INTERNATIONAL STANDARD

ISO/IEC 9592-3

Second edition
1997-11-15

Information technology — Computer graphics and image processing — Programmer's Hierarchical Interactive Graphics System (PHIGS) —

Part 3:
Specification for clear-text encoding of archive file

Technologies de l'information — Infographie et traitement de l'image — Interface de programmation du système graphique hiérarchisé (PHIGS) —

Partie 3: Spécification du codage mode texte en clair du fichier d'archive

Reference number
ISO/IEC 9592-3:1997(E)

Contents

1 Scope 1
2 Normative references 2
3 Definitions 3
4 Clear text encoding format 4
4.1 Notational conventions 4
4.2 Archive file format 4
4.2.1 Introduction 4
4.2.2 Character repertoire 5
4.2.3 Separators 6
4.2.3.1 Element separators 6
4.2.3.2 Parameter separators 6
4.2.3.3 Comments in the archive file 7
4.2.4 Encoding of parameter types 7
4.2.4.1 Integer-bound types 7
4.2.4.2 Real-bound types 8
4.2.4.3 String-bound types 9
4.2.4.4 Enumerated types 9
4.2.4.5 Derived types 9
4.2.5 Forming archive file element names 15
4.2.5.1 Terms deleted 15
4.2.5.2 Words added 15
4.2.5.3 Words used unabbreviated 16
4.2.5.4 Abbreviations 16
4.2.5.5 Abbreviating compound types 17
4.2.5.6 Sentinel character sequence 18
4.2.5.7 The derived archive file element names 18
4.3 Encoding the PHIGS archive file elements 22
4.3.1 Encoding delimiter elements 22
4.3.2 Encoding archive file descriptor elements 22
4.3.3 The structure element production 22
4.3.4 Encoding output primitive elements 26
4.3.5 Encoding attribute elements 32
4.3.6 Encoding modelling transformation elements 40
4.3.7 Encoding miscellaneous elements 41
4.3.8 Encoding external elements 42
4.4 Clear-text encoding conformance 42

A Clear-text encoding-dependent formal grammar 43

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 9592-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

This second edition cancels and replaces the first edition (ISO/IEC 9592-3:1989), which has been technically revised. It also incorporates Amendment 1:1992.

ISO/IEC 9592 consists of the following parts, under the general title *Information technology — Computer graphics and image processing — Programmer's Hierarchical Interactive Graphics System (PHIGS)*:

- *Part 1: Functional description*
- *Part 2: Archive file format*
- *Part 3: Specification for clear-text encoding of archive file*

Annex A of this part of ISO/IEC 9592 is for information only.

Introduction

The clear-text encoding of the PHIGS archive file provides a representation of the archive file syntax that is easy to type, edit, and read. It allows an archive file to be edited with any standard text editor, using the internal character code of the host computer system. The primary objectives are:

a) HUMAN EDITABLE: The clear-text encoding should be able to be hand-edited or, if desired, hand-constructed.

b) HUMAN-FRIENDLY: The clear-text encoding should be easy and natural for people to read and edit. Although what is easiest and most natural is a subjective judgement that varies among users, contributing factors such as ease of recognition, ease of remembering, avoidance of ambiguity, and prevention of mistyping have all been considered.

c) MACHINE-READABLE: The clear-text encoding should be able to be parsed by software.

d) USABLE IN A WIDE VARIETY OF EDITORS: The clear-text encoding should not have any features that make it difficult to edit in normal text editors.

e) INTERCHANGEABLE BETWEEN DIVERSE SYSTEMS: The clear-text encoding should be encoded in such a way as to maximize the set of systems which can utilize it. No assumptions should be made as to word size or arithmetic modes used to interpret the archive file.

f) USES STANDARDIZED ABBREVIATIONS: Where language encoding of other graphics standards have established standard abbreviations, or where common practice in the data processing and graphics industries has established well-known abbreviations, these abbreviations are used. In accordance with the principle of "least astonishment", this approach should minimize the time needed to learn to use this encoding.

This part of ISO/IEC 9592 draws extensively for its model of an archive file format on ISO 8632. The set of characters needed to implement the clear-text encoding is a subset of those included in national versions of ISO 646. Any character set that can be mapped to and from that subset may be used to implement the encoding.

Information technology – Computer graphics and image processing –

Programmer's Hierarchical Interactive Graphics System (PHIGS) – Part 3: Specification for clear-text encoding of archive file

1 Scope

This part of ISO/IEC 9592 specifies a clear-text encoding of the PHIGS archive file. For each of the archive file elements specified in ISO/IEC 9592-2, a clear text encoding is specified. This part of ISO/IEC 9592 specifies the overall format of the archive file and the means by which comments may be interspersed in the archive file.

This encoding of the PHIGS archive file allows archive files to be created and maintained in a form which is simple to type, easy to edit and convenient to read.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9592. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 9592 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 646:1991, *Information technology - ISO 7-bit coded character set for information interchange.*

ISO/IEC 2022:1994, *Information technology - Character code structure and extension techniques.*

ISO 6093:1985, *Information processing - Representation of numerical values in character strings for information interchange.*

ISO/IEC 8632:1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information*

- *Part 1 : Functional description*
- *Part 2 : Character encoding*
- *Part 3 : Binary encoding*
- *Part 4 : Clear text encoding*

3 Definitions

For the purposes of this part of ISO/IEC 9592 the following definitions apply.

NOTE

As far as possible, graphics terminology which is commonly accepted and consistent with other graphics Standards is used.

3.1 archive file descriptor: A group of elements that describe the functional capabilities needed to process the archive file.

3.2 archive file generation: The process that produces a PHIGS archive file.

3.3 archive file retrieval: The process that reads a PHIGS archive file, retrieves the contents, and transfers the result to the PHIGS centralized structure store.

4 Clear-text encoding format

4.1 Notational conventions

a) Unbracketed strings are terminals of this grammar which appear exactly, subject to the variations on case and null characters given in 4.2.2.

b) Bracketed strings are either non-terminals (with further productions given), character symbol names (such as COMMA), or parameters of the PHIGS archive file element in the form <x:y>.

c) The following metasymbols define productions, grouping, and repetition.

::=	→	''becomes'' or ''is realized as''
<...>*	→	*star closure (0 or more occurrences)
<...>+	→	plus closure (1 or more occurrences)
<...>o	→	optional (exactly 0 or 1 occurrences)
<x:y>	→	parameter type x with meaning y
x | y | z	→	exactly one of the items x, y and z (any of x, y and z may be multiple non-terminals if there is no ambiguity)
<x|y|z>	→	exactly one of x or y or z, brackets delineate the scope
{...}	→	a comment (not part of the production)
<...>(n)	→	exactly n occurrences, n=0,1,2,...
<x,..,x>	→	a list of occurences of x each separated by <SEP>

The meaning of the last notation can be expressed as follows:

<A,..,A> ::= <A> < <SEP> <A> >*

d) SPACES are used for readability in the grammar description; SPACES in the actual archive file are indicated through the separator productions given in 4.3.3.

e) The metasymbols used in describing the grammar do not appear in the actual archive file.

4.2 Archive file format

4.2.1 Introduction

A clear-text encoding of a PHIGS archive file consists of a stream of characters forming a series of elements, each of which starts with an element name and ends with one of the element delimiters, either the SLASH character (also known as SLANT or SOLIDUS) or the SEMICOLON character. These characters do not act as element delimiters when occurring within the bounds of a string parameter, as defined below.

The order of elements within a clear-text encoding of a PHIGS archive file is specified by ISO/IEC 9592-2. This specifies a formal grammar over the following eight symbols:

BEGIN ARCHIVE FILE
END ARCHIVE FILE
BEGIN STRUCTURE
END STRUCTURE
ARCHIVE FILE VERSION
ARCHIVE FILE DESCRIPTION
STRUCTURE ELEMENT
EXTERNAL ELEMENT

Each of these symbols is treated as a non-terminal in the formal grammar that follows. Taken together, the formal

grammar of ISO/IEC 9592-2 and this part of ISO/IEC 9592 provide a formal grammar for a PHIGS archive file over the ISO/IEC 646 character set.

4.2.2 Character repertoire

In order that the metasymbols used in describing the grammar do not appear in the actual archive file, the character repertoire of the clear-text encoding will be limited to those characters enumerated below, except for string parameters, which, at the minimum, support the full ISO/IEC 646 character set and, optionally may include characters which shift into other character sets. Each string is assumed to start in the ISO/IEC 646 character set.

Upper-case characters:
"A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M",
"N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"

- Lower-case characters:
"a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m",
"n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"

Digits:
"0", "1", "2", "3", "4", "5", "6", "7", "8","9"

" " (SPACE character)
"+" (PLUS character)
"-" (MINUS character)
"#" (NUMBER SIGN)
";" (SEMICOLON character)
"/" (SLASH, SLANT, or SOLIDUS character)
"(" (LEFT or OPEN PARENTHESIS character)
")" (RIGHT or CLOSE PARENTHESIS character)
"," (COMMA character)
"." (DECIMAL POINT or PERIOD character)
"'" (APOSTROPHE or SINGLE QUOTE character)
"""" (DOUBLE QUOTE character)
"_" (UNDERSCORE character)
"$" (DOLLAR SIGN or CURRENCY symbol)
"%" (PERCENT SIGN character)

Lower-case characters are considered to be the same as upper-case characters when occurring outside of string parameters. Any combination of lower-case and upper-case characters may be used within an element or enumerated parameter name.

The UNDERSCORE and DOLLAR SIGN symbols are defined as "null characters" within this encoding. They may appear anywhere within an archive file, and are mandated to have no effect on parsing (outside of string parameters). They are available for the generator or editor of an archive file to use in enhancing readability of tokens. For example, the following are all equivalent:

linetype, LINETYPE, LineType, line_type, $LINETYPE, L_I_N_E$T_Y_P_E.

The following are all equivalent:

123456, $123456, 123_456, $123_456, $12$34$56.

Those control characters that are format effectors (BACKSPACE, CARRIAGE RETURN, LINEFEED, NEWLINE, HORIZONTAL TAB, VERTICAL TAB, and FORMFEED) are permitted in the archive file, but are treated as SPACE characters (that is, as soft delimiters) by the archive file interpreter whenever they occur outside of a string parameter. They may be used to assist in formatting the archive file to improve its readability. A PHIGS archive file written in the clear-text encoding is considered not to be a conforming interchange if it includes characters other than those listed in

the repertoire and the format effectors (outside of string parameters). Implementation-dependent extensions which require use of characters other than the above should be embedded in the string parameters of the GSE or APPLICATION DATA elements, or in comments.

The code set of the characters is not fixed by this part of ISO/IEC 9592. In order to accomplish the objective of editability, it is permitted to encode the clear-text encoding using the character set codes native to the system. It is presumed that standard conversion facilities can be used in translating clear-text PHIGS archive files from one system's character set codes to another, consistent with the treatment of other text files being transferred between systems. The ISO/IEC 646 codes should be used to encode clear-text archive files for transport between diverse systems.

Null characters or format effectors outside of text strings which do not exist in the target system's encoding may be dropped in such translation, and lower-case letters translated to upper case as necessary, without altering the information content of the archive file. Likewise, the two statement delimiter characters are interchangeable and may be changed in such a translation without affecting the information content of the archive file. The two string delimiter characters are interchangeable, but any translation shall correctly handle the possible occurrence of either string delimiter character within the string parameter.

4.2.3 Separators

4.2.3.1 Element separators

<TERM> ::= <OPTSEP> <SLASH | SEMICOLON> <OPTSEP>

The SEMICOLON and SLASH characters may be used to delimit elements in a clear text archive file. These elements do not, however, terminate an element when they occur within a string parameter, as described in 4.2.4.3.

The elements of the archive file are not terminated by the ends of records, as indicated by control characters such as CR (carriage return) or LF (linefeed). Multiple elements may exist on one line, and any element may extend over multiple lines.

4.2.3.2 Parameter separators

The following productions are used in the clear-text encoding for parameter separators:

<SEPCHAR> ::= <SPACE | CARRIAGE RETURN | LINEFEED | HORIZONTAL TAB | VERTICAL TAB | FORMFEED>
<SOFTSEP> ::= <SEPCHAR>+
<OPTSEP> ::= <SEPCHAR>*
<HARDSEP> ::= <OPTSEP> <COMMA> <OPTSEP>
<SEP> ::= <SOFTSEP> | <HARDSEP>

Most commands require a SOFTSEP after the element name (e.g., at least one space). This permits element names to be formed from a mixture of alpha and numeric characters.

The separator between parameters is usually a SEP. This format permits omission of parameters. (Two consecutive COMMAs indicate an omitted parameter.)

Since the enclosing APOSTROPHE or DOUBLE QUOTE character sufficiently delineates string parameters, and the statement delimiter SLASH also sets off the data on either side of it, the separators between these characters and adjacent parameters or element names are optional (OPTSEP).

SEPCHAR characters are not permitted within a name (element or enumerated type), or within the representation of a numeric parameter. Any place where a SEPCHAR is permitted (other than inside a string parameter), an arbitrary number of SEPCHARs may be used.

4.2.3.3 Comments in the archive file

Comments may be included in a clear-text archive file to enhance its readability and usefulness. Some uses of comments might be to document hand-cdited changes to the archive file, or as "notes to one's self" made while reading an archive file. To includc other forms of nongraphical information in an archive file, it is suggested that the EXTERNAL clement be used.

Comments are encoded as a series of printing characters and <SEPCHAR>s surrounded by "%" (PERCENT SIGN) characters. The text of the comment may not include this comment delimiter character.

Comments may be included any place that a separator may be used, and are equivalent to a <SOFTSEP>; they may be replaced by a SPACE character in parsing, without affecting the meaning of the archive file.

4.2.4 Encoding of parameter types

4.2.4.1 Integer-bound types

Integers, integer coordinates and indices are all bound to signed integers, indicated in the encoding as I.

<I>	::=	<decimal integer> | <based integer>
<decimal integer>	::=	<sign>o <digit>+
<sign>	::=	<PLUS SIGN> | <MINUS SIGN>
<digit>	::=	0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<based integer>	::=	<sign>o <base> <NUMBER SIGN> <extended digit>+
<base>	::=	2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16
<extended digit>	::=	<digit> | A | B | C | D | E | F | a | b | c | d | e | f

The null characters are permitted within numbers, but are not shown in the productions for simplicity.

A decimal integer has an optional sign and at least one digit. If the sign appears, it immediately precedes the number with no intervening SPACE (or other <SEPCHAR>) characters allowed.

A based integer has an optional sign, a base (an unsigned integer in the range 2..16 inclusive, represented in base 10), a "#", and a string of one or more extended digits. If the sign appears, it immediately precedes the number with no intervening SPACE (or other <SEPCHAR>) characters allowed. The extended digits used shall be valid for the base named or the archive file is not conforming; e.g., for base 8 the digits "8", "9", etc. are not valid, for base 2 only the digits "0" and "1" are valid, and so forth. Case is not significant for the extended digits.

If the sign is omitted for either form, the number is considered non-negative.

Both the base and the <extended digit>+ are interpreted as unsigned numbers, and the final result negated if a MINUS SIGN preceded the number. No assumptions should be made as to the word size of thc archivc file retrieval process, or whether the underlying arithmetic is one's complement, two's complement, or sign-magnitude. For example, -1 would be encoded in hexadecimal as -16#1, -16#0001, etc. rather than 16#FFFF. Of course, archive files may be created utilizing prior knowledge of thc intended target machine, but any such assumptions will limit the portability of archive files and arc discouraged.

The PHIGS functional data types "C", "G2", "G3", "GS", "AI", "PI", "EI", "FN" and "WI" are represented by integers in this encoding.

Some examples are:

0, 007, -5, +123_456

The following are equivalent:

65535, 16#FFFF, 16#ffff, 8#177777, 2#1111_1111_1111_1111

The following are equivalent:

-32_768, -16#8000, -8#100000, -2#10000000_00000000

Interpretation of numerically bound parameters will be "free field", that is, there is an implied radix point to the right of the rightmost digit, and neither leading nor trailing spaces are significant. Leading zeroes are not significant.

4.2.4.2 Real-bound types

Reals and real coordinates are bound to real numbers, indicated in the encoding as R. These are written as either explicit-point or scaled_real numbers (or decimal integers, where appropriate).

```
<R>                      ::=  <explicit_point_number> | <scaled_real_number> | <decimal integer>
<explicit_point_number>  ::=  <sign>o
                              <
                              < <digit> + <PERIOD> <digit>* >
                              |
                              < <digit>* <PERIOD> <digit>+ >
                              >
<scaled_real_number>     ::=  <body> < E | e > <exponent>
<body>                   ::=  <explicit_point_number> | <decimal integer>
<exponent>               ::=  <decimal integer>
```

The interpretation of the scaled_real_number is the same as standard scientific notation (similar to FORTRAN "E" format), where the number represented by <body> is multiplied by 10 taken to the power <exponent>.

There shall be at least one digit in an explicit_point_number and in the body of a scaled_real_number, which in the case of a single-digit number may appear on either side of the radix point. It is recommended but not required that there be at least one digit before the radix point, for numbers with only a fractional part. Zero may be encoded as "0.", ".0", "0.0", "0", etc, although the second form is not recommended.

In the case of a scaled_real_number number (one where an "E" or "e" appears), at least one digit shall appear in the <exponent>. No SPACE or other <SEPCHAR> characters shall be included between the <body> and the "E" or "e", or between the "E" or "e" and the <exponent>.

The interpretation of parameters bound to this data type will be "free field", that is, if there is an explicit radix point, it sets the radix point of the internal representation, and neither leading nor trailing spaces or zeroes are significant. If the radix point is omitted, it is implied to be at the right of the rightmost digit of the explicit_point_number or of the <body> of the scaled_real_number. Thus, decimal I-format numbers may appear in a conforming archive file for parameters bound to real numbers when there is no fractional part.

For real numbers in all formats, the only base of representation that shall be utilized is base 10.

If the <sign> ("+" or "-") is omitted, the number is assumed to be non-negative. If the sign is present, it immediately precedes the body of the number, with no SPACE (or other <SEPCHAR>) characters allowed between it and the leftmost digit or radix point of the body of the number.

COMMA, SPACE and other <SEPCHAR> characters are not allowed within a number, but <NULLCHAR> characters may be included (and have no effect on parsing).

Examples:

3.14159
7.853982E-7
271828e-5
42
-.04321 (not recommended form)
-0.043_21
42E2
$5321.46

4.2.4.3 String-bound types

String parameters are represented as character strings immediately surrounded by a matched pair of either APOSTROPHE (SINGLE QUOTE) or DOUBLE QUOTE characters.

If an APOSTROPHE is needed in a string delimited with APOSTROPHE characters, it is represented by two adjacent APOSTROPHE characters at that position in the string. Likewise, if a DOUBLE QUOTE character is needed in a string delimited with DOUBLE QUOTE characters, it is represented by two adjacent DOUBLE QUOTE characters. For example, the following are equivalent:

"Murphy's Law: ""If it can go wrong, it will.""";
'Murphy''s Law: "If it can go wrong, it will."';

DATA RECORD (D) data type is represented as a string in this encoding.

STRING parameters are indicated in the encoding as S.

4.2.4.4 Enumerated types

Enumerated types are bound to names, just as element names are. Where an implementation wishes to support private enumerated type values, these shall be encoded as the letters "PRIV" followed by a string of <alpha | digit | null_character>*.

4.2.4.5 Derived types

In addition to the I, R, and S parameter formats, the following abbreviations are used as shorthand for the productions shown.

<(>	::=	<LEFT PAREN> <OPTSEP>
<)>	::=	<OPTSEP> <RIGHT PAREN>
CELLLIST	::=	<I:CELL,..,I:CELL>o
CELLROW	::=	<SEP> <CELLLIST> | <SEP> <(> <CELLLIST> <)>
COLRCURVE	::=	<(> <I:ORDER> <SEP> <RLIST:KNOTS> <SEP> <RATIONAL | NONRATIONAL> <SEP> <I:COLOUR TYPE> <SEP> <COLRVLIST:CONTROL_POINTS> <)>
COLRSURF	::=	<(> <I:U_ORDER> <SEP> <I:V_ORDER> <SEP> <RLIST:U_KNOTS> <SEP> <RLIST:V_KNOTS> <SEP> NONRATIONAL <SEP> <I:COLOUR TYPE> <SEP> <COLRVLISTS:CONTROL_POINTS> <)> {Each COLRVLIST contains control points along the *u* dimension.}
COLRSURFH	::=	<(> <I:U_ORDER> <SEP> <I:V_ORDER> <SEP> <RLIST:U_KNOTS> <SEP> <RLIST:V_KNOTS> <SEP> RATIONAL <SEP> <I:COLOUR TYPE> <SEP> <COLRVLISTS:CONTROL_POINTS> <)> {Each COLRVLIST contains control points along the *u* dimension.}
COLRV	::=	<I:COLOUR_INDEX> | <COORDLIST:COLOUR_COORDINATES>