

INTERNATIONAL STANDARD

ISO/IEC
9593-3

First edition
1990-04-15

Information technology — Computer graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) language bindings —

Part 3 : Ada

*Technologies de l'information — Infographie — Interfaces langage avec PHIGS —
Partie 3 : Ada*



Reference number
ISO/IEC 9593-3 : 1990 (E)



Documents

Document Sharing Hub

Contents

Foreword iv

Introduction v

1 Scope 1

2 Normative references 2

3 Principles 3

 3.1 Conformance 3

 3.2 Implications of the Language 3

 3.2.1 Functional Mapping.....3

 3.2.2 Implementation and Host Dependencies 4

 3.2.3 Error Handling.....4

 3.2.4 Data mapping 4

 3.2.5 Multi-tasking.....6

 3.2.6 Packaging 6

 3.2.7 Application Program Environment 7

 3.2.8 Registration 7

4 Tables 8

 4.1 Abbreviations used in procedure names 8

 4.1.1 List of procedures using the abbreviations 8

 4.1.2 Alphabetical by bound name 11

 4.1.3 Alphabetical PHIGS functions 15

 4.2 Data type definitions 15

 4.2.1 Abbreviations used in the data type definitions 16

 4.2.2 Alphabetical list of type definitions 16

 4.2.3 Alphabetical List of Private Type Definitions 66

 4.2.4 List of Constant Declarations.....68

 4.2.5 PHIGS Configuration Values.....69

 4.3 Error Codes 71

 4.3.1 Precluded Error Codes 72

5 Functions in the Ada Binding of PHIGS	73
5.1 Control functions.....	73
5.2 Output primitive functions.....	74
5.3 Attribute specification functions.....	77
5.4 Transformation and clipping functions.....	84
5.5 Structure content functions.....	91
5.6 Structure manipulation functions.....	94
5.7 Structure display functions.....	95
5.8 Structure archive functions.....	95
5.9 Input functions.....	98
5.10 Metafile functions.....	106
5.11 Inquiry functions.....	107
5.12 Error control functions.....	132
5.13 Special interface functions.....	133
5.14 Additional Functions.....	134
5.14.1 Subprograms for Manipulating Input Data Records.....	134
5.14.2 PHIGS Generic Coordinate System Package.....	138
5.14.3 PHIGS Generic List Utility Package.....	141
5.14.4 PHIGS Name Set Facility Package.....	144
5.14.5 Deallocation of structure element records.....	147
5.14.6 Metafile Function Utilities.....	149
5.15 Conformal Variants.....	149
Annexes	
A Compilable PHIGS Specification	151
B Cross Reference Listing of Implementation Defined Items	243
C Example Programs	245
C.1 Example Program 1: STAR.....	245
C.2 Example Program 2: IRON.....	248
C.3 Example Program 3: DYNASTAR.....	255
C.4 Example Program 4: TRANSFORM POLYLINE.....	261
C.5 Example Program 5: SHOW_LINETYPES.....	269
D PHIGS Multi-Tasking	274
E Index	279

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) together form a system for worldwide standardization as a whole. 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 approval before their acceptance as International Standards. They are approved in accordance with procedures requiring at least 75 % approval by the national bodies voting.

International Standard ISO/IEC 9593-3 was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*.

Introduction

ISO/IEC 9592 is specified in a language independent manner and needs to be embedded in language dependent layers (language bindings) for use with particular programming languages.

The purpose of this part of ISO/IEC 9593 is to define a standard binding of PHIGS to the Ada computer programming language.

Information technology — Computer graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) language bindings —

Part 3 : Ada

1 Scope

ISO/IEC 9592 specifies a language independent nucleus of a graphics system. For integration into a programming language, PHIGS is embedded in a language dependent layer obeying the particular conventions of that language. This part of ISO/IEC 9593 specifies such a language dependent layer for the Ada computer programming language.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9593. 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 9593 are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 8652 : 1987, *Programming languages - Ada (Endorsement of ANSI Standard 1815A-1983)*.

ISO/IEC 9592-1 : 1989, *Information processing systems - Computer graphics - Programmer's Hierarchical Interactive Graphics System (PHIGS) - Part 1: Functional description*.

3 Principles

The PHIGS Binding to Ada is intended to be implementation independent except as it can be adapted by the PHIGS CONFIGURATION package. The PHIGS Binding to Ada makes no assumptions concerning the availability of implementation dependent facilities defined by the Ada language. It does, however, limit the use of multi-tasking as described in 3.2.5. The Ada compiler shall be able to support the number of declarations contained in this PHIGS Binding to Ada.

This binding does not make any assumptions regarding the machine representation of the predefined Ada numeric types.

This binding assumes that the application programmer will supply an error file name, archive file names, and connection identifiers that are in an acceptable format for the Ada implementation.

This binding makes no assumptions regarding the format of a string specifying an error file name, archive file names, or connection identifiers for devices or metafiles.

3.1 Conformance

This binding incorporates the rules of conformance defined in the PHIGS Standard (ISO/IEC 9592) for PHIGS implementations, with these additional requirements specifically defined for Ada implementations of PHIGS.

The following criteria are established for determining conformance or non-conformance of an implementation to this binding:

- The semantics of an implementation shall be those stated in ISO/IEC 9592 as modified or extended for Ada as stated in this part of ISO/IEC 9593.
- The package corresponding to PHIGS shall be an available Ada library unit, with all names as specified by this part of ISO/IEC 9593.

3.2 Implications of the Language

3.2.1 Functional Mapping

The functions of PHIGS are all mapped to Ada procedures. The mapping utilizes a one-to-one correspondence between the PHIGS functions and Ada procedures except for the PHIGS function INQUIRE TEXT EXTENT. This function is mapped to two overloaded Ada functions each named INQ_TEXT_EXTENT one which

supports modelling text and one which supports annotation text. Certain functions required by the binding but not defined by PHIGS are mapped to Ada functions and procedures.

3.2.2 Implementation and Host Dependencies

There are a number of implementation and host dependencies associated with the Ada compiler and runtime system used. These will affect the portability of application programs and their use of PHIGS. The application programmer should follow accepted practices for ensuring portability of Ada programs to avoid introducing problems when rehosting the application on another system. Implementation dependencies include runtime storage management and processor management.

3.2.3 Error Handling

The inquiry functions utilize error indicator parameters for the error returns, and do not raise Ada exceptions. The application program shall ensure that these error indicators are checked before attempting to access other parameters, since Ada implementations are not required to raise an exception if an undefined value is accessed.

The error handling requirements of PHIGS can be summarized as follows:

1. By default, a procedure named `ERROR_HANDLING` will be provided that simply reports the error by calling `ERROR_LOGGING`. This is called from the PHIGS function that detects the error.
2. The `ERROR_HANDLING` procedure may be replaced by one defined by the user.

The procedure `ERROR_HANDLING` is defined as a library subprogram:

```
with PHIGS_TYPES;
use PHIGS_TYPES;
procedure ERROR_HANDLING
  (ERROR_INDICATOR : in ERROR_NUMBER;
   PHIGS_FUNCTION  : in STRING;
   ERROR_FILE      : in FILE_ID := DEFAULT_ERROR_FILE);

-- The procedure ERROR_HANDLING is defined as a library subprogram,
-- and is not declared within package PHIGS.
```

This binding defines two different bodies for this subprogram; each shall be supplied by the implementation. The default body is the one required by PHIGS semantics. It simply calls `ERROR_LOGGING` and returns. The second body calls `ERROR_LOGGING` and then raises the exception `PHIGS_ERROR`. The PHIGS function shall be written so as not to handle `PHIGS_ERROR` (this is a requirement of the implementation). Thus, by Ada rules, the exception will be propagated back to the application program that called the PHIGS function in which the error was detected.

The means by which the user replaces the default body by either the exception-raising version or another one of his or her choosing is dependent upon the Ada library manager. Some implementations support multiple versions of a body with a single specification or otherwise allow hierarchical libraries with the sharing of common units. In other implementations, it may be necessary to duplicate the PHIGS library for each version of `ERROR_HANDLING`.

3.2.4 Data mapping

The simple and compound data types of PHIGS are bound to a variety of Ada scalar and compound types. Constraints on permitted values are reflected where possible in the type definitions. The general correspondence between the PHIGS data types and Ada binding data types is summarized below:

- PHIGS integer types (I) are mapped to Ada integer types.
- PHIGS real types (R) are mapped to Ada floating-point types.
- PHIGS string types (S) are mapped to the predefined Ada type STRING, or to a type providing for variable length strings.
- PHIGS point types (P2, P3) are mapped to Ada record types.
- PHIGS vector types (V2, V3) are mapped to Ada record types.
- PHIGS enumeration types (E) are mapped to Ada enumeration types.
- PHIGS name types (NM) are mapped to an Ada integer type. The PHIGS name set composite type SET(NM) is mapped to an Ada private type. A set of subprograms for operating on objects of this private type is explicitly defined by this binding.
- PHIGS filter types (FR) are mapped to an Ada record type.
- PHIGS pick path item type (PP) is mapped to an Ada record type. PHIGS pick paths are mapped to an Ada array type.
- PHIGS element reference type (ER) is mapped to an Ada record type.
- PHIGS half-space types (HS2, HS3) are mapped to Ada record types.
- PHIGS font/precision pair type (FP) is mapped to an Ada record type.
- PHIGS structure element type (SE) is mapped to an Ada record type.
- PHIGS posted structure type (PS) is mapped to an Ada record type.
- PHIGS bounding range types (B) are mapped to Ada record types.
- PHIGS colour specification type (CLR) is mapped to an Ada record type.
- PHIGS chromaticity coefficient type (CC) is mapped to an Ada record type.
- PHIGS connection identifier type (C) is mapped to the Ada STRING type.
- PHIGS file types (F) are mapped to Ada STRING types.
- PHIGS workstation type (W) is mapped to an Ada integer type.
- PHIGS modelling clipping volume type (MCV) is not used by the binding. Implementations can map this type to an implementation specific private type.
- PHIGS generalized drawing primitive identifier types (G2, G3) are mapped to Ada integer types.
- PHIGS generalized structure element identifier type (GS) is mapped to an Ada integer type.
- PHIGS archive file identifier type (AI) is mapped to an Ada integer type.
- PHIGS pick identifier type (PI) is mapped to an Ada integer type.
- PHIGS escape identifier type (EI) is mapped to an Ada integer type.
- PHIGS function name type (FN) is mapped to an Ada type providing for variable length strings.

- PHIG workstation identifier type (WI) is mapped to an Ada integer type.
- PHIGS tuples are mapped to Ada record types.
- PHIGS matrices are mapped to Ada array types.
- PHIGS arrays are mapped to either an unconstrained Ada array type, or to a record type providing for variable length arrays.
- PHIGS lists are mapped to an Ada private type declared in the generic PHIGS_LIST_UTILITIES package.
- PHIGS set types are mapped to Ada private types with generic accessing functions provided as necessary.
- PHIGS data records are mapped to Ada private types. In some cases, a set of subprograms for operating on the data records is explicitly defined by this binding. This is because the content and structure of the data record is implementation dependent. An implementation of PHIGS may provide other subprograms for manipulating implementation dependent data records.

Additional types used by the binding are declared as needed in a manner compatible with the PHIGS types.

3.2.5 Multi-tasking

The Ada language definition provides explicit support for concurrency. The Ada tasking model includes facilities for declaring and allocating tasks, and operations allowing intertask communication and synchronization.

The PHIGS standard, and hence this binding standard, neither requires nor prohibits an implementation from protecting against problems which could arise from asynchronous access to the PHIGS data structures from concurrent tasks. Implementors of PHIGS should provide information in the user's documentation regarding whether protection against such problems is implemented.

Annex D contains guidelines for implementors who want to support multi-tasking application programs. This annex does not form an integral part of the binding standard, but provides additional information.

3.2.6 Packaging

The PHIGS standard defines all of its graphic functionality as a cohesive whole. An implementation of PHIGS shall implement the entire functionality of PHIGS. To support this concept, this binding defines a single Ada package which corresponds to all of the PHIGS functionality. This package is named

```
package PHIGS is ... end PHIGS;
```

Associated with this package is a data type package which provides the type declarations as defined in 4.2 and the exception defined in 4.3. This package is

```
package PHIGS_TYPES is ... end PHIGS_TYPES;
```

A minimal program referencing PHIGS is shown below.

```
with PHIGS,  
    PHIGS_TYPES;  
  
procedure APPLICATION is  
begin
```

Principles

```
    null;  
end APPLICATION;
```

Several additional Ada packages are defined in this binding. These packages are

- generic package PHIGS_COORDINATE_SYSTEM
- generic package PHIGS_LIST_UTILITIES
- package PHIGS_NAME_SET_FACILITY
- package PHIGS_CONFIGURATION

These packages support the declaration types in the PHIGS_TYPES package described above. PHIGS_COORDINATE_SYSTEM is a generic package which defines an assortment of types supporting each of the PHIGS coordinate systems. PHIGS_LIST_UTILITIES is also a generic package which provides type declarations and operations for list types which correspond to the PHIGS list types. The PHIGS_NAME_SET_FACILITY package defines private types for name sets and provides functions for building and manipulating name sets. The PHIGS_CONFIGURATION package sets implementation dependent limit values for various types defined in this binding. The PHIGS_CONFIGURATION package also provides for possible application modification of these limits.

3.2.7 Application Program Environment

An application program utilizing an Ada implementation of PHIGS will need to be aware of the environment in which both PHIGS and the application program(s) reside.

One aspect of the environment is the Ada program library. The Ada language requires that the application program have access to the program library in which the PHIGS software resides. The Ada standard ISO 8652 does not specify whether there is a single library or multiple libraries, or how access to the libraries is granted or managed. The user's documentation for the PHIGS implementation should specify where the PHIGS library exists in the system, and how access to the library is acquired.

Input/Output interfaces are also implementation-dependent, and are required to be described in the user's documentation. Besides the obvious graphics device interface information, interfaces to the file system shall be included in the documentation. Specifically, this includes the interface to the PHIGS error file, archive files, and metafile storage.

Subclause 3.2.2 discusses implications about the application program environment which may also apply to application programs.

3.2.8 Registration¹

PHIGS reserves certain value ranges for registration as graphical items. The registered graphical items will be bound to Ada (and other programming languages). The registered item bindings will be consistent with the binding presented in this document.

1) For the purpose of this International Standard and according to the rules for the designation and operation of registration authorities in the ISO/IEC JTC 1 procedures, the ISO and IEC Councils have designated the National Institute of Standards and Technology (National Computer Systems Laboratory) A-266 Technology Building, Gaithersburg, MD 20899, USA, to act as registration authority.

4 Tables

4.1 Abbreviations used in procedure names

ASF	aspect source flag
CHAR	character
ESC	escape
GDP	generalized drawing primitive
GSE	generalized structure element
HLHSR	hidden line/hidden surface removal
INQ	inquire
PHIGS	Programmer's Hierarchical Interactive Graphics System
U	unregistered
WS	workstation

4.1.1 List of procedures using the abbreviations

ASF	SET_INDIVIDUAL_ASF
CHAR	SET_ANNOTATION_TEXT_CHAR_HEIGHT SET_ANNOTATION_TEXT_CHAR_UP_VECTOR SET_CHAR_EXPANSION_FACTOR SET_CHAR_HEIGHT SET_CHAR_SPACING SET_CHAR_UP_VECTOR
ESC	PHIGS_ESCAPE.GENERALIZED_ESC PHIGS_ESC <name of the escape procedure>.ESC PHIGS_UESC <name of the escape procedure>.ESC
GDP	INQ_GDP (2D) INQ_GDP (3D) INQ_LIST_OF_AVAILABLE_GDP (2D) INQ_LIST_OF_AVAILABLE_GDP (3D) PHIGS_GDP.GENERALIZED_GDP PHIGS_GDP <name of the GDP procedure>.GDP PHIGS_UGDP <name of the GDP procedure>.GDP
GSE	INQ_GSE_FACILITIES INQ_LIST_OF_AVAILABLE_GSE PHIGS_GSE.GENERALIZED_GSE PHIGS_GSE <name of the GSE procedure>.GSE PHIGS_UGSE <name of the GSE procedure>.GSE
HLHSR	INQ_HLHSR_FACILITIES INQ_HLHSR_MODE SET_HLHSR_IDENTIFIER SET_HLHSR_MODE
INQ	INQ_ALL_CONFLICTING_STRUCTURES

INQ_ANNOTATION FACILITIES
 INQ_ARCHIVE FILES
 INQ_ARCHIVE STATE VALUE
 INQ_CHOICE_DEVICE_STATE (2D)
 INQ_CHOICE_DEVICE_STATE (3D)
 INQ_COLOUR FACILITIES
 INQ_COLOUR MODEL
 INQ_COLOUR MODEL FACILITIES
 INQ_COLOUR REPRESENTATION
 INQ_CONFLICT RESOLUTION
 INQ_CONFLICTING STRUCTURES IN NETWORK
 INQ_CURRENT ELEMENT CONTENT
 INQ_CURRENT ELEMENT TYPE AND SIZE
 INQ_DEFAULT_CHOICE_DEVICE_DATA (2D)
 INQ_DEFAULT_CHOICE_DEVICE_DATA (3D)
 INQ_DEFAULT_DISPLAY_UPDATE STATE
 INQ_DEFAULT_LOCATOR_DEVICE_DATA (2D)
 INQ_DEFAULT_LOCATOR_DEVICE_DATA (3D)
 INQ_DEFAULT_PICK_DEVICE_DATA (2D)
 INQ_DEFAULT_PICK_DEVICE_DATA (3D)
 INQ_DEFAULT_STRING_DEVICE_DATA (2D)
 INQ_DEFAULT_STRING_DEVICE_DATA (3D)
 INQ_DEFAULT_STROKE_DEVICE_DATA (2D)
 INQ_DEFAULT_STROKE_DEVICE_DATA (3D)
 INQ_DEFAULT_VALUATOR_DEVICE_DATA (2D)
 INQ_DEFAULT_VALUATOR_DEVICE_DATA (3D)
 INQ_DISPLAY_SPACE_SIZE (2D)
 INQ_DISPLAY_SPACE_SIZE (3D)
 INQ_DISPLAY_UPDATE STATE
 INQ_DYNAMICS OF STRUCTURES
 INQ_DYNAMICS OF WS ATTRIBUTES
 INQ_EDGE FACILITIES
 INQ_EDGE REPRESENTATION
 INQ_EDIT MODE
 INQ_ELEMENT CONTENT
 INQ_ELEMENT POINTER
 INQ_ELEMENT TYPE AND SIZE
 INQ_ERROR HANDLING MODE
 INQ_GDP (2D)
 INQ_GDP (3D)
 INQ_GSE FACILITIES
 INQ_HIGHLIGHTING FILTER
 INQ_HLHSR FACILITIES
 INQ_HLHSR MODE
 INQ_INPUT_QUEUE_OVERFLOW
 INQ_INTERIOR FACILITIES
 INQ_INTERIOR REPRESENTATION
 INQ_INVISIBILITY FILTER
 INQ_LIST_OF_AVAILABLE_GDP (2D)
 INQ_LIST_OF_AVAILABLE_GDP (3D)
 INQ_LIST_OF_AVAILABLE_GSE
 INQ_LIST_OF_AVAILABLE_WS TYPES
 INQ_LIST_OF COLOUR INDICES
 INQ_LIST_OF_EDGE INDICES
 INQ_LIST_OF_INTERIOR INDICES
 INQ_LIST_OF_PATTERN INDICES
 INQ_LIST_OF_POLYLINE INDICES
 INQ_LIST_OF_POLYMARKER INDICES
 INQ_LIST_OF TEXT INDICES
 INQ_LIST_OF VIEW INDICES
 INQ_LOCATOR_DEVICE_STATE (2D)
 INQ_LOCATOR_DEVICE_STATE (3D)
 INQ_MODELLING CLIPPING FACILITIES
 INQ_MORE SIMULTANEOUS EVENTS
 INQ_NUMBER_OF_AVAILABLE LOGICAL INPUT DEVICES
 INQ_NUMBER_OF_DISPLAY PRIORITIES SUPPORTED
 INQ_OPEN STRUCTURE
 INQ_PATHS TO ANCESTORS
 INQ_PATHS TO DESCENDANTS
 INQ_PATTERN FACILITIES
 INQ_PATTERN REPRESENTATION

	INQ_PHIGS_FACILITIES INQ_PICK_DEVICE_STATE (2D) INQ_PICK_DEVICE_STATE (3D) INQ_POLYLINE_FACILITIES INQ_POLYLINE_REPRESENTATION INQ_POLYMARKER_FACILITIES INQ_POLYMARKER_REPRESENTATION INQ_POSTED_STRUCTURES INQ_PREDEFINED_COLOUR_REPRESENTATION INQ_PREDEFINED_EDGE_REPRESENTATION INQ_PREDEFINED_INTERIOR_REPRESENTATION INQ_PREDEFINED_PATTERN_REPRESENTATION INQ_PREDEFINED_POLYLINE_REPRESENTATION INQ_PREDEFINED_POLYMARKER_REPRESENTATION INQ_PREDEFINED_TEXT_REPRESENTATION INQ_PREDEFINED_VIEW_REPRESENTATION INQ_SET_OF_OPEN_WS INQ_SET_OF_WS_TO_WHICH_POSTED INQ_STRING_DEVICE_STATE (2D) INQ_STRING_DEVICE_STATE (3D) INQ_STROKE_DEVICE_STATE (2D) INQ_STROKE_DEVICE_STATE (3D) INQ_STRUCTURE_IDENTIFIERS INQ_STRUCTURE_STATE_VALUE INQ_STRUCTURE_STATUS INQ_SYSTEM_STATE_VALUE INQ_TEXT_EXTENT (text) INQ_TEXT_EXTENT (annotation text) INQ_TEXT_FACILITIES INQ_TEXT_REPRESENTATION INQ_VALUATOR_DEVICE_STATE (2D) INQ_VALUATOR_DEVICE_STATE (3D) INQ_VIEW_FACILITIES INQ_VIEW_REPRESENTATION INQ_WS_CATEGORY INQ_WS_CLASSIFICATION INQ_WS_CONNECTION_AND_TYPE INQ_WS_STATE_TABLE_LENGTHS INQ_WS_STATE_VALUE INQ_WS_TRANSFORMATION (2D) INQ_WS_TRANSFORMATION (3D)
PHIGS	CLOSE_PHIGS EMERGENCY_CLOSE_PHIGS INQ_PHIGS_FACILITIES OPEN_PHIGS
U	PHIGS_UESC <name of the escape procedure>.ESC PHIGS_UGDP <name of the GDP procedure>.GDP PHIGS_UGSE <name of the GSE procedure>.GSE
WS	CLOSE_WS INQ_DYNAMICS_OF_WS_ATTRIBUTES INQ_LIST_OF_AVAILABLE_WS_TYPES INQ_SET_OF_OPEN_WS INQ_SET_OF_WS_TO_WHICH_POSTED INQ_WS_CATEGORY INQ_WS_CLASSIFICATION INQ_WS_CONNECTION_AND_TYPE INQ_WS_STATE_TABLE_LENGTHS INQ_WS_STATE_VALUE INQ_WS_TRANSFORMATION (2D) INQ_WS_TRANSFORMATION (3D) OPEN_WS SET_WS_VIEWPORT (2D) SET_WS_VIEWPORT (3D) SET_WS_WINDOW (2D) SET_WS_WINDOW (3D) UPDATE_WS

4.1.2 Alphabetical by bound name

This subclause alphabetically lists the Ada procedures corresponding to each PHIGS function. Instances in which the binding of a name corresponds to more than one PHIGS function indicates the use of Ada overloading.

ADD NAMES TO SET	add names to set
ANNOTATION TEXT RELATIVE	annotation text relative
ANNOTATION TEXT RELATIVE	annotation text relative 3
APPLICATION DATA	application data
ARCHIVE ALL STRUCTURES	archive all structures
ARCHIVE STRUCTURE NETWORKS	archive structure networks
ARCHIVE STRUCTURES	archive structures
AWAIT EVENT	await event
BUILD TRANSFORMATION MATRIX	build transformation matrix
BUILD TRANSFORMATION MATRIX	build transformation matrix 3
CELL ARRAY	cell array
CELL ARRAY	cell array 3
CHANGE STRUCTURE IDENTIFIER	change structure identifier
CHANGE STRUCTURE IDENTIFIER AND REFERENCES	change structure identifier and references
CHANGE STRUCTURE REFERENCES	change structure references
CLOSE ARCHIVE FILE	close archive file
CLOSE PHIGS	close PHIGS
CLOSE STRUCTURE	close structure
CLOSE WS	close workstation
COMPOSE MATRIX	compose matrix
COMPOSE MATRIX	compose matrix 3
COMPOSE TRANSFORMATION MATRIX	compose transformation matrix
COMPOSE TRANSFORMATION MATRIX	compose transformation matrix 3
COPY ALL ELEMENTS FROM STRUCTURE	copy all elements from structure
DELETE ALL STRUCTURES	delete all structures
DELETE ALL STRUCTURES FROM ARCHIVE	delete all structures from archive
DELETE ELEMENT	delete element
DELETE ELEMENT RANGE	delete element range
DELETE ELEMENTS BETWEEN LABELS	delete elements between labels
DELETE STRUCTURE	delete structure
DELETE STRUCTURE NETWORK	delete structure network
DELETE STRUCTURE NETWORKS FROM ARCHIVE	delete structure networks from archive
DELETE STRUCTURES FROM ARCHIVE	delete structures from archive
ELEMENT SEARCH	element search
EMERGENCY CLOSE PHIGS	emergency close PHIGS
EMPTY STRUCTURE	empty structure
ERROR HANDLING	error handling
ERROR LOGGING	error logging
ESCAPE	escape
EVALUATE VIEW MAPPING MATRIX	evaluate view mapping matrix
EVALUATE VIEW MAPPING MATRIX	evaluate view mapping matrix 3
EVALUATE VIEW ORIENTATION MATRIX	evaluate view orientation matrix
EVALUATE VIEW ORIENTATION MATRIX	evaluate view orientation matrix 3
EXECUTE STRUCTURE	execute structure
FILL AREA	fill area
FILL AREA	fill area 3
FILL AREA SET	fill area set
FILL AREA SET	fill area set 3
FLUSH DEVICE EVENTS	flush device events
GENERALIZED GDP	generalized drawing primitive
GENERALIZED GDP	generalized drawing primitive 3
GENERALIZED GSE	generalized structure element
GET CHOICE	get choice
GET ITEM TYPE FROM METAFILE	get item type from metafile
GET LOCATOR	get locator
GET LOCATOR	get locator 3
GET PICK	get pick
GET STRING	get string
GET STROKE	get stroke

GET_STROKE	get stroke 3
GET_VALUATOR	get valuator
INCREMENTAL_SPATIAL_SEARCH	incremental spatial search
INCREMENTAL_SPATIAL_SEARCH	incremental spatial search 3
INITIALIZE_CHOICE	initialize choice
INITIALIZE_CHOICE	initialize choice 3
INITIALIZE_LOCATOR	initialize locator
INITIALIZE_LOCATOR	initialize locator 3
INITIALIZE_PICK	initialize pick
INITIALIZE_PICK	initialize pick 3
INITIALIZE_STRING	initialize string
INITIALIZE_STRING	initialize string 3
INITIALIZE_STROKE	initialize stroke
INITIALIZE_STROKE	initialize stroke 3
INITIALIZE_VALUATOR	initialize valuator
INITIALIZE_VALUATOR	initialize valuator 3
INQ_ALL_CONFLICTING_STRUCTURES	inquire all conflicting structures
INQ_ANNOTATION_FACILITIES	inquire annotation facilities
INQ_ARCHIVE_FILES	inquire archive files
INQ_ARCHIVE_STATE_VALUE	inquire archive state value
INQ_CHOICE_DEVICE_STATE	inquire choice device state
INQ_CHOICE_DEVICE_STATE	inquire choice device state 3
INQ_COLOUR_FACILITIES	inquire colour facilities
INQ_COLOUR_MODEL	inquire colour model
INQ_COLOUR_MODEL_FACILITIES	inquire colour model facilities
INQ_COLOUR_REPRESENTATION	inquire colour representation
INQ_CONFLICT_RESOLUTION	inquire conflict resolution
INQ_CONFLICTING_STRUCTURES_IN_NETWORK	inquire conflicting structures in network
INQ_CURRENT_ELEMENT_CONTENT	inquire current element content
INQ_CURRENT_ELEMENT_TYPE_AND_SIZE	inquire current element type and size
INQ_DEFAULT_CHOICE_DEVICE_DATA	inquire default choice device data
INQ_DEFAULT_CHOICE_DEVICE_DATA	inquire default choice device data 3
INQ_DEFAULT_DISPLAY_UPDATE_STATE	inquire default display update state
INQ_DEFAULT_LOCATOR_DEVICE_DATA	inquire default locator device data
INQ_DEFAULT_LOCATOR_DEVICE_DATA	inquire default locator device data 3
INQ_DEFAULT_PICK_DEVICE_DATA	inquire default pick device data
INQ_DEFAULT_PICK_DEVICE_DATA	inquire default pick device data 3
INQ_DEFAULT_STRING_DEVICE_DATA	inquire default string device data
INQ_DEFAULT_STRING_DEVICE_DATA	inquire default string device data 3
INQ_DEFAULT_STROKE_DEVICE_DATA	inquire default stroke device data
INQ_DEFAULT_STROKE_DEVICE_DATA	inquire default stroke device data 3
INQ_DEFAULT_VALUATOR_DEVICE_DATA	inquire default valuator device data
INQ_DEFAULT_VALUATOR_DEVICE_DATA	inquire default valuator device data 3
INQ_DISPLAY_SPACE_SIZE	inquire display space size
INQ_DISPLAY_SPACE_SIZE	inquire display space size 3
INQ_DISPLAY_UPDATE_STATE	inquire display update state
INQ_DYNAMICS_OF_STRUCTURES	inquire dynamics of structures
INQ_DYNAMICS_OF_WS_ATTRIBUTES	inquire dynamics of workstation attributes
INQ_EDGE_FACILITIES	inquire edge facilities
INQ_EDGE_REPRESENTATION	inquire edge representation
INQ_EDIT_MODE	inquire edit mode
INQ_ELEMENT_CONTENT	inquire element content
INQ_ELEMENT_POINTER	inquire element pointer
INQ_ELEMENT_TYPE_AND_SIZE	inquire element type and size
INQ_ERROR_HANDLING_MODE	inquire error handling mode
INQ_GDP	inquire generalized drawing primitives
INQ_GDP	inquire generalized drawing primitives 3
INQ_GSE_FACILITIES	inquire generalized structure element facilities
INQ_HIGHLIGHTING_FILTER	inquire highlighting filter
INQ_HLHSR_FACILITIES	inquire hlhsr facilities
INQ_HLHSR_MODE	inquire hlhsr mode
INQ_INPUT_QUEUE_OVERFLOW	inquire input queue overflow
INQ_INTERIOR_FACILITIES	inquire interior facilities
INQ_INTERIOR_REPRESENTATION	inquire interior representation
INQ_INVISIBILITY_FILTER	inquire invisibility filter
INQ_LIST_OF_AVAILABLE_GDP	inquire list of available generalized drawing primitives
INQ_LIST_OF_AVAILABLE_GDP	inquire list of available generalized drawing primitives 3
INQ_LIST_OF_AVAILABLE_GSE	inquire list of available generalized structure elements
INQ_LIST_OF_AVAILABLE_WS_TYPES	inquire list of available workstation types
INQ_LIST_OF_COLOUR_INDICES	inquire list of colour indices
INQ_LIST_OF_EDGE_INDICES	inquire list of edge indices

INQ LIST OF INTERIOR INDICES	inquire list of interior indices
INQ LIST OF PATTERN INDICES	inquire list of pattern indices
INQ LIST OF POLYLINE INDICES	inquire list of polyline indices
INQ LIST OF POLYMARKER INDICES	inquire list of polymarker indices
INQ LIST OF TEXT INDICES	inquire list of text indices
INQ LIST OF VIEW INDICES	inquire list of view indices
INQ LOCATOR DEVICE STATE	inquire locator device state
INQ LOCATOR DEVICE STATE 3	inquire locator device state 3
INQ MODELLING CLIPPING FACILITIES	inquire modelling clipping facilities
INQ MORE SIMULTANEOUS EVENTS	inquire more simultaneous events
INQ NUMBER OF AVAILABLE LOGICAL INPUT DEVICES	inquire number of available logical input devices
INQ NUMBER OF DISPLAY PRIORITIES SUPPORTED	inquire number of structure priorities supported
INQ OPEN STRUCTURE	inquire open structure
INQ PATHS TO ANCESTORS	inquire paths to ancestors
INQ PATHS TO DESCENDANTS	inquire paths to descendants
INQ PATTERN FACILITIES	inquire pattern facilities
INQ PATTERN REPRESENTATION	inquire pattern representation
INQ PHIGS FACILITIES	inquire phigs facilities
INQ PICK DEVICE STATE	inquire pick device state
INQ PICK DEVICE STATE 3	inquire pick device state 3
INQ POLYLINE FACILITIES	inquire polyline facilities
INQ POLYLINE REPRESENTATION	inquire polyline representation
INQ POLYMARKER FACILITIES	inquire polymarker facilities
INQ POLYMARKER REPRESENTATION	inquire polymarker representation
INQ POSTED STRUCTURES	inquire posted structures
INQ PREDEFINED COLOUR REPRESENTATION	inquire predefined colour representation
INQ PREDEFINED EDGE REPRESENTATION	inquire predefined edge representation
INQ PREDEFINED INTERIOR REPRESENTATION	inquire predefined interior representation
INQ PREDEFINED PATTERN REPRESENTATION	inquire predefined pattern representation
INQ PREDEFINED POLYLINE REPRESENTATION	inquire predefined polyline representation
INQ PREDEFINED POLYMARKER REPRESENTATION	inquire predefined polymarker representation
INQ PREDEFINED TEXT REPRESENTATION	inquire predefined text representation
INQ PREDEFINED VIEW REPRESENTATION	inquire predefined view representation
INQ SET OF OPEN WS	inquire set of open workstations
INQ SET OF WS TO WHICH POSTED	inquire set of workstations to which posted
INQ STRING DEVICE STATE	inquire string device state
INQ STRING DEVICE STATE 3	inquire string device state 3
INQ STROKE DEVICE STATE	inquire stroke device state
INQ STROKE DEVICE STATE 3	inquire stroke device state 3
INQ STRUCTURE IDENTIFIERS	inquire structure identifiers
INQ STRUCTURE STATE VALUE	inquire structure state value
INQ STRUCTURE STATUS	inquire structure status
INQ SYSTEM STATE VALUE	inquire system state value
INQ TEXT EXTENT (text)	inquire text extent
INQ TEXT EXTENT (annotation text)	inquire text extent
INQ TEXT FACILITIES	inquire text facilities
INQ TEXT REPRESENTATION	inquire text representation
INQ VALUATOR DEVICE STATE	inquire valuator device state
INQ VALUATOR DEVICE STATE 3	inquire valuator device state 3
INQ VIEW FACILITIES	inquire view facilities
INQ VIEW REPRESENTATION	inquire view representation
INQ WS CATEGORY	inquire workstation category
INQ WS CLASSIFICATION	inquire workstation classification
INQ WS CONNECTION AND TYPE	inquire workstation connection and type
INQ WS STATE TABLE LENGTHS	inquire workstation state table lengths
INQ WS STATE VALUE	inquire workstation state value
INQ WS TRANSFORMATION	inquire workstation transformation
INQ WS TRANSFORMATION 3	inquire workstation transformation 3
INTERPRET_ITEM	interpret item
LABEL	label
MESSAGE	message
OFFSET_ELEMENT_POINTER	offset element pointer
OPEN_ARCHIVE_FILE	open archive fill
OPEN_PHIGS	open PHIGS
OPEN_STRUCTURE	open structure
OPEN_WS	open workstation
POLYLINE	polyline
POLYLINE 3	polyline 3
POLYMARKER	polymarker
POLYMARKER 3	polymarker 3
POST_STRUCTURE	post structure

READ ITEM FROM METAFILE	read item from metafile
REDRAW ALL STRUCTURES	redraw all structures
REMOVE NAMES FROM SET	remove names from set
REQUEST CHOICE	request choice
REQUEST LOCATOR	request locator
REQUEST LOCATOR 3	request locator 3
REQUEST PICK	request pick
REQUEST STRING	request string
REQUEST STROKE	request stroke
REQUEST STROKE 3	request stroke 3
REQUEST VALUATOR	request valuator
RESTORE MODELLING CLIPPING VOLUME	restore modelling clipping volume
RETRIEVE ALL STRUCTURES	retrieve all structures
RETRIEVE PATHS TO ANCESTORS	retrieve paths to ancestors
RETRIEVE PATHS TO DESCENDANTS	retrieve paths to descendants
RETRIEVE STRUCTURE IDENTIFIERS	retrieve structure identifiers
RETRIEVE STRUCTURE NETWORKS	retrieve structure networks
RETRIEVE STRUCTURES	retrieve structures
ROTATE	rotate
ROTATE X	rotate x
ROTATE Y	rotate y
ROTATE Z	rotate z
SAMPLE CHOICE	sample choice
SAMPLE LOCATOR	sample locator
SAMPLE LOCATOR 3	sample locator 3
SAMPLE PICK	sample pick
SAMPLE STRING	sample string
SAMPLE STROKE	sample stroke
SAMPLE STROKE 3	sample stroke 3
SAMPLE VALUATOR	sample valuator
SCALE	scale
SCALE 3	scale 3
SET ANNOTATION STYLE	set annotation style
SET ANNOTATION TEXT ALIGNMENT	set annotation text alignment
SET ANNOTATION TEXT CHAR HEIGHT	set annotation text character height
SET ANNOTATION TEXT CHAR UP VECTOR	set annotation text character up vector
SET ANNOTATION TEXT PATH	set annotation text path
SET CHAR EXPANSION FACTOR	set character expansion factor
SET CHAR HEIGHT	set character height
SET CHAR SPACING	set character spacing
SET CHAR UP VECTOR	set character up vector
SET CHOICE MODE	set choice mode
SET COLOUR MODEL	set colour model
SET COLOUR REPRESENTATION	set colour representation
SET CONFLICT RESOLUTION	set conflict resolution
SET DISPLAY UPDATE STATE	set display update state
SET EDGE COLOUR INDEX	set edge colour index
SET EDGE FLAG	set edge flag
SET EDGE INDEX	set edge index
SET EDGE REPRESENTATION	set edge representation
SET EDGETYPE	set edgetype
SET EDGEWIDTH SCALE FACTOR	set edgewidth scale factor
SET EDIT MODE	set edit mode
SET ELEMENT POINTER	set element pointer
SET ELEMENT POINTER AT LABEL	set element pointer at label
SET ERROR HANDLING MODE	set error handling mode
SET GLOBAL TRANSFORMATION	set global transformation
SET GLOBAL TRANSFORMATION 3	set global transformation 3
SET HIGHLIGHTING FILTER	set highlighting filter
SET HLHSR IDENTIFIER	set hlhsr identifier
SET HLHSR MODE	set hlhsr mode
SET INDIVIDUAL ASF	set individual asf
SET INTERIOR COLOUR INDEX	set interior colour index
SET INTERIOR INDEX	set interior index
SET INTERIOR REPRESENTATION	set interior representation
SET INTERIOR STYLE	set interior style
SET INTERIOR STYLE INDEX	set interior style index
SET INVISIBILITY FILTER	set invisibility filter
SET LINETYPE	set linetype
SET LINEWIDTH SCALE FACTOR	set linewidth scale factor
SET LOCAL TRANSFORMATION	set local transformation

Tables

SET LOCAL TRANSFORMATION	set local transformation 3
SET LOCATOR MODE	set locator mode
SET MARKER SIZE SCALE_FACTOR	set marker size scale factor
SET MARKER TYPE	set marker type
SET MODELLING CLIPPING_INDICATOR	set modelling clipping indicator
SET MODELLING CLIPPING_VOLUME	set modelling clipping volume
SET MODELLING CLIPPING_VOLUME	set modelling clipping volume 3
SET PATTERN REFERENCE_POINT	set pattern reference point
SET PATTERN REFERENCE_POINT_AND_VECTORS	set pattern reference point and vectors
SET PATTERN REPRESENTATION	set pattern representation
SET PATTERN SIZE	set pattern size
SET PICK FILTER	set pick filter
SET PICK IDENTIFIER	set pick identifier
SET PICK MODE	set pick mode
SET POLYLINE COLOUR_INDEX	set polyline colour index
SET POLYLINE INDEX	set polyline index
SET POLYLINE REPRESENTATION	set polyline representation
SET POLYMARKER COLOUR_INDEX	set polymarker colour index
SET POLYMARKER INDEX	set polymarker index
SET POLYMARKER REPRESENTATION	set polymarker representation
SET STRING MODE	set string mode
SET STROKE MODE	set stroke mode
SET TEXT ALIGNMENT	set text alignment
SET TEXT COLOUR_INDEX	set text colour index
SET TEXT FONT	set text font
SET TEXT INDEX	set text index
SET TEXT PATH	set text path
SET TEXT PRECISION	set text precision
SET TEXT REPRESENTATION	set text representation
SET VALUATOR MODE	set valuator mode
SET VIEW INDEX	set view index
SET VIEW REPRESENTATION	set view representation
SET VIEW REPRESENTATION	set view representation 3
SET VIEW TRANSFORMATION_INPUT_PRIORITY	set view transformation input priority
SET WS VIEWPORT	set workstation viewport
SET WS VIEWPORT	set workstation viewport 3
SET WS WINDOW	set workstation window
SET WS WINDOW	set workstation window 3
TEXT	text
TEXT	text 3
TRANSFORM POINT	transform point
TRANSFORM POINT	transform point 3
TRANSLATE	translate
TRANSLATE	translate 3
UNPOST ALL STRUCTURES	unpost all structures
UNPOST STRUCTURE	unpost structure
UPDATE WS	update workstation
WRITE_ITEM_TO_METAFILE	write item to metafile

4.1.3 Alphabetical PHIGS functions

The functions are in the same order when listed alphabetically according to the PHIGS function names as they are when listed alphabetically according to the bound names. Therefore, the table provided in 4.1.2 alphabetically lists the PHIGS functions.

4.2 Data type definitions

4.2.1 Abbreviations used in the data type definitions

ARCL	archive closed
AROP	archive open
ASAP	as soon as possible
ASF	aspect source flag
ASTI	at some time
BNIG	before next interaction globally
BNIL	before next interaction locally
CBS	can be simulated
CHAR	character
CSS	central structure store
DC	device coordinate
GDP	generalized drawing primitive
GSE	generalized structure element
HLHSR	hidden line/hidden surface removal
ID	identifier
IDS	identifiers
IMM	immediate
IND	index
IRG	implicit regeneration
MAX	maximum
MC	modelling coordinates
MI	metafile input
MIN	minimum
MISC	miscellaneous
MO	metafile output
NIVE	no immediate visual effect
NPC	normalized projection coordinates
PHCL	PHIGS closed
PHIGS	Programmer's Hierarchical Interactive Graphics System
PHOP	PHIGS open
PT	point
REF	reference
SF	scale factor
STCL	structure closed
STOP	structure open
UQUM	use quick update method
UWOR	update without regeneration
VC	viewing coordinates
WC	world coordinates
WS	workstation
WSCL	workstation closed
WSOP	workstation open

4.2.2 Alphabetical list of type definitions

This subclause contains an alphabetical listing of all of the data type definitions used to define the Ada binding to PHIGS. Each element declared includes a comment about the type and/or use of the type. Some of the declarations employ constant values in the definition of the type. All of these constant declarations are included in the PHIGS_TYPES package.

ACCESS_COLOUR_MATRIX

type ACCESS_COLOUR_MATRIX is access COLOUR_MATRIX;

- Provides for returning variable sized matrices containing colour
 - indices corresponding to a cell array or pattern array.
-

ACCESS_STRING

type ACCESS_STRING is access PHIGS_STRING;

- Defines a variable length string needed to return strings in
 - structure element records.
-

ANGLE

type ANGLE is digits PHIGS_PRECISION;

- Values used in the modelling transformation utility functions
 - for specifying angles of rotation in radians. Positive
 - indicates a counterclockwise direction.
-

ANNOTATION_STYLE

type ANNOTATION_STYLE is new PHIGS_INTEGER;

- Defines the annotation styles for annotation primitives.
-

ANNOTATION_STYLES

package ANNOTATION_STYLES is
 new PHIGS_LIST_UTILITIES (ANNOTATION_STYLE,
 MAX_ANNOTATION_STYLES_SUPPORTED);

- Provides for lists of annotation styles.
-

APPLICATION_DATA_RANGE

subtype APPLICATION_DATA_RANGE is
 PHIGS_NATURAL range 0..MAX_APPLICATION_DATA;

- Defines range of lengths for application data objects.

APPLICATION_DATA_RECORD

```

type APPLICATION_DATA_RECORD (LENGTH : APPLICATION_DATA_RANGE := 0) is
    record
        DATA : PHIGS_STRING (1..LENGTH);
    end record;

```

```

-- Defines type for defining application data objects.

```

ARCHIVE_ID

```

type ARCHIVE_ID is new PHIGS_NATURAL;

```

```

-- Provides for an application specified pointer to archive
-- files separate from the file identifier.

```

ARCHIVE_IDS

```

package ARCHIVE_IDS is new PHIGS_LIST_UTILITIES (ARCHIVE_ID,
                                                    MAX_ARCHIVE_IDS_SUPPORTED);

```

```

-- Provides for lists of archive identifiers.

```

ARCHIVE_STATE

```

type ARCHIVE_STATE is (ARCL,
                       AROP);

```

```

-- The type used to return the archive state.

```

ASF

```

type ASF is (BUNDLED,
             INDIVIDUAL);

```

```

-- This type defines an aspect source flag whose
-- value indicates whether an aspect of a primitive
-- should be set from a bundle table or from an
-- individual attribute.

```

ASPECT

```

type ASPECT is (TYPE_OF_LINE,
                LINEWIDTH_SF,
                LINE_COLOUR,

```

```

TYPE_OF_MARKER,
SIZE,
MARKER_COLOUR,

FONT,
PRECISION,
EXPANSION,
SPACING,
TEXT_COLOUR,

STYLE_OF_INTERIOR,
STYLE_IND,
INTERIOR_COLOUR,

FLAG,
TYPE_OF_EDGE,
EDGEWIDTH_SF,
EDGE_COLOUR);

```

- This type lists the aspects for which an aspect source
- flag exists.

ATTRIBUTES_USED

```

package ATTRIBUTES_USED is
    new PHIGS_LIST_UTILITIES
        (ATTRIBUTES_USED_TYPE,
         ATTRIBUTES_USED_TYPE'POS(ATTRIBUTES_USED_TYPE'LAST)+1);

```

- Provides for a list of the attributes used which is
- returned by the INQ_GDP and LOCATOR_ATTRIBUTES_USED
- functions.

ATTRIBUTES_USED_TYPE

```

type ATTRIBUTES_USED_TYPE is (POLYLINE_ATTRIBUTES,
                               POLYMARKER_ATTRIBUTES,
                               TEXT_ATTRIBUTES,
                               INTERIOR_ATTRIBUTES,
                               EDGE_ATTRIBUTES);

```

- The types of attributes which may be used in generating
- output for a GDP and in generating prompt and echo
- information for certain prompt and echo types of certain
- classes of input devices.

CHAR_EXPANSION

```

type CHAR_EXPANSION is new SCALE_FACTOR range
    SCALE_FACTOR'SAFE_SMALL..SCALE_FACTOR'LAST;

```

- Defines a character expansion factor.

CHAR_SET

type CHAR_SET is new PHIGS_NATURAL;

- Provides identifications for the available character
 - sets. ISO 646 character set maps to value zero.
-

CHAR_SETS

package CHAR_SETS is new PHIGS_LIST_UTILITIES (CHAR_SET,
MAX_CHAR_SETS_SUPPORTED);

- Provides for lists of character set identifications.
-

CHAR_SPACING

type CHAR_SPACING is new SCALE_FACTOR;

- Defines a character spacing factor. The factors are
 - unitless. A positive value indicates the amount of extra
 - space between character boxes in a text string, and a
 - negative value indicates the amount of overlap between
 - character boxes in a text string.
-

CHOICE_DEVICE_NUMBER

type CHOICE_DEVICE_NUMBER is new DEVICE_NUMBER;

- Provides for choice device identifiers.
-

CHOICE_NUMBER

type CHOICE_NUMBER is new PHIGS_POSITIVE;

- Defines the choice numbers available on an implementation.
-

CHOICE_PROMPT

type CHOICE_PROMPT is new OFF_ON;

- Indicates whether a specified choice prompt is to be displayed
- or not.

CHOICE_PROMPTS

```
package CHOICE_PROMPTS is
  new PHIGS_LIST_UTILITIES
  (CHOICE_PROMPT,
   MAX_CHOICE_PROMPTS_SUPPORTED);
```

-- Provides for lists of choice prompts.

CHOICE_PROMPT_ECHO_TYPE

```
type CHOICE_PROMPT_ECHO_TYPE is new PHIGS_INTEGER;
```

-- Defines the choice prompt and echo type.

CHOICE_PROMPT_ECHO_TYPES

```
package CHOICE_PROMPT_ECHO_TYPES is
  new PHIGS_LIST_UTILITIES
  (CHOICE_PROMPT_ECHO_TYPE,
   MAX_CHOICE_PROMPT_ECHO_TYPES_SUPPORTED);
```

-- Provides for lists of choice prompt and echo types.

CHOICE_PROMPT_STRING

```
type CHOICE_PROMPT_STRING (LENGTH : STRING_SMALL_NATURAL := 0) is
  record
    CONTENTS : PHIGS_STRING (1..LENGTH);
  end record;
```

-- Provides for a variable length prompt. Objects of this
-- type should be declared unconstrained to allow for dynamic
-- modification of the length.

CHOICE_PROMPT_STRING_ARRAY

```
type CHOICE_PROMPT_STRING_ARRAY is array (PHIGS_POSITIVE range <>) of
  CHOICE_PROMPT_STRING;
```

-- Provides for an array of prompt strings.

CHOICE_PROMPT_STRING_LIST

type CHOICE_PROMPT_STRING_LIST (LENGTH : CHOICE_SMALL_NATURAL := 0) is
 record
 LIST : CHOICE_PROMPT_STRING_ARRAY (1..LENGTH);
 end record;

-- Provides for lists of prompt strings.

CHOICE_REQUEST_STATUS

type CHOICE_REQUEST_STATUS is (OK,
 NOCHOICE,
 NONE);

-- Defines the status of a choice input operation for the
-- request function.

CHOICE_SMALL_NATURAL

subtype CHOICE_SMALL_NATURAL is
 PHIGS_NATURAL range 0..MAX_CHOICE_SMALL_NATURAL;

-- This is a subtype declaration which allows for
-- unconstrained record objects for CHOICE_PROMPT_STRING_LIST
-- type without causing the exception STORAGE_ERROR to be raised.

CHOICE_STATUS

subtype CHOICE_STATUS is
 CHOICE_REQUEST_STATUS range OK..NOCHOICE;

-- Indicates if a choice was made by the operator for the
-- sample, get, and inquiry functions.

CHROMATICITY_COEFFICIENT

type CHROMATICITY_COEFFICIENT is
 record
 X : COLOUR_COEFFICIENT;
 Y : COLOUR_COEFFICIENT;
 end record;

-- Defines a CIE colour coefficient pair.

CHROMATICITY_COEFFICIENT_SET

type CHROMATICITY_COEFFICIENT_SET is
 record
 R : CHROMATICITY_COEFFICIENT;
 G : CHROMATICITY_COEFFICIENT;
 B : CHROMATICITY_COEFFICIENT;
 end record;

-- Defines a set of CIE primary colour chromaticity coefficients.

CLIPPING_INDICATOR

type CLIPPING_INDICATOR is (CLIP,
 NOCLIP);

-- Indicates whether or not clipping is to be performed.

COLOUR_AVAILABLE

type COLOUR_AVAILABLE is (COLOUR,
 MONOCHROME);

-- Indicates whether colour output is available on a
-- workstation.

COLOUR_COEFFICIENT

type COLOUR_COEFFICIENT is digits PHIGS_PRECISION;

-- Defines a general type for colour components.

COLOUR_COEFFICIENT_ARRAY

type COLOUR_COEFFICIENT_ARRAY is
 array (COLOUR_COMPONENTS) of COLOUR_COEFFICIENT;

-- Defines a general type for colour components.

COLOUR_COEFFICIENTS

package COLOUR_COEFFICIENTS is
 new PHIGS_LIST_UTILITIES (COLOUR_COEFFICIENT,
 MAX_COLOUR_COEFFICIENTS);

-- Provides for a lists of colour coefficients for the generalized
-- colour model.

COLOUR_COMPONENTS

type COLOUR_COMPONENTS is range 1..MAX_COLOUR_COEFFICIENTS;

-- Defines a type for the size of colour component arrays.

COLOUR_INDEX

type COLOUR_INDEX is new PHIGS_NATURAL;

-- Indices into colour tables are of this type.

COLOUR_INDICES

package COLOUR_INDICES is
 new PHIGS_LIST_UTILITIES (COLOUR_INDEX,
 MAX_COLOUR_INDICES_SUPPORTED);

-- Provides for a set of colour indices which are available
 -- on a particular workstation.

COLOUR_MATRIX

type COLOUR_MATRIX is array (COLOUR_MATRIX_SMALL_NATURAL range < >,
 COLOUR_MATRIX_SMALL_NATURAL range < >)
 of COLOUR_INDEX;

-- Provides for matrices containing colour indices corresponding
 -- to a cell array or pattern array.

COLOUR_MATRIX_SMALL_NATURAL

subtype COLOUR_MATRIX_SMALL_NATURAL is
 PHIGS_NATURAL range 0..MAX_COLOUR_MATRIX_SMALL_NATURAL;

-- This is a subtype declaration which allows for unconstrained
 -- record objects for COLOUR_MATRIX type without causing the
 -- exception STORAGE_ERROR to be raised.

COLOUR_MODEL

type COLOUR_MODEL is new PHIGS_INTEGER;

-- Indicates the colour models available in PHIGS.

COLOUR_MODELS

```

package COLOUR_MODELS is
    new PHIGS_LIST_UTILITIES (COLOUR_MODEL,
                              MAX_COLOUR_MODELS_SUPPORTED);

-- Provides for lists of colour models.

```

COLOUR_REPRESENTATION

```

type COLOUR_REPRESENTATION (MODEL : COLOUR_MODEL := RGB) is
    record
        case MODEL is

```

```

            when RGB = >
                RED_RGB      : INTENSITY;
                GREEN_RGB    : INTENSITY;
                BLUE_RGB     : INTENSITY;

```

```

            when CIELUV = >
                L_STAR_CIE   : COLOUR_COEFFICIENT;
                U_STAR_CIE   : COLOUR_COEFFICIENT;
                V_STAR_CIE   : COLOUR_COEFFICIENT;

```

```

            when HSV = >
                HUE_HSV      : INTENSITY;
                SATURATION_HSV : INTENSITY;
                VALUE_HSV    : INTENSITY;

```

```

            when HLS = >
                HUE_HLS      : INTENSITY;
                LIGHTNESS_HLS : INTENSITY;
                SATURATION_HLS : INTENSITY;

```

```

            when others = >
                GENERIC_COMPONENTS : COLOUR_COEFFICIENT_ARRAY;

```

```

        end case;
    end record;

```

```

-- Defines the representation of a colour as a combination of components
-- whose meaning depends on the colour model. Additional variants may be
-- included when additional registered or unregistered colour models
-- are supported by an implementation.

```

COMPOSITION_TYPE

```

type COMPOSITION_TYPE is (PRECONCATENATE,
                           POSTCONCATENATE,
                           REPLACE);

```

```

-- Defines the type of matrix composition allowed between
-- modelling transformations.

```

CONFLICT_RESOLUTION

type CONFLICT_RESOLUTION is (MAINTAIN,
ABANDON,
UPDATE);

-- Defines a type for specifying the conflict resolution mode.

CONNECTION_ID

type CONNECTION_ID is new PHIGS_STRING;

-- Provides for specifying connection identifiers.

CONTROL_FLAG

type CONTROL_FLAG is (CONDITIONALLY,
ALWAYS);

-- The control flag is used to indicate the conditions
-- under which the display surface should be cleared.

DC

package DC is new PHIGS_COORDINATE_SYSTEM (DC_TYPE);

-- Defines the Device Coordinate System.

DC_TYPE

type DC_TYPE is digits PHIGS_PRECISION;

-- The type of a coordinate component in the Device Coordinate
-- System.

DC_UNITS

type DC_UNITS is (METRES,
OTHER);

-- Device coordinate units for a particular workstation
-- may be in metres, or some other unit (such as inches).

DEFERRAL_MODE

type DEFERRAL_MODE is (ASAP,
BNIG,
BNIL,
ASTI,
WAIT);

-- Defines the five PHIGS deferral modes.

DEVICE_NUMBER

subtype DEVICE_NUMBER is PHIGS_POSITIVE;

-- Defines the base type for input device numbers.

DISPLAY_CLASS

type DISPLAY_CLASS is (VECTOR_DISPLAY,
RASTER_DISPLAY,
OTHER_DISPLAY);

-- The classification of a workstation of category OUTPUT
-- or OUTIN.

DISPLAY_PRIORITY

type DISPLAY_PRIORITY is
digits PHIGS_PRECISION range 0.0..1.0;

-- Defines type for specifying structure display priority.

DISPLAY_SURFACE_EMPTY

type DISPLAY_SURFACE_EMPTY is (EMPTY,
NOTEMPTY);

-- Indicates whether there is graphical output on the display
-- surface.

DYNAMIC_MODIFICATION

type DYNAMIC_MODIFICATION is (IRG,
IMM,
CBS);

- Indicates whether an update to the state list is performed
 - immediately (IMM), requires implicit regeneration (IRG), or
 - can be simulated (CBS).
-

ECHO_SWITCH

type ECHO_SWITCH is (ECHO,
NOECHO);

- Indicates whether or not echoing of the measure is
 - performed.
-

EDGE_DATA

type EDGE_DATA is record
 FLAG_ASF : ASF;
 TYPE_OF_EDGE_ASF : ASF;
 EDGEWIDTH_SF_ASF : ASF;
 EDGE_COLOUR_ASF : ASF;
 FLAG : EDGE_FLAG;
 TYPE_OF_EDGE : EDGETYPE;
 EDGEWIDTH_SF : EDGEWIDTH;
 EDGE_COLOUR : COLOUR_INDEX;
end record;

- A record containing information needed to specify the
 - appearance of a fill area set edge.
-

EDGE_FLAG

type EDGE_FLAG is new OFF_ON;

- Defines a type for indicating whether or not to depict
 - edges on fill area set and related primitives.
-

EDGE_INDEX

type EDGE_INDEX is new PHIGS_POSITIVE;

- Defines the edge bundle table indices.

EDGE_INDICES

package EDGE_INDICES is
 new PHIGS_LIST_UTILITIES (EDGE_INDEX,
 MAX_EDGE_INDICES_SUPPORTED);

-- Provides for a list of edge indices.

EDGETYPE

type EDGETYPE is new PHIGS_INTEGER;

-- Defines a type for describing the form of edges.

EDGETYPES

package EDGETYPES is new PHIGS_LIST_UTILITIES (EDGETYPE,
 MAX_EDGETYPES_SUPPORTED);

-- Provides for lists of edgetypes.

EDGEWIDTH

type EDGEWIDTH is
 new SCALE_FACTOR range 0.0..SCALE_FACTOR'LAST;

-- Defines a type for describing the width scale factor of
-- edges.

EDIT_MODE

type EDIT_MODE is (INSERT,
 REPLACE);

-- Defines a type for specifying the mode in which editing
-- takes place.

ELEMENT_POSITION

type ELEMENT_POSITION is new PHIGS_INTEGER;

-- Base type for element pointer values and offsets.

ELEMENT_TYPE

type ELEMENT_TYPE is
 (ALL_ELEMENT_TYPES,
 NIL,

 POLYLINE_3,
 POLYLINE,

 POLYMARKER_3,
 POLYMARKER,

 TEXT_3,
 TEXT,

 ANNOTATION_TEXT_RELATIVE_3,
 ANNOTATION_TEXT_RELATIVE,

 FILL_AREA_3,
 FILL_AREA,
 FILL_AREA_SET_3,
 FILL_AREA_SET,

 CELL_ARRAY_3,
 CELL_ARRAY,

 GDP_3,
 GDP,

 SET_POLYLINE_INDEX,
 SET_POLYMARKER_INDEX,
 SET_TEXT_INDEX,
 SET_INTERIOR_INDEX,
 SET_EDGE_INDEX,

 SET_LINETYPE,
 SET_LINEWIDTH_SCALE_FACTOR,
 SET_POLYLINE_COLOUR_INDEX,

 SET_MARKER_TYPE,
 SET_MARKER_SIZE_SCALE_FACTOR,
 SET_POLYMARKER_COLOUR_INDEX,

 SET_TEXT_FONT,
 SET_TEXT_PRECISION,
 SET_CHAR_EXPANSION_FACTOR,
 SET_CHAR_SPACING,
 SET_TEXT_COLOUR_INDEX,
 SET_CHAR_HEIGHT,
 SET_CHAR_UP_VECTOR,
 SET_TEXT_PATH,
 SET_TEXT_ALIGNMENT,

 SET_ANNOTATION_TEXT_CHAR_HEIGHT,
 SET_ANNOTATION_TEXT_CHAR_UP_VECTOR,
 SET_ANNOTATION_TEXT_PATH,
 SET_ANNOTATION_TEXT_ALIGNMENT,
 SET_ANNOTATION_STYLE,

```

SET_INTERIOR_STYLE,
SET_INTERIOR_STYLE_INDEX,
SET_INTERIOR_COLOUR_INDEX,

SET_EDGE_FLAG,
SET_EDGE_TYPE,
SET_EDGEWIDTH_SCALE_FACTOR,
SET_EDGE_COLOUR_INDEX,

SET_PATTERN_SIZE,
SET_PATTERN_REFERENCE_POINT_AND_VECTORS,
SET_PATTERN_REFERENCE_POINT,

ADD_NAMES_TO_SET,
REMOVE_NAMES_FROM_SET,

SET_INDIVIDUAL_ASF,
SET_HLHSR_IDENTIFIER,

SET_LOCAL_TRANSFORMATION_3,
SET_LOCAL_TRANSFORMATION,
SET_GLOBAL_TRANSFORMATION_3,
SET_GLOBAL_TRANSFORMATION,
SET_MODELLING_CLIPPING_VOLUME_3,
SET_MODELLING_CLIPPING_VOLUME,
SET_MODELLING_CLIPPING_INDICATOR,
RESTORE_MODELLING_CLIPPING_VOLUME,
SET_VIEW_INDEX,

EXECUTE_STRUCTURE,

LABEL,
APPLICATION_DATA,
GSE,
SET_PICK_IDENTIFIER);

```

-- Provides for specification of element types.

ELEMENT_TYPES

```

package ELEMENT_TYPES is
    new PHIGS_LIST_UTILITIES (ELEMENT_TYPE,
                               ELEMENT_TYPE'POS(ELEMENT_TYPE'LAST)+1);

```

-- Provides for lists of element types.

ERROR_HANDLING_MODE

```

type ERROR_HANDLING_MODE is new OFF_ON;

```

-- Type for inquiring the error handling mode.

ERROR_NUMBER

type ERROR_NUMBER is new PHIGS_INTEGER;

-- Defines the type for error indicator values.

ESCAPE_ID

type ESCAPE_ID is new PHIGS_INTEGER;

-- Defines a type for identifying different escapes.

ESCAPE_IDS

package ESCAPE_IDS is new PHIGS_LIST_UTILITIES (ESCAPE_ID,
MAX_ESCAPE_IDS_SUPPORTED);

-- Provides for lists of Escape ID's.

ESCAPE_DATA_RECORD

type ESCAPE_DATA_RECORD (ESCAPE : ESCAPE_ID := 0) is
 record
 case ESCAPE is

 -- For each ESCAPE which needs a record, an
 -- implementation dependent entry is defined.
 -- For others, a null record is defined.

 when others =>
 null;
 end case;
 end record;

-- Provides for the definition of ESCAPE data records.

EVENT_DEVICE_NUMBER

type EVENT_DEVICE_NUMBER (CLASS : INPUT_CLASS := NONE) is
 record
 case CLASS is

 when NONE => null;

 when LOCATOR_INPUT
 => LOCATOR_EVENT_DEVICE : LOCATOR_DEVICE_NUMBER;

 when STROKE_INPUT
 => STROKE_EVENT_DEVICE : STROKE_DEVICE_NUMBER;

```

when VALUATOR INPUT
    => VALUATOR_EVENT_DEVICE : VALUATOR_DEVICE_NUMBER;

when CHOICE INPUT
    => CHOICE_EVENT_DEVICE : CHOICE_DEVICE_NUMBER;

when PICK INPUT
    => PICK_EVENT_DEVICE : PICK_DEVICE_NUMBER;

when STRING INPUT
    => STRING_EVENT_DEVICE : STRING_DEVICE_NUMBER;

end case;
end record;

```

-- Provides for returning any class of device number from the
-- event queue.

EVENT_QUEUE_DEVICE_NUMBER

```

type EVENT_QUEUE_DEVICE_NUMBER
    (CLASS : INPUT_QUEUE_CLASS := LOCATOR_INPUT) is
    record
        case CLASS is

            when LOCATOR INPUT
                => LOCATOR_EVENT_DEVICE : LOCATOR_DEVICE_NUMBER;

            when STROKE INPUT
                => STROKE_EVENT_DEVICE : STROKE_DEVICE_NUMBER;

            when VALUATOR INPUT
                => VALUATOR_EVENT_DEVICE : VALUATOR_DEVICE_NUMBER;

            when CHOICE INPUT
                => CHOICE_EVENT_DEVICE : CHOICE_DEVICE_NUMBER;

            when PICK INPUT
                => PICK_EVENT_DEVICE : PICK_DEVICE_NUMBER;

            when STRING INPUT
                => STRING_EVENT_DEVICE : STRING_DEVICE_NUMBER;

        end case;
    end record;

```

-- Allows EVENT_DEVICE_NUMBERS which cannot have value NONE. This is
-- used when returning only real input classes as causes of event
-- queue overflow and for flushing the queue.

FILE_ID

```

subtype FILE_ID is PHIGS_STRING;

```

-- Provides for file identifiers.

FILE_SMALL_NATURAL

```

subtype FILE_SMALL_NATURAL is
    PHIGS_NATURAL range 0..MAX_FILE_SMALL_NATURAL;

-- This is a subtype declaration which allows for unconstrained
-- record objects for VARIABLE_FILE_ID type without
-- causing the exception STORAGE_ERROR to be raised.

```

GDP_3_ID

```

type GDP_3_ID is new PHIGS_INTEGER;

-- Defines a type for selecting a Generalized Drawing
-- Primitive.

```

GDP_3_IDS

```

package GDP_3_IDS is new PHIGS_LIST_UTILITIES (GDP_3_ID,
                                                MAX_GDP_3_IDS_SUPPORTED);

-- Provides for lists of 3D Generalized Drawing Primitive ID's.

```

GDP_3_RECORD

```

type GDP_3_RECORD (GDP : GDP_3_ID := 0) is
    record
        case GDP is
            -- For each supported GDP_3 which needs a record, an
            -- implementation dependent entry is defined.
            -- For others, a generic record is defined.

            when others =>
                GDP_INFO : APPLICATION_DATA_RECORD;
            end case;
        end record;

-- Provides for the definition of GDP_3 data records.

```

GDP_ID

```

type GDP_ID is new PHIGS_INTEGER;

-- Defines a type for selecting a Generalized Drawing
-- Primitive.

```

GDP_IDS

```
package GDP_IDS is new PHIGS_LIST_UTILITIES (GDP_ID,
                                              MAX_GDP_IDS_SUPPORTED);

-- Provides for lists of Generalized Drawing Primitive ID's.
```

GDP_RECORD

```
type GDP_RECORD (GDP : GDP_ID := 0) is
  record
    case GDP is
      -- For each supported GDP which needs a record, an
      -- implementation dependent entry is defined.
      -- For others, a generic record is defined.

      when others =>
        GDP_INFO : APPLICATION_DATA_RECORD;
    end case;
  end record;

-- Provides for the definition of GDP data records.
```

GENERAL_FLOAT_PARAMETER

```
type GENERAL_FLOAT_PARAMETER is digits PHIGS_PRECISION;

-- Defines a type for specifying implementation-specific real
-- data in GDP, GSE, and ESCAPE data records.
```

GSE_ID

```
type GSE_ID is new PHIGS_INTEGER;

-- Defines a type for selecting a Generalized Structure
-- Element.
```

GSE_IDS

```
package GSE_IDS is new PHIGS_LIST_UTILITIES (GSE_ID,
                                              MAX_GSE_IDS_SUPPORTED);

-- Provides for lists of Generalized Structure Element ID's.
```

GSE_RECORD

```

type GSE_RECORD (GSE : GSE_ID := 0) is
  record
    case GSE is
      -- For each supported GSE which needs a record, an
      -- implementation dependent entry is defined.
      -- For others, a generic record is defined.

      when others =>
        GSE_INFO : APPLICATION_DATA_RECORD;
      end case;
    end record;

    -- Provides for the definition of GSE data records.

```

HATCH_STYLE

```

subtype HATCH_STYLE is STYLE_INDEX;

-- Defines the interior hatch styles type.

```

HATCH_STYLES

```

package HATCH_STYLES is
  new PHIGS_LIST_UTILITIES (HATCH_STYLE,
                           MAX_HATCH_STYLES_SUPPORTED);

-- Provides for lists of hatch styles.

```

HLHSR_ID

```

type HLHSR_ID is new PHIGS_INTEGER;

-- A number used as an attribute for workstation independent
-- HLHSR factors.

```

HLHSR_IDS

```

package HLHSR_IDS is new PHIGS_LIST_UTILITIES (HLHSR_ID,
                                                MAX_HLHSR_IDS_SUPPORTED);

-- Provides for lists of HLHSR identifiers.

```

Tables

HLHSR_MODE

type HLHSR_MODE is new PHIGS_INTEGER;

-- A number used for controlling workstation dependent
-- HLHSR factors.

HLHSR_MODES

package HLHSR_MODES is
 new PHIGS_LIST_UTILITIES (HLHSR_MODE,
 MAX_HLHSR_MODES_SUPPORTED);

-- Provides for lists of HLHSR modes.

HORIZONTAL_ALIGNMENT

type HORIZONTAL_ALIGNMENT is (NORMAL,
 LEFT,
 CENTRE,
 RIGHT);

-- The alignment of the text extent rectangle with
-- respect to horizontal positioning of the text.

INPUT_CLASS

type INPUT_CLASS is (NONE,
 LOCATOR_INPUT,
 STROKE_INPUT,
 VALUATOR_INPUT,
 CHOICE_INPUT,
 PICK_INPUT,
 STRING_INPUT);

-- Defines the input device classifications for
-- workstations of category INPUT or OUTIN.

INPUT_QUEUE_CLASS

subtype INPUT_QUEUE_CLASS is
 INPUT_CLASS range LOCATOR_INPUT .. STRING_INPUT;

-- Defines the input device classifications for workstations
-- of category INPUT and OUTIN that do not return a NONE
-- classification.

INPUT_STATUS

type INPUT_STATUS is (OK,
NONE);

- Defines the status of a locator, stroke, valuator, or
 - string operation.
-

INPUT_STRING

type INPUT_STRING is (LENGTH : INPUT_STRING_SMALL_NATURAL := 0) is
record
CONTENTS : PHIGS_STRING (1..LENGTH);
end record;

- Provides a variable length string for use in STRING input functions.
 - Objects of this type should be declared unconstrained to allow for
 - dynamic modification of the length.
-

INPUT_STRING_SMALL_NATURAL

subtype INPUT_STRING_SMALL_NATURAL is
PHIGS_NATURAL range 0..MAX_INPUT_STRING_SMALL_NATURAL;

- This is a subtype declaration which allows for unconstrained
 - record objects for INPUT_STRING type without
 - causing the exception STORAGE_ERROR to be raised.
-

INTENSITY

subtype INTENSITY is COLOUR_COEFFICIENT range 0.0..1.0;

- Defines the restricted range of colour components required
- by some colour models.

INTERIOR_DATA

```
type INTERIOR_DATA is
  record
    STYLE_OF_INTERIOR_ASF : ASF;
    STYLE_IND_ASF         : ASF;
    INTERIOR_COLOUR_ASF   : ASF;
    INTERIOR_IND           : INTERIOR_INDEX;
    STYLE_OF_INTERIOR      : INTERIOR_STYLE;
    STYLE_IND              : STYLE_INDEX;
    INTERIOR_COLOUR        : COLOUR_INDEX;
  end record;
```

```
-- A record containing information needed to specify the
-- appearance of a filled area interior.
```

INTERIOR_INDEX

```
type INTERIOR_INDEX is new PHIGS_POSITIVE;
```

```
-- Defines interior bundle table indices.
```

INTERIOR_INDICES

```
package INTERIOR_INDICES is
  new PHIGS_LIST_UTILITIES (INTERIOR_INDEX,
                             MAX_INTERIOR_INDICES_SUPPORTED);
```

```
-- Provides for lists of interior bundle table indices.
```

INTERIOR_STYLE

```
type INTERIOR_STYLE is (HOLLOW,
                        SOLID,
                        PATTERN,
                        HATCH,
                        EMPTY);
```

```
-- Defines the interior styles for filled areas.
```

INTERIOR_STYLES

```
package INTERIOR_STYLES is
  new PHIGS_LIST_UTILITIES
    (INTERIOR_STYLE,
     INTERIOR_STYLE'POS(INTERIOR_STYLE'LAST)+1);
```

```
-- Provides for lists of interior styles.
```

INVALID_VALUES_INDICATOR

type INVALID_VALUES_INDICATOR is (ABSENT,
PRESENT);

- Indicates whether invalid values are contained in
 - a colour array or matrix.
-

LABEL_ID

type LABEL_ID is new PHIGS_INTEGER;

- Defines a type for specifying structure element labels.
-

LINE_DATA

type LINE_DATA is
 record
 TYPE_OF_LINE_ASF : ASF;
 WIDTH_ASF : ASF;
 LINE_COLOUR_ASF : ASF;
 POLYLINE_IND : POLYLINE_INDEX;
 TYPE_OF_LINE : LINETYPE;
 WIDTH : LINEWIDTH;
 LINE_COLOUR : COLOUR_INDEX;
 end record;

- A record containing information needed to specify the
 - appearance of a line for input data records.
-

LINETYPE

type LINETYPE is new PHIGS_INTEGER;

- Defines the types of line styles provided by PHIGS.
-

LINETYPES

package LINETYPES is new PHIGS_LIST_UTILITIES (LINETYPE,
MAX_LINETYPES_SUPPORTED);

- Provides for lists of linetypes.

LINEWIDTH

type LINEWIDTH is
new SCALE_FACTOR range 0.0..SCALE_FACTOR'LAST;

-- The width of a line is indicated by a scale factor.

LOCATOR_DEVICE_NUMBER

type LOCATOR_DEVICE_NUMBER is new DEVICE_NUMBER;

-- Provides for locator device identifiers.

LOCATOR_PROMPT_ECHO_TYPE

type LOCATOR_PROMPT_ECHO_TYPE is new PHIGS_INTEGER;

-- Defines the locator prompt and echo types supported by
-- the implementation.

LOCATOR_PROMPT_ECHO_TYPES

package LOCATOR_PROMPT_ECHO_TYPES is
new PHIGS_LIST_UTILITIES
(LOCATOR_PROMPT_ECHO_TYPE,
MAX_LOCATOR_PROMPT_ECHO_TYPES_SUPPORTED);

-- Provides for lists of locator prompt and echo types.

MARKER_DATA

type MARKER_DATA is
record
TYPE_OF_MARKER_ASF : ASF;
SIZE_ASF : ASF;
MARKER_COLOUR_ASF : ASF;
POLYMARKER_IND : POLYMARKER_INDEX;
TYPE_OF_MARKER : MARKER_TYPE;
SIZE : MARKER_SIZE;
MARKER_COLOUR : COLOUR_INDEX;
end record;

-- A record containing information needed to specify the
-- appearance of a marker for input data records.

MARKER_SIZE

type MARKER_SIZE is new SCALE_FACTOR;
-- The size of a marker is indicated by a scale factor.

MARKER_TYPE

type MARKER_TYPE is new PHIGS_INTEGER;
-- Defines the type for markers provided by PHIGS.

MARKER_TYPES

package MARKER_TYPES is
 new PHIGS_LIST_UTILITIES (MARKER_TYPE,
 MAX_MARKER_TYPES_SUPPORTED);
-- Provides for lists of marker types.

MC

package MC is new PHIGS_COORDINATE_SYSTEM (MC_TYPE);
-- Defines the Modelling Coordinate System.

MC_TYPE

type MC_TYPE is digits PHIGS_PRECISION;
-- Defines the type of coordinate in the Modelling
-- Coordinate System.

MEMORY_UNITS

type MEMORY_UNITS is range 0..MAX_MEMORY_UNITS;
-- Defines the type for units of memory that may be
-- allocated by PHIGS.

METAFILE_ITEM_LENGTH

type METAFILE_ITEM_LENGTH is
new PHIGS_NATURAL range 0..MAX_METAFILE_ITEM_LENGTH;

-- The length of an item contained in a metafile.

METAFILE_ITEM_TYPE

type METAFILE_ITEM_TYPE is
new PHIGS_NATURAL
range MIN_METAFILE_ITEM_TYPE..MAX_METAFILE_ITEM_TYPE;

-- The type of an item contained in a metafile.

MODELLING_CLIP_OPERATION_TYPE

type MODELLING_CLIP_OPERATION_TYPE is new PHIGS_INTEGER;

-- This type provides for specifying modelling clip operation
-- types.

MODELLING_CLIP_OPERATION_TYPES

package MODELLING_CLIP_OPERATION_TYPES is
new PHIGS_LIST_UTILITIES
(MODELLING_CLIP_OPERATION_TYPE,
MAX_MODELLING_CLIP_OPERATION_TYPES_SUPPORTED);

-- This type provides for lists of modelling clip operation types.

MODIFICATION_MODE

type MODIFICATION_MODE is (NIVE,
UWOR,
UQUM);

-- Defines type for specifying modification modes.

MORE_EVENTS

type MORE_EVENTS is (NOMORE,
MORE);

-- Indicates whether more simultaneous events are contained
-- in the input queue.

NAME

subtype NAME is
 PHIGS_NAME_SET_FACILITY.NAME;

-- Provides for names for each of the name set classes.
-- See 5.14.4.

NAMES

package NAMES renames
 PHIGS_NAME_SET_FACILITY.NAMES;

-- Provides for lists of name set classes. See 5.14.4.

NAME_SET

subtype NAME_SET is PHIGS_NAME_SET_FACILITY.NAME_SET;

-- Used to define name sets. The internal structure of
-- a name set is implementation-dependent. "Build" functions
-- contained in the package PHIGS_NAME_SET_FACILITIES shall be
-- used to create and manipulate name sets (See 5.14.4).

NAME_SET_ACCEPTANCE

subtype NAME_SET_ACCEPTANCE is
 PHIGS_NAME_SET_FACILITY.NAME_SET_ACCEPTANCE;

-- Used to indicate the results of applying a name set
-- against a filter pair. See 5.14.4.

NAME_SET_FILTER

subtype NAME_SET_FILTER is
 PHIGS_NAME_SET_FACILITY.NAME_SET_FILTER;

-- Used to define name set pairs used as filters.
-- See 5.14.4.

Tables

NAME_SET_FILTERS

package NAME_SET_FILTERS is
 new PHIGS_LIST_UTILITIES
 (NAME_SET_FILTER,
 MAX_NAME_SET_FILTER_LIST_LENGTH_SUPPORTED);

-- Used to provide lists of name set filters.
-- See 5.14.4.

NAME_SET_MEMBERSHIP

subtype NAME_SET_MEMBERSHIP is
 PHIGS_NAME_SET_FACILITY.NAME_SET_MEMBERSHIP;

-- Provides for indicating whether a class is contained in a
-- name set. See 5.14.4.

NPC

package NPC is new PHIGS_COORDINATE_SYSTEM (NPC_TYPE);

-- Defines the Normalized Projection Coordinate System.

NPC_TYPE

type NPC_TYPE is digits PHIGS_PRECISION;

-- Defines the type of a coordinate in the Normalized
-- Projection Coordinate System.

OFF_ON

type OFF_ON is (OFF,
 ON);

-- Generic type for off/on switches.

OPEN_STRUCTURE_STATUS

type OPEN_STRUCTURE_STATUS is (NONE,
 OPEN);

-- Returns the status of the open structure.