 6 (byte): outlining level

Offset 7 (byte): unused

Offset 8 (word): unused

Record type 2 holds information about printer set-up. It is always 58 bytes:

Offset 0 (word): page width

Offset 2 (word): page height (Note: the above fields assume that the paper orientation is "portrait")

Offset 4 (word): left margin

Offset 6 (word): top margin

Offset 8 (word): width of printing area

Offset 10 (word): height of printing area (Note: these four fields have only been checked for portrait)

Offset 12 (word): header offset (bottom of header to top of text) Offset 14 (word): footer offset (bottom of footer to bottom of text) Offset 16 (word): paper orientation: 0=portrait, 1=landscape

Offset 18 (word): unknown

Offset 20 (word): first page to print (1=first page) Offset 22 (word): last page to print ($FFFF=end of file)

Offset 24 (word): header font code number

Offset 26 (byte): header style

- Bit 0: underline
- Bit 1: bold
- Bit 2: italic
- Bit 3: superscript
- Bit 4: subscript

Offset 27 (byte): unused

Offset 28 (word): header font size

Offset 30 (byte): header alignment:

- 0 = left
- 1 = right
- 2 = centered
- 3 = justified
- 4 = two column
- 5 = three column

Offset 31 (byte): header on first page: 0=no, 1=yes Offset 32 to 39: as 24 to 31, but apply to footer, not header

Offset 40 (word): page number of first page minus 1

Offset 42 (word): number of pages Offset 44 (word): page number style: 0="1,2,3", 1="I,II,III", 2="i,ii,iii"

Offset 46 (word): base font code number

Offset 48 (byte): base style (as offset 26)

Offset 49 (byte): unused

Offset 50 (word): base font size

Offset 52 (byte): paper size code:

- 0 = A4 (11906 x 16838)
- 1 = Custom
- 2 = Executive (10440 x 15120)

```
    3 = Legal     (12240 x 20160)
    4 = Letter    (12240 x 15840)
    5 = Monarch   ( 5580 x 10800)
    6 = DL        ( 6236 x 12472)
  Offset 53 (byte): widows/orphans allowed: 0=no, 1=yes
  Offset 54 (long): unusedThe base font code, style, and font size are unused by Word (and
should be set
to code 0, style 0, size 240). Other applications using this record layout mayuse them and
provide means to set them.
Record type 3 holds information about the printer driver:  Offset  0 (byte): printer driver
model number
  Offset  1 (cstr): printer driver libraryA printer driver library can support several similar
printers; the model number
specifies which is selected.
Record types 4 and 5 hold cstrs giving the header and footer text respectively.
Record types 6 and 7 have a similar layout. Record type 6 describes a styleand uses all 80
bytes. Record type 7 describes an emphasis and uses only the
first 28 bytes.
  Offset  0 to   1: short code, as uppercase letters
  Offset  2 (cstr): full name
  Offset 18 (byte):
    Bit 0: 0=style, 1=emphasis
    Bit 1: set if style or emphasis undeletable
    Bit 2: set for default style or emphasis
  Offset 19 (byte): unused
  Offset 20 (word): font code number (can be inherited)  Offset 22 (byte): style (bits inherited
must be clear in this byte)
    Bit 0: underline
    Bit 1: bold
    Bit 2: italic
    Bit 3: superscript (available in emphasis only)    Bit 4: subscript   (available in emphasis
only)
  Offset 23 (byte): unused
  Offset 24 (word): font size
  Offset 26 (byte):
    Bit 0: inherit underline setting
    Bit 1: inherit bold setting
```

```
    Bit 2: inherit italic setting    Bit 3: inherit superscript setting (available in emphasis
only)
    Bit 4: inherit subscript setting   (available in emphasis only)
  Offset 27 (byte): unused
  Offset 28 (word): left indent
  Offset 30 (word): right indent
  Offset 32 (word): first line indent  Offset 34 (word): alignment: 0=left, 1=right, 2=centred,
3=justified
  Offset 36 (word): line spacing
  Offset 38 (word): space above paragraph
  Offset 40 (word): space below paragraph
  Offset 42 (byte): spacing controls:
    Bit 0: set to keep with next
    Bit 1: set to keep together
    Bit 2: set to start new page
  Offset 43 (byte): unused
  Offset 44 (word): Outline level (1 to 9)
  Offset 46 (word): number of tab stops set
  Offset 48 (word): position of first tab stop  Offset 50 (word): type of first tab stop:
0=left, 1=right, 2=centred
  Offset 52 to  55: as offsets 48 to 51 for second tab stop  Offset 56 to  79: as offsets 48 to
51 for third to eighth tab stops
Record type 8 holds the actual text. The following bytes have special
meanings:
   0 = paragraph separator
   7 = unbreakable hyphen
  14 = soft hyphen (displayed only if used to break line)
  15 = unbreakable space
The record is modified in encrypted files.
Record type 9 consists of a sequence of blocks giving the style and emphasisfor the text; each
block covers some number of consecutive bytes, and theblocks between them cover the entire text.
No block crosses a paragraphboundary, but the last block of the paragraph includes the zero byte
separating
it from the next paragraph. Each block is 6 bytes:
  Offset  0 (word): number of bytes covered
  Offset  2 to   3: shortcode of style applied  Offset  4 to   5: shortcode of emphasis applied
The last block should cover an extra byte (an imaginary extra zero separator),so that the sum of
```

the bytes covered is one more than the size of the type 8
record.
Encryption

Word files can be encrypted with a password. If so, this fact is indicatedin the header (see above), and the type 8 record is modified.
The password is used to generate two 9 byte sequences, called the key valueand the key check value; there is no obvious relationship between the twosequences. The key check value is written into the header, while the keyvalue is used for the actual encryption. The key value is generated with thesystem call GenMaskInit; there is no documentation of the algorithm used to generate the check value.
[Note: different passwords may generate the same key value but different keycheck values, or vice versa (passwords "AA" and "AAA" are an example of the
latter). This can confuse Word badly.]
Encryption is carried out using the system call GenMaskEncrypt. Forconvenience, the description is repeated here with an example.
The first 7 bytes of the key value are appended to all 9 to generate a 16byte sequence. This is then repeated to generate a sequence of the same sizeas the type 8 record, and placed in 1-to-1 correspondence with it. In otherwords, for byte N of the type 8 record, the corresponding key byte is given
by byte N AND $F of the 16 byte sequence, counting from 0 (or, in C notation,byte N % 16 % 9 of the 9 byte sequence). The key byte is then added to theplain text byte, modulo $100, to get the encrypted byte.
For example, suppose that the text is a single paragraph containing: Jackdaws love my 21 big sphinxes of quartz.
and the key value is:
  $91 $20 $E3 $92 $42 $F9 $5C $57 $A9(which corresponds to the password "AA") then the encrypted type 8 record
is determined as follows:
  Text: J a c k d a w s l o v e m y 4A 61 63 6B 64 61 77 73 20 6C 6F 76 65 20 6D 79
  Key: 91 20 E3 92 42 F9 5C 57 A9 91 20 E3 92 42 F9 5C Encr: DB 81 46 FD A6 5A D3 CA C9 FD 8F 59 F7 62 66 D5
  Text: 2 1 b i g s p h i n x e s 20 32 31 20 62 69 67 20 73 70 68 69 6E 78 65 73
  Key: 91 20 E3 92 42 F9 5C 57 A9 91 20 E3 92 42 F9 5C Encr: B1 52 14 B2 A4 62 C3 77 1C 01 88 4C 00 BA 5E CF

```
    Text:    o  f     q  u  a  r  t  z  .
          20 6F 66 20 71 75 61 72 74 7A 2E
    Key:  91 20 E3 92 42 F9 5C 57 A9 91 20
    Encr: B1 8F 49 B2 B3 6E BD C9 1D 0B 4E
```

Of course, standard Kerchoffs techniques can be used to break the encryption.

Revision #1

Created Thu, Jan 24, 2019 10:35 AM by Alex

Updated Thu, Jan 24, 2019 10:36 AM by Alex