The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Note

Before using this information and the product it supports, read the information in “Notices,” on page 529.

Fourth Edition (October 2000)

This edition applies to the IBM PEX/PHIGS for AIX Version 4, Program Number 5696-907, and to all subsequent releases of this product until otherwise indicated in new editions.

A reader's comment form is provided at the back of this publication. If the form has been removed, address comments to Information Development, Department H6DS-905-6C006, 11501 Burnet Road, Austin, Texas 78758-3493. To send comments electronically, use this commercial Internet address: aix6kpub@austin.ibm.com. Any information that you supply may be used without incurring any obligation to you.

© Copyright International Business Machines Corporation 1994, 2002. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
# Contents

**About This Book** ................................................................. xi
Who Should Use This Book .................................................. xi
Highlighting ........................................................................... xi
ISO 9000 .................................................................................. xi
Related Publications ............................................................. xi

## Chapter 1. Introduction

- Calling Conventions for ISO PHIGS Subroutines .................... 1
- graPHIGS API Subroutines .................................................... 1
- Subroutine Descriptions ...................................................... 2
- Reference Manual Abbreviations ......................................... 2

## Chapter 2. Control Subroutines

- CLOSE PHIGS (PHOP,WSCL,STCL,ARCL) ............................. 5
- Close Workstation (PHOP,*,STOP,*) .................................. 5
- MESSAGE (PHOP,WSOP,*) ................................................ 6
- OPEN PHIGS (PHCL, WSCL, STCL, ARCL) ......................... 7
- OPEN WORKSTATION (PHOP,*,*,*) ................................... 9
- REDRAW ALL STRUCTURES (PHOP,WSOP,*) ....................... 11
- SET DISPLAY UPDATE STATE (PHOP,WSOP,*) ................. 12
- UPDATE WORKSTATION(PHOP,WSOP,*) ............................ 13

## Chapter 3. Output Primitives

- ANNOTATION TEXT RELATIVE (PHOP,*,STOP,*) .................... 15
- ANNOTATION TEXT RELATIVE 3 (PHOP,*,STOP,*) .............. 17
- CELL ARRAY (PHOP,*,STOP,*) ........................................... 18
- CELL ARRAY 3 (PHOP,*,STOP,*) ....................................... 20
- FILL AREA (PHOP,*,STOP,*) ............................................. 21
- FILL AREA 3 (PHOP,*,STOP,*) .......................................... 22
- FILL AREA SET (PHOP,*,STOP,*) ...................................... 23
- FILL AREA SET 3 (PHOP,*,STOP,*) .................................. 24
- GENERALIZED DRAWING PRIMITIVE (PHOP,*,STOP,*) ...... 25
- GENERALIZED DRAWING PRIMITIVE 3 (PHOP,*,STOP,*) .... 27
- POLYLINE (PHOP,*,STOP,*) ............................................. 28
- POLYLINE 3 (PHOP,*,STOP,*) .......................................... 29
- POLYMARKER (PHOP,*,STOP,*) ....................................... 30
- POLYMARKER 3 (PHOP,*,STOP,*) .................................... 30
- TEXT (PHOP,*,STOP,*) ................................................... 31
- TEXT 3 (PHOP,*,STOP,*) .................................................. 33

## Chapter 4. Attribute Specification

- ADD NAMES TO SET (PHOP,*,STOP,*) ................................ 37
- REMOVE NAMES FROM SET (PHOP,*,STOP,*) ....................... 38
- SET ANNOTATION STYLE (PHOP,*,STOP,*) ......................... 39
- SET ANNOTATION TEXT ALIGNMENT (PHOP,*,STOP,*) .......... 40
- SET ANNOTATION TEXT CHARACTER HEIGHT (PHOP,*,STOP,*) 41
- SET ANNOTATION TEXT CHARACTER UP VECTOR (PHOP,*,STOP,*) 42
- SET ANNOTATION TEXT PATH (PHOP,*,STOP,*) ..................... 43
- SET CHARACTER EXPANSION FACTOR (PHOP,*,STOP,*) ....... 44
- SET CHARACTER HEIGHT (PHOP,*,STOP,*) ......................... 44
- SET CHARACTER SPACING (PHOP,*,STOP,*) ........................ 45
- SET CHARACTER UP VECTOR (PHOP,*,STOP,*) .................... 46
- SET EDGE COLOR INDEX (PHOP,*,STOP,*) .......................... 47

© Copyright IBM Corp. 1994, 2002
Chapter 5. Miscellaneous Structure Element Subroutines

APPLICATION DATA (PHOP*,STOP*). 77
EXECUTE STRUCTURE (PHOP*,STOP*). 77
GENERALIZED STRUCTURE ELEMENT (PHOP*,STOP*). 78

Chapter 6. Structure Operation Subroutines

CHANGE STRUCTURE IDENTIFIER (PHOP*,**,*). 81
CHANGE STRUCTURE IDENTIFIER AND REFERENCES (PHOP*,**,*). 82
CHANGE STRUCTURE REFERENCES (PHOP*,**,*). 83
CLOSE STRUCTURE (PHOP*,STOP*). 84
COPY ALL ELEMENTS FROM STRUCTURE (PHOP*,STOP*). 84
DELETE ALL STRUCTURES (PHOP*,**,*). 85
DELETE ELEMENT (PHOP*,STOP*). 85
DELETE ELEMENT RANGE (PHOP*,STOP*). 86
DELETE ELEMENTS BETWEEN LABELS (PHOP*,STOP*). 87
DELETE STRUCTURE (PHOP*,**,*). 88
DELETE STRUCTURE NETWORK (PHOP*,**,*). 88
EMPTY STRUCTURE (PHOP*,**,*). 89
LABEL (PHOP*,STOP*). 90
OFFSET ELEMENT POINTER (PHOP*,STOP*). 91
OPEN STRUCTURE (PHOP*,STCL*). 91
SET EDIT MODE (PHOP*,**,*). 92
SET ELEMENT POINTER (PHOP*,STOP*). 93
SET ELEMENT POINTER AT LABEL (PHOP*,STOP*). 94

Chapter 7. Workstation Table Settings

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chapter 8. Structure Display Subroutines</strong></td>
<td>113</td>
</tr>
<tr>
<td>POST STRUCTURE (PHOP,WSOP,***)</td>
<td>113</td>
</tr>
<tr>
<td>UNPOST ALL STRUCTURES (PHOP,WSOP,**)</td>
<td>114</td>
</tr>
<tr>
<td>UNPOST STRUCTURE (PHOP,WSOP,***)</td>
<td>115</td>
</tr>
<tr>
<td><strong>Chapter 9. Structure Archiving Subroutines</strong></td>
<td>117</td>
</tr>
<tr>
<td>ARCHIVE ALL STRUCTURES (PHOP,**,AROP)</td>
<td>117</td>
</tr>
<tr>
<td>ARCHIVE STRUCTURE NETWORKS (PHOP,**,AROP)</td>
<td>118</td>
</tr>
<tr>
<td>ARCHIVE STRUCTURES (PHOP,**,AROP)</td>
<td>119</td>
</tr>
<tr>
<td>CLOSE ARCHIVE FILE (PHOP,**,AROP)</td>
<td>120</td>
</tr>
<tr>
<td>DELETE ALL STRUCTURES FROM ARCHIVE (PHOP,**,AROP)</td>
<td>120</td>
</tr>
<tr>
<td>DELETE STRUCTURE NETWORKS FROM ARCHIVE (PHOP,**,AROP)</td>
<td>121</td>
</tr>
<tr>
<td>DELETE STRUCTURES FROM ARCHIVE (PHOP,**,AROP)</td>
<td>122</td>
</tr>
<tr>
<td>OPEN ARCHIVE FILE (PHOP,**,* )</td>
<td>123</td>
</tr>
<tr>
<td>RETRIEVE ALL STRUCTURES (PHOP,**,AROP)</td>
<td>125</td>
</tr>
<tr>
<td>RETRIEVE PATHS TO ANCESTORS (PHOP,**,AROP)</td>
<td>126</td>
</tr>
<tr>
<td>RETRIEVE PATHS TO DESCENDANTS (PHOP,**,AROP)</td>
<td>127</td>
</tr>
<tr>
<td>RETRIEVE STRUCTURE IDENTIFIERS (PHOP,**,AROP)</td>
<td>129</td>
</tr>
<tr>
<td>RETRIEVE STRUCTURE NETWORKS (PHOP,**,AROP)</td>
<td>130</td>
</tr>
<tr>
<td>SET CONFLICT RESOLUTION (PHOP,**,*)</td>
<td>131</td>
</tr>
<tr>
<td><strong>Chapter 10. Transformation Subroutines</strong></td>
<td>133</td>
</tr>
<tr>
<td>RESTORE MODELING CLIPPING VOLUME (PHOP,<em>STOP,</em>)</td>
<td>135</td>
</tr>
<tr>
<td>SET GLOBAL TRANSFORMATION (PHOP,<em>STOP,</em>)</td>
<td>136</td>
</tr>
<tr>
<td>SET GLOBAL TRANSFORMATION 3 (PHOP,<em>STOP,</em>)</td>
<td>136</td>
</tr>
<tr>
<td>SET LOCAL TRANSFORMATION (PHOP,<em>STOP,</em>)</td>
<td>137</td>
</tr>
<tr>
<td>SET LOCAL TRANSFORMATION 3 (PHOP,<em>STOP,</em>)</td>
<td>138</td>
</tr>
<tr>
<td>SET MODELING CLIPPING VOLUME (PHOP,<em>STOP,</em>)</td>
<td>139</td>
</tr>
<tr>
<td>SET MODELING CLIPPING VOLUME 3 (PHOP,<em>STOP,</em>)</td>
<td>140</td>
</tr>
<tr>
<td>SET MODELING CLIPPING INDICATOR (PHOP,<em>STOP,</em>)</td>
<td>142</td>
</tr>
<tr>
<td>SET WORKSTATION VIEWPORT (PHOP,WSOP,***)</td>
<td>143</td>
</tr>
<tr>
<td>SET WORKSTATION VIEWPORT 3 (PHOP,WSOP,***)</td>
<td>144</td>
</tr>
<tr>
<td>SET WORKSTATION WINDOW (PHOP,WSOP,***)</td>
<td>145</td>
</tr>
<tr>
<td>SET WORKSTATION WINDOW 3 (PHOP,WSOP,***)</td>
<td>146</td>
</tr>
<tr>
<td><strong>Chapter 11. Input Subroutines</strong></td>
<td>149</td>
</tr>
<tr>
<td>AWAIT EVENT (PHOP,WSOP,***)</td>
<td>149</td>
</tr>
<tr>
<td>FLUSH DEVICE EVENTS (PHOP,WSOP,***)</td>
<td>151</td>
</tr>
<tr>
<td>GET CHOICE (PHOP,WSOP,***)</td>
<td>152</td>
</tr>
<tr>
<td>GET LOCATOR (PHOP,WSOP,***)</td>
<td>153</td>
</tr>
</tbody>
</table>
Chapter 12. Utility Subroutines

BUILD TRANSFORMATION MATRIX (PHOP,*,*) ......................................................... 217
BUILD TRANSFORMATION MATRIX 3 (PHOP,*,*) ...................................................... 218
COMPOSE MATRIX (PHOP,*,*) .................................................................................. 220
COMPOSE MATRIX 3 (PHOP,*,*) ............................................................................... 221
COMPOSE TRANSFORMATION MATRIX (PHOP,*,*) .................................................... 222
COMPOSE TRANSFORMATION MATRIX 3 (PHOP,*,*) .................................................. 224
CREATE STORE (PHOP,*,*) ....................................................................................... 226
DELETE STORE (PHOP,*,*) ......................................................................................... 226
EVALUATE VIEW MAPPING MATRIX (PHOP,*,*) ........................................................ 227
EVALUATE VIEW MAPPING MATRIX 3 (PHOP,*,*) ...................................................... 228
EVALUATE VIEW ORIENTATION MATRIX (PHOP,*,*) .................................................. 230
EVALUATE VIEW ORIENTATION MATRIX 3 (PHOP,*,*) .............................................. 231
PACK DATA RECORD (PHOP,*,*) ............................................................................... 233
Chapter 13. Error Control Subroutines ............................................. 251
EMERGENCY CLOSE PHIGS (PHCL,WSCL,STCL,ARCL) ......................... 251
ERROR HANDLING (PHCL,WSCL,STCL,ARCL) .................................... 251
ERROR LOGGING (PHCL,WSCL,STCL,ARCL) ..................................... 253
SET ERROR HANDLING (PHCL,WSCL,STCL,ARCL) .............................. 253
SET ERROR HANDLING MODE (PHOP,**,***) .................................... 254

Chapter 14. Special Interface Subroutines ........................................ 257
ESCAPE (PHOP,WSCL,STCL,ARCL) .................................................. 257

Chapter 15. Inquire Subroutines ..................................................... 259
ELEMENT SEARCH (PHOP,**,***) ................................................... 259
INQUIRE ALL CONFLICTING STRUCTURES (PHOP,**,***,AROP) ............... 262
INQUIRE ANNOTATION FACILITIES (PHOP,**,***) ............................... 263
INQUIRE ARCHIVE FILES (PHOP,**,***) .......................................... 264
INQUIRE ARCHIVE STATE VALUE (PHCL,WSCL,STCL,ARCL) .................. 266
INQUIRE CHOICE DEVICE STATE (PHOP,WSOP,**,***) .......................... 266
INQUIRE CHOICE DEVICE STATE 3 (PHOP,WSOP,**,***) ....................... 268
INQUIRE COLOR FACILITIES (PHOP,**,***) ....................................... 270
INQUIRE COLOR MODEL (PHOP,WSOP,**,***) .................................... 271
INQUIRE COLOR MODEL FACILITIES (PHOP,**,**,***) ......................... 272
INQUIRE COLOR REPRESENTATION (PHOP,WSOP,**,***) ....................... 274
INQUIRE CONFLICT RESOLUTION (PHOP,**,***) ................................ 275
INQUIRE CONFLICTING STRUCTURES IN NETWORK (PHOP,**,***,AROP) .... 276
INQUIRE CURRENT ELEMENT CONTENT (PHOP,*STOP,* ) ....................... 278
INQUIRE CURRENT ELEMENT TYPE AND SIZE (PHOP,*STOP,* ) ............... 280
INQUIRE DEFAULT CHOICE DEVICE DATA (PHOP,**,***) ....................... 281
INQUIRE DEFAULT CHOICE DEVICE DATA 3 (PHOP,**,***) ..................... 283
INQUIRE DEFAULT DISPLAY UPDATE STATE (PHOP,**,***) ..................... 285
INQUIRE DEFAULT LOCATOR DEVICE DATA (PHOP,**,***) ...................... 287
INQUIRE DEFAULT LOCATOR DEVICE DATA 3 (PHOP,**,**,***) ............... 288
INQUIRE DEFAULT PICK DEVICE DATA (PHOP,**,***) ........................... 290
INQUIRE DEFAULT PICK DEVICE DATA 3 (PHOP,**,**,***) ..................... 292
INQUIRE DEFAULT STRING DEVICE DATA (PHOP,**,***) ....................... 294
INQUIRE DEFAULT STRING DEVICE DATA 3 (PHOP,**,**,***) ................. 296
INQUIRE DEFAULT STROKE DEVICE DATA (PHOP,**,***) ....................... 298
INQUIRE DEFAULT STROKE DEVICE DATA 3 (PHOP,**,**,***) ................... 300
INQUIRE DEFAULT VALUATOR DEVICE DATA (PHOP,**,***) .................... 302
INQUIRE DEFAULT VALUATOR DEVICE DATA 3 (PHOP,**,**,***) ............... 304
INQUIRE DISPLAY SPACE SIZE (PHOP,**,***) .................................... 305
INQUIRE DISPLAY SPACE SIZE 3 (PHOP,**,**,***) ................................ 307
INQUIRE DISPLAY UPDATE STATE (PHOP,WSOP,**,***) ........................ 308
INQUIRE DYNAMICS OF STRUCTURES (PHOP,**,***) ............................ 310
INQUIRE DYNAMICS OF WORKSTATION ATTRIBUTES (PHOP,**,***) ........... 311
Chapter 16. ISO PHIGS Transformations

3-by-3 Matrix. 435
4-by-4 Matrix. 435

Chapter 17. FORTRAN Structure Content Data Records 437

Chapter 18. ISO PHIGS C Type and Macro Definitions 457
Function identifiers 484
Error codes 487

Chapter 19. ISO PHIGS FORTRAN Enumeration Types 497

Chapter 20. ISO PHIGS Subroutines to GPxxxx Subroutines 503

Chapter 21. GPxxxx Subroutines to ISO PHIGS Subroutines 509

Chapter 22. Implementation Errors and graPHIGS API Messages for ISO PHIGS-Defined Errors 517

Chapter 23. graPHIGS API Extensions and Compatibility with the ISO PHIGS Standard 521
graPHIGS API Extensions to the ISO PHIGS Standard 521
Compatibility between graPHIGS API Extensions and the ISO PHIGS Standard 521

Chapter 24. graPHIGS API Deviations from the ISO PHIGS Standard 527
Unsupported Subroutines 527
Structure Building 527
Color Components 527
Input 528
Traversal Defaults 528
Errors 528
About This Book

This book contains information you need to code your ISO PHIGS calls and to declare variables correctly. Each subroutine has information about error codes and functional relations to help you identify the source of errors resulting from data and program flow. Each subroutine description explains the result of the subroutine call and a list of the ISO PHIGS standard errors associated with the subroutine.

Who Should Use This Book

This book is intended for application programmers.

Highlighting

The following highlighting conventions are used in this book:

**Bold** Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.

*Italics* Identifies parameters whose actual names or values are to be supplied by the user.

Monospace Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

Related Publications

The following books contain information on graPHIGS API products:

- [The graPHIGS Programming Interface: Customization and Problem Diagnosis](#)
- [The graPHIGS Programming Interface: Getting Started](#)
- [The graPHIGS Programming Interface: ISO PHIGS Quick Reference](#)
- [The graPHIGS Programming Interface: Messages and Codes](#)
- [The graPHIGS Programming Interface: Quick Reference](#)
- [The graPHIGS Programming Interface: Subroutine Reference](#)
- [The graPHIGS Programming Interface: Technical Reference](#)
- [The graPHIGS Programming Interface: Understanding Concepts](#)
Chapter 1. Introduction

This manual describes the syntax and operations of the subroutines available in the graPHIGS API ISO PHIGS interface.

Calling Conventions for ISO PHIGS Subroutines

The graPHIGS API supports C, FORTRAN 77, and FORTRAN 77 Subset bindings for ISO PHIGS defined subroutines. Each invocation consists of the subroutine name with the appropriate parameters required by the subroutine. Mnemonic conventions make the purpose of the subroutine calls evident in the names. In your application, you must use all the parameters specified in each subroutine and the parameters must be in the order shown in the subroutine description.

graPHIGS API Subroutines

For each ISO PHIGS subroutine, you will find the purpose of the subroutine with a description, the processing performed, a list and description of the parameters, a list of associated ISO-defined error codes and messages, and a list of related subroutines. Refer to the appendices for enumerations and type definitions for the supported bindings. Binding errors and implementation errors are not listed with each subroutine. See Chapter 21. “Implementation Errors and graPHIGS API Messages for ISO PHIGS-Defined Errors” for a discussion of implementation errors and a list of binding errors. See Chapter 23. “graPHIGS API Deviations from the ISO PHIGS Standard” for a discussion of the handling of asynchronous errors.

The ISO PHIGS subroutines have been grouped as follows:

- 1. Control
- 2. Output Primitives
- 3. Attribute Structure Elements
- 4. Miscellaneous Structure Elements
- 5. Structure Operations
- 6. Workstation Table Settings
- 7. Display
- 8. Transformation
- 9. Input
- 10. Utilities
- 11. Error Handling
- 12. Miscellaneous
- 13. Inquiries

In addition, this manual contains the following appendixes:

- A. Transformations for the C and FORTRAN bindings
- B. Format and content of Structure Element Records for the FORTRAN binding
- C. C binding enumerations and type definitions
- D. FORTRAN binding enumerations
- E. ISO PHIGS subroutines mapped to GPxxxx subroutines
- F. GPxxxx subroutines mapped to ISO PHIGS subroutines
- G. ISO PHIGS subroutine errors
- H. graPHIGS API extensions and the combining of GPxxxx subroutine with ISO PHIGS subroutines
- I. graPHIGS API deviations from the ISO PHIGS standard
Subroutine Descriptions

The page preceding each group of subroutines presents a brief overview of the purpose and result caused by invoking the subroutines referenced within the given section. This introductory material highlights important information that applies to all the subroutines in that section.

Reference Manual Abbreviations

The following abbreviations are used frequently:

- **ARCL**: Archive Closed
- **AROP**: Archive Open
- **ASAP**: As Soon As Possible
- **ASF**: Attribute Source Flag
- **ASTI**: At Some Time
- **BNIG**: Before Next Interaction Globally
- **BNIL**: Before Next Interaction Locally
- **CIELUV**: CIELUV color model system
- **CMY**: Cyan-Magenta-Yellow color model
- **CSID**: Character Set Identifier
- **CSS**: Centralized Structure Store
- **EDF**: External Defaults File
- **GDP**: Generalized Drawing Primitive
- **GSE**: Generalized Structure Element
- **HLHSR**: Hidden Line Hidden Surface Removal
- **HSV**: Hue-Saturation-Value color model
- **NROP**: Non-Retained Structure Open
- **PDT**: graPHIGS API Description Table
- **PET**: Prompt and Echo Type
- **PHCL**: graPHIGS Closed
- **PHOP**: graPHIGS Open
- **PSL**: graPHIGS API State List
- **RGB**: Red-Green-Blue color model
- **STCL**: Structure Closed
- **STOP**: Structure Open
- **USL**: Utility State List
WAIT  When Application Requests It  
WDT   Workstation Description Table  
WSCL  Workstation Closed  
WSID  Workstation Identifier  
WSL   Workstation State List  
WSOP  Workstation Open  
WSTYPE Workstation Type  

The following abbreviations for coordinate spaces are used:  
MC    Modeling Coordinates  
WC    World Coordinates  
VC    Viewing Coordinates  
NPC   Normalized Projection Coordinates  
DC    Device Coordinates  

Note: For more information, see The graPHIGS Programming Interface: Understanding Concepts.
Chapter 2. Control Subroutines

The control subroutines allow your application to have access to and control over the graphical resources available when using the graPHIGS API.

Your application can open and close the graPHIGS API.

When an application opens the graPHIGS API, a nucleus, which is the collection of resources available to your application, is connected to your application. When connected, you can open, update, or close workstation resources.

Also invoke these subroutines to affect the timing of update operations and to explicitly control the update and redraw operations on a workstation.

These subroutines do not store or modify graphics data.

CLOSE PHIGS (PHOP,WSCL,STCL,ARCL)

Purpose

Use Close PHIGS to terminate all graPHIGS API processing for this application process. This subroutine function detaches all attached resources created by the application and disconnects all nuclei connected to the application. Close PHIGS closes files and releases system resources, such as storage and locks. Close PHIGS sets the graPHIGS system state to graPHIGS Closed (PHCL). Reopen the graPHIGS API by invoking the Open PHIGS subroutine.

Language Bindings

C

pclose_phigs();

FORTRAN

PCLPH

Errors

4 FUNCTION REQUIRES STATE (PHOP,WSCL,STCL,ARCL)

Related Subroutines

• Open PHIGS

Close Workstation (PHOP,*,*,*)

Purpose

Use Close Workstation to close the specified workstation. The workstation updates automatically before closing.

This subroutine function releases the workstation state list and deletes the workstation’s identifier from the set of opened workstations in the graPHIGS API state list. Additionally, it flushes the input queue of all events from all input devices on that workstation, and releases the connection to the workstation. If no workstation remains open, the workstation state is set to Workstation Closed (WSCL).
Close Workstation clears the workstation. For workstations that keep a local copy of the structure store, the graPHIGS API frees the structure storage at this time.

**Language Bindings**

**C**

```
pclose_ws(ws_id);
```

**Input Parameters**

*Pint ws_id*

Workstation identifier.

**FORTRAN**

```
pcwlk(wkid)
```

**Input Parameters**

*integer wkid*

Workstation identifier.

**Errors**

<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>FUNCTION REQUIRES STATE (PHOP,WSOP,<em>,</em>)</td>
</tr>
<tr>
<td>54</td>
<td>SPECIFIED WORKSTATION IS NOT OPEN</td>
</tr>
<tr>
<td>256</td>
<td>WARNING, INPUT QUEUE HAS OVERFLOWED</td>
</tr>
</tbody>
</table>

**Related Subroutines**

- Open Workstation.
- Inquire Set of Open Workstations.

---

**MESSAGE (PHOP,WSOP,*,*)**

**Purpose**

Use Message to display a message on the specified workstation.

The message text appears in the lower left corner of the workstation viewport and the graPHIGS API clips the message text to this viewport.

Deferral state settings do not affect the message. The message remains displayed until removed by another message. Clear the message by calling Message with a length of zero.

The appearance (size and color) of the message text is workstation dependent. The graPHIGS API uses the workstation’s primary character set to convert the text if necessary.

For more details, refer to the specific device-support information in *The graPHIGS Programming Interface: Technical Reference*.

**Language Bindings**
C

`pmessage (ws_id, message);`

**Input Parameters**

*Pint* `ws_id`  
Workstation identifier

*`const char *message`*  
Message string to be displayed.

**FORTRAN**

`pmsg (wkid, mess)`

**Input Parameters**

`integer wkid`  
Workstation identifier

`character(*) mess`  
Message string to be displayed.

**FORTRAN Subset**

`pmsgs (wkid, lstr, mess)`

**Input Parameters**

`integer wkid`  
Workstation identifier

`integer lstr`  
Length of the message string in characters.

`character(*) mess`  
Message string to be displayed.

**Errors**

3  
FUNCTION REQUIRES STATE (PHOP,WSOP,*,*)

54  
SPECIFIED WORKSTATION IS NOT OPEN

**Related Subroutines**

None

---

**OPEN PHIGS (PHCL, WSCL, STCL, ARCL)**

**Purpose**

Use Open PHIGS to open and initialize the graPHIGS API. Open PHIGS makes all the graPHIGS API subroutines available. Call Open PHIGS before invoking most other graPHIGS API subroutines.
This subroutine function initializes the graPHIGS API state list (PSL). Open PHIGS sets the system state to graPHIGS Open (PHOP), sets the workstation state value to Workstation Closed (WSCL), sets the structure state value to Structure Closed (STCL), and sets the archive state value to Archive Closed (ARCL).

The External Defaults File (EDF) allows the application to modify the graPHIGS API system and workstation defaults. See The graPHIGS Programming Interface: Technical Reference, “Defaults and Nicknames,” for contents and formats of the External Defaults File (EDF).

When your application issues this Open PHIGS subroutine, the graPHIGS API implicitly connects to a private nucleus, unless the EDF explicitly controls the connection to a specified nucleus. For your application to use the 6090 workstation, you must indicate it by using the DEFNUC default in the External Defaults File (EDF).

With Open PHIGS you cannot suppress nucleus creation via the External Defaults File (EDF).

The error file parameter determines the target for logged error messages.

To determine if the Open PHIGS subroutine was successful, use the Inquire System State Value subroutine.


Language Bindings

C

popen_phigs (err_file, mem_units);

Input Parameters

const char *err_file

Name of the error file. This parameter determines the target for logged error messages. Handling of error messages varies depending on the error file character string and the environment (where 0<character string<=80). This parameter looks like a Unix file descriptor which consists of a [path]/filename[extension]. Path is the route of directories through the file system on an AIX system. Path is optional and ignored for MVS and VM. An example of a full file descriptor:

/phigs/errors/appl1

where:

• path = /phigs which says go from the root directory to directory phigs
• filename
• extension = .appl1

The following rules apply to the name depending on which system the shell is running in:

AIX If you did not specify the path, then the graPHIGS API uses the default directory at the time of the execution of the subroutine.

MVS, MVS/XA

• filename - You must supply a filename. If you specify the extension, then as the filename you must specify the member name within a partition data set. If you do not specify the extension, then as the filename you must specify the DD-name of the partition data set.
• extension - The extension is optional. If you specify the extension, then you must specify it as the DD-name of the partition data set including the member filename.

VM/CMS

• filename - You must specify the filename.
• extension - The extension is optional. If you specify the extension, then it is the filetype. If you do not specify the extension, then the graPHIGS API uses a filetype of AFMPELOG.

• The graPHIGS API uses a filemode of A1.
• The graPHIGS API makes the filename.extension upper case.

size_t mem_units

size_t units of memory available for buffer space. The graPHIGS API ignores this parameter.

FORTRAN

popph (errfil, bufa)

Input Parameters

integer errfil

Name of the error file. This parameter determines the target for logged error messages. (0=console, 1=afmerror). Handling of error messages varies depending on the error file and the environment.

If the error file identifier has a value of 0:

AIX      Messages are sent to stderr
MVS      Messages are logged to the console.
VM       Messages are logged to the console.

If the error file identifier has a value of 0:

AIX      Filename afmerror.
MVS      AFMERROR is the DDNAME of the dataset.
VM       Filename AFMERROR and filetype AFMPELOG.

integer bufa

Amount of units of memory available for buffer area. The graPHIGS API ignores this parameter.

Errors

1      FUNCTION REQUIRES STATE (PHCL, WSCL, STCL, ARCL)
450    SPECIFIED ERROR FILE IS INVALID

Related Subroutines

• Close PHIGS
• Inquire System State Value

OPEN WORKSTATION (PHOP,*,*,*)

Purpose

Use Open Workstation to open and initialize a specified workstation. This subroutine function sets the workstation state value to Workstation Open (WSOP). The graPHIGS API requests the operating system to establish the specified connection. This subroutine function allocates and initializes the Workstation
State List (WSL) according to the Workstation Descriptor Table (WDT) for the specified type. The workstation is associated with the specified identifier. Open Workstation adds this specified identifier to the set of open workstations in the graPHIGS API state list.

The graPHIGS API External Defaults File (EDF) allows the application to denote, indirectly, the actual values of both the workstation type and the connection identifier. For more information, see *The graPHIGS Programming Interface: Technical Reference*, “Defaults and Nicknames,” for contents and formats of the External Defaults File (EDF).

**Language Bindings**

**C**

```c
popen_ws (ws_id, conn_id, ws_type);
```

**Input Parameters**

- `Pint ws_id`
  - Workstation identifier.
- `const void *conn_id`
  - Connection identifier indicates the physical device to be opened. See *The graPHIGS Programming Interface: Technical Reference* for valid connection identifiers. The connection identifier is a pointer to a character string.
- `Pint ws_type`
  - One of the graPHIGS API supported workstation types (1=6090, 2=5080, 3=GDDM, 4=GDF, 5=CGM, 6=X, 8=XSOFT, 9=XPEX, 10=IMAGE). See *The graPHIGS Programming Interface; Technical Reference*, for explanations of these workstation types.

**FORTRAN**

```fortran
popwk (wkid, conid, wtype);
```

- `integer wkid`
  - Workstation identifier.
- `integer conid`
  - Connection identifier indicates the physical device to be opened. Select any integer value between 1 and 99999999 for a connection identifier. An association between this integer and a valid connection string is then made via an EDF (External Defaults File). (See *The graPHIGS Programming Interface: Technical Reference* for valid connection strings and explanations of the EDF file).

For example, the connection identifier of ‘*’ is attained if the EDF contains the line:

```
AFMMNICK CONNID=99, TOCONNID=*
```

and the application is coded:

```
WSID=1
CONID=99
WTYPE=6
POPKW=(WKID, CONID, WTYPE)
```

- `integer wtype`
  - One of the graPHIGS API supported workstation types (1=6090, 2=5080, 3=GDDM, 4=GDF, 5=CGM, 6=X, 8=XSOFT, 9=XPEX, 10=IMAGE). See *The graPHIGS Programming Interface; Technical Reference*, for explanations of these workstation types.
Errors
None

Related Subroutines
Close Workstation, Inquire Workstation Connection and Type.

REDRAW ALL STRUCTURES (PHOP,WSOP,*,*)

Purpose
Use Redraw All Structures to redraw all structures on the specified workstation.

When your application invokes this subroutine, the graPHIGS API executes all the actions in the sequence outlined below:

1. The graPHIGS API executes all deferred actions for the specified workstation without an intermediate clearing of the display surface.
2. If the control flag is set to CONDITIONALLY, and only if the Display Surface Empty entry in the Workstation State List (WSL) is set to NOTEMPTY, then the graPHIGS API clears the display surface. If the control flag is set to ALWAYS, then the graPHIGS API clears the display surface regardless of the setting of the Display Surface Empty entry. At the conclusion of this step, the graPHIGS API sets the entry in the WSL to EMPTY.
3. If the view orientation matrix, view mapping matrix, view clipping limits, x to y clipping indicator, back clipping indicator, or the front clipping indicator have changed for any view, then the graPHIGS API assigns the current entries in the Workstation State List (WSL) to the corresponding values from the requested entries. The Transformation Update State is set to NOTPENDING.
4. If the Hidden Line/Hidden Surface Removal (HLHSR) mode has changed, then the graPHIGS API assigns the current WSL entry to the corresponding value from the requested entry. The HLHSR Update State is set to NOTPENDING.
5. Finally the graPHIGS API retraverses all structures posted to this workstation. If the set of structures associated with this workstation is not empty, retraversal usually sets the Display Surface Empty entry in the WSL to NOTEMPTY. The graPHIGS API sets the state of visual representation in the WSL to CORRECT.

Language Bindings

C

predraw_all_structs (ws_id, ctrl_flag);

Input Parameters
Pint ws_id

   Workstation identifier
Pctrl_flag ctrl_flag

   Control flag (0=PFLAG_COND, 1=PFLAG_ALWAYS).

FORTRAN

prst (wkid, coff)

Input Parameters
integer wkid
    Workstation identifier.

integer cofl
    Control flag (0=PCONDI, 1=PALWAY).

Errors
3    FUNCTION REQUIRES STATE (PHOP,WSOP,*,*)
54   SPECIFIED WORKSTATION IS NOT OPEN
59   SPECIFIED WORKSTATION DOES NOT HAVE OUTPUT CAPABILITY

Related Subroutines
• Update Workstation

SET DISPLAY UPDATE STATE (PHOP,WSOP,*,*)

Purpose
Use Set Display Update State to set the deferral state and modification mode for the specified workstation state list.

Possible deferral modes include: AS SOON AS POSSIBLE, BEFORE NEXT INTERACTION GLOBALLY, BEFORE NEXT INTERACTION LOCALLY, AT SOME TIME and WAIT. Possible modification modes include: NO IMMEDIATE VISUAL EFFECTS, UPDATE WITHOUT REGENERATION, and USE QUICK UPDATE METHOD.

These settings determine when pending updates are processed for display on a workstation and how the workstation performs the modifications. For an explanation of the abbreviations and modes, see The graPHIGS Programming Interface: Understanding Concepts. For specific workstation information, see the section on "General Output Facilities" in The graPHIGS Programming Interface: Technical Reference. Quick update methods are discussed in The graPHIGS Programming Interface: Writing Applications.

Language Bindings

C

pset_disp_upd_st (ws_id, def_mode, mod_mode);

Input Parameters

Pint ws_id
    Workstation identifier.

Pdefer_mode def_mode
    Deferral mode (0=PDEFER_ASP, 1=PDEFER_BNIG, 2=PDEFER_BNIL, 3=PDEFER_ASTI, 4=PDEFER_WAIT).

Pmod_mode mod_mode
    Modification mode (0=PMODE_NIVE, 1=PMODE_UWOR, 2=PMODE_UQUM).

FORTRAN

PSDUS (WKID, DEFMOD, MODMOD)

Input Parameters
INTEGER WKID
   Workstation identifier.

INTEGER DEFMOD
   Deferral mode (0=PASSP, 1=PBNIG, 2=PBNIL, 3=PASTI, 4=PWAITD).

INTEGER MODMOD
   Modification mode (0=PNIVE, 1=PUWOR, 2=PUQUM).

Errors
3      Function Requires State (PHOP,WSOP,*,*)
54     Specified Workstation Is Not Open
59     Specified Workstation Does Not Have Output Capability

Related Subroutines
• Inquire Display Update State
• Inquire Default Display Update State

UPDATE WORKSTATION(PHOP,WSOP,*,*)

Purpose

Use Update Workstation to update the specified workstation.

This subroutine function executes all deferred actions for the specified workstation without intermediate clearing of the display surface. If the regeneration flag is set to PERFORM and the state of visual representation in the Workstation State List (WSL) is DEFERRED or SIMULATED, then the graPHIGS API executes all the actions in the sequence outlined below:
1. If the Display Surface Empty entry in the Workstation State List (WSL) is set to NOTEMPTY, then the graPHIGS API clears the display surface. At the conclusion of this step (Step 1), the graPHIGS API sets the entry to EMPTY.
2. If the view orientation matrix, view mapping matrix, view clipping limits, x to y clipping indicator, back clipping indicator, or the front clipping indicator have changed for any view, then the graPHIGS API assigns the current entries in the Workstation State List (WSL) to the corresponding values from the requested entries. The Transformation Update State is set to NOTPENDING.
3. If the Hidden Line/Hidden Surface Removal (HLHSR) mode has changed, then the graPHIGS API assigns the current WSL entry to the corresponding value from the requested entry. The HLHSR Update State is set to NOTPENDING.
4. The graPHIGS API re-displays all structures posted to this workstation. Usually, this action sets the Display Surface Empty entry in the WSL to NOTEMPTY.
5. The state of visual representation is set to CORRECT in the WSL. If the state of visual representation in the Workstation State List (WSL) is DEFERRED or SIMULATED and the regeneration flag is set to PERFORM, then this subroutine is functionally equivalent to the Redraw All Structures subroutine.

When the regeneration flag is set to POSTPONE, the device sends the pending update information without forcing a retraversal if possible. This function is workstation dependent.

Language Bindings

C

pupd_ws (ws_id, regen_flag);
Input Parameters

\textit{Pint ws\_id}

Workstation identifier.

\textit{Pregen\_flag regen\_flag}

Regeneration flag (0=\textit{PFLAG\_POSTPONE}, 1=\textit{PFLAG\_PERFORM}).

\textbf{FORTRAN}

\texttt{PUWK(WKID, REGFL)}

Input Parameters

\textit{INTEGER WKID}

Workstation identifier.

\textit{INTEGER REGFL}

Regeneration flag (0=\textit{PPOSTP}, 1=\textit{PPERFO}).

\textbf{Errors}

3 Function Requires State (PHOP,WSOP,*;*)

54 Specified Workstation Is Not Open

59 Specified Workstation Does Not Have Output Capability

\textbf{Related Subroutines}

- Inquire Display Update State
Chapter 3. Output Primitives

These subroutines address the specification and creation of output primitives, which are structure elements. Many have both two- and three-dimensional forms and are displayed when the structure elements defining them are encountered during structure traversal. To use primitive subroutines, the structure state must be Structure Open (STOP).

For all two-dimensional output primitive subroutines, the z coordinate is assumed to equal zero by default.

If a specified workstation does not support a requested output primitive in a structure, then the graPHIGS API updates only the element number in the graPHIGS traversal state list.

**Note:** When the application inserts an element into the open structure following the element pointer, the pointer updates to that element.

---

**ANNOTATION TEXT RELATIVE (PHOP,*,STOP,*)**

**Purpose**

Use Annotation Text Relative to insert a two-dimensional Annotation Text Relative 2 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with an Annotation Text Relative 2 structure element, depending on the current edit mode.

During structure traversal, this element annotates the specified reference point according to the annotation style in the traversal state list. The specified annotation offset determines the position of the annotation string. The annotation string defines the origin of a local text coordinate system relative to a specified reference point after transformation to Normalized Projection Coordinates (NPC). The text plane is always parallel to the x, y plane (z= transformed location) in NPC. If the resulting text position is outside the usable NPC space [0,1]x[0,1]x[0,1], then the graPHIGS API may clip part or all of the string.

The graPHIGS API positions and renders the text string in the local coordinate system according to the annotation text attributes in the traversal state list. If the graPHIGS API clips the specified reference point to NPC during structure traversal, then no representation for this primitive is displayed. If the graPHIGS API does not clip the reference point, then the graPHIGS API clips the displayed representation according to the rules for the corresponding primitive type (e.g., text, polyline, etc.).

The graPHIGS API treats control characters in a character string as undefined characters and displays the default for the character set. This default for the character set is the default character in the graPHIGS API character set file. For U.S. English, this is the hyphen character (\texttt{EBCDIC X'60', ASCII X'2D')}.

If the annotation style attribute entry in the PHIGS traversal state list is set to \textit{LEAD LINE}, then after transformation the graPHIGS API draws a single line segment from the specified reference point to the origin of the local text coordinate system using the polyline attributes in the PHIGS traversal state list.

**Language Bindings**

**C**

\texttt{panno_text_rel(ref_pt, offset, char_string)}

**Input Parameters**

\texttt{const Ppoint *ref_pt}

Reference point in MC.
**const Pvec *offset**
Annotation offset in NPC. Determines the position of the annotation character string.

**const char *char_string**
Annotation character string to be displayed.

**FORTRAN**

`PATR(rpx, rpy, apx, apy, chars)`

**Input Parameters**

- **real rpx**
  x coordinate of the reference location, in MC, that is to be annotated.

- **real rpy**
  y coordinate of the reference location, in MC, that is to be annotated.

- **real apx**
  x component of the annotation offset in NPC. Determines the position of the annotation character string (x component).

- **real apy**
  y component of the annotation offset in NPC. Determines the position of the annotation character string (y component).

- **character*80 chars**
  Annotation character string to be displayed.

**FORTRAN Subset**

`PATRS(rpx, rpy, apx, apy, lstr, chars)`

**Input Parameters**

- **real rpx**
  x coordinate of the reference location, in MC, that is to be annotated.

- **real rpy**
  y coordinate of the reference location, in MC, that is to be annotated.

- **real apx**
  x component of the annotation offset in NPC. Determines the position of the annotation character string (x component).

- **real apy**
  y component of the annotation offset in NPC. Determines the position of the annotation character string (y component).

- **integer lstr**
  Length of string in characters.

- **character*80 chars**
  Annotation character string to be displayed.

**Errors**

5 Function Requires State (PHOP,*,STOP,*

**Related Subroutines**

- Set Annotation Style
ANNOTATION TEXT RELATIVE 3 (PHOP*, STOP*)

Purpose

Use Annotation Text Relative 3 to insert an Annotation Text Relative 3 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with an Annotation Text Relative 3 structure element, depending on the current edit mode.

During structure traversal, this element annotates the specified reference point according to the annotation style in the traversal state list. The specified annotation offset determines the position of the annotation string. The annotation string defines the origin of a local text coordinate system relative to the specified reference point after transformation to Normalized Projection Coordinates (NPC). The text plane is always parallel to the x, y plane in NPC. If the resulting text position is outside the usable NPC space \([0,1] \times [0,1] \times [0,1]\), then the graPHIGS API may clip part or all of the string.

The graPHIGS API positions and renders the text string in the local coordinate system according to the annotation text attributes in the traversal state list. If the graPHIGS API clips the specified reference point to NPC during structure traversal, then no representation for this primitive is displayed. If the graPHIGS API does not clip the specified reference point, then the graPHIGS API clips the displayed representation according to the rules for the corresponding primitive type (e.g., text, polyline, etc.).

The graPHIGS API treats control characters in a character string as undefined characters and displays the default for the character set. This default for the character set is the default character in the graPHIGS API character set file. For U.S. English, this is the hyphen character \((EBCDIC X'60', ASCII X'2D')\).

If the annotation style attribute entry in the PHIGS traversal state list is set to \(LEAD\ LINE\), then after transformation the graPHIGS API draws a single line segment from the specified reference point to the origin of the local text coordinate system using the polyline attributes in the PHIGS traversal state list.

Language Bindings

C

panno_text_rel3(ref_pt, offset, char_string)

Input Parameters

\texttt{const Ppoint3 \textbullet\ ref\_pt}

Reference point in MC.

\texttt{const Pvec3 \textbullet\ offset}

Annotation offset in NPC. Determines the position of the annotation character string.

\texttt{const char \textbullet\ char\_string}

Annotation character string to be displayed.

FORTRAN

PATR3(rpx, rpy, rpz, apx, apy, apz, chars)

Input Parameters

\texttt{real rpx}

\(x\) coordinate of the reference location, in MC, that is to be annotated.

\texttt{real rpy}

\(y\) coordinate of the reference location, in MC, that is to be annotated.
real rpz
  z coordinate of the reference location, in MC, that is to be annotated.

real apx
  x component of the annotation offset in NPC. Determines the position of the annotation character string (x component).

real apy
  y component of the annotation offset in NPC. Determines the position of the annotation character string (y component).

real apz
  z component of the annotation offset in NPC. Determines the position of the annotation character string (z component).

character*80 chars
  Annotation character string to be displayed.

FORTRAN Subset

PATR3S(rp, rpz, apx, apy, apz, lstr, chars)

Input Parameters

real rp
  x coordinate of the reference location, in MC, that is to be annotated.

real rp
  y coordinate of the reference location, in MC, that is to be annotated.

real rpz
  z coordinate of the reference location, in MC, that is to be annotated.

real apx
  x component of the annotation offset in NPC. Determines the position of the annotation character string (x component).

real apy
  y component of the annotation offset in NPC. Determines the position of the annotation character string (y component).

real apz
  z component of the annotation offset in NPC. Determines the position of the annotation character string (z component).

integer lstr
  Length of string in characters.

character*80 chars
  Annotation character string to be displayed.

Errors

5  Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Annotation Style

CELL ARRAY (PHOP,*,STOP,*)

Purpose
Use Cell Array to create a two-dimensional \((x, y)\) cell array primitive with the \(z\) coordinate assumed to be zero, and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Cell Array structure element, depending on the current edit mode.

This structure element defines a two-dimensional array of cells with individual colors. The primitive is defined by two points, \(P\) and \(Q\), which define a rectangle aligned with the modeling coordinate axes. This rectangle is conceptually divided into a grid of \(DX\) by \(DY\) cells. Each cell has a width of \(|PX-QX|/DX\), and a height of \(|PY-QY|/DY\), where \((PX, PY)\) are the coordinates of the corner point \(P\), and \((QX, QY)\) are the coordinates of the corner point \(Q\). The color of each cell is specified by the index of the corresponding element of the color index array. The color indexes are mapped into the two-dimensional cell array on a row-wise basis starting at corner \((PX, PY)\) and proceeding to corner \(Q\) and so on. If an index is not present in the color table on a workstation, then the graPHIGS API uses an index value of 1 on that workstation.

When the graPHIGS API encounters an element of this type, it does a minimal simulation by drawing the transformed boundaries of the cell rectangle using polyline color, a line width value of 1, and a line type of \(SOLID\).

**Language Bindings**

**C**

```
pcell_array (rect, colr_array)
```

**Input Parameters**

- `const Prect *rect`
  - Cell rectangle in MC.
- `const Ppat_rep *colr_array`
  - Color array.

**FORTRAN**

```
PCA (px, py, qx, qy, dimx, dimy, isc, isr, dx, dy, colia)
```

**Input Parameters**

- `real px`
  - \(x\) coordinate of the point \(P\) in MC.
- `real py`
  - \(y\) coordinate of the point \(P\) in MC.
- `real qx`
  - \(x\) coordinate of the point \(Q\) in MC.
- `real qy`
  - \(y\) coordinate of the point \(Q\) in MC.
- `integer dimx`
  - \(x\) dimension of \(COLIA\) which contains the cell array.
- `integer dimy`
  - \(y\) dimension of \(COLIA\) which contains the cell array.
- `integer isc`
  - Index of the start column of the cell array within \(COLIA\).
- `integer isr`
  - Index of the start row of the cell array within \(COLIA\).
integer dx
  Number of cell array columns.

integer dy
  Number of cell array rows.

integer cola(dimx,dimy)
  Color index array containing the cell array.

Errors
5  Function Requires State (PHOP,*,STOP,*)
117 One Dimension Of Color Index Array < Zero
113  Color Index Value < ZERO

Related Subroutines
  • Set Polyline Color Index

CELL ARRAY 3 (PHOP,*,STOP,*)

Purpose

Use Cell Array 3 to create a three-dimensional cell array primitive, and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Cell Array 3 structure element, depending on the current edit mode.

This structure element defines a two-dimensional array of cells with individual colors. The plane in which the Cell Array 3 primitive lies is defined by three points, \( P \), \( Q \), and \( R \), given in modeling coordinates (MC). A parallelogram is defined by the points \( P \), \( Q \), and \( R \), and \((QX+RX-PX, QY+RY-PY, QZ+RZ-PZ)\). This parallelogram is conceptually divided into a grid of \( DX \) by \( DY \) cells, where \( DX \) and \( DY \) are the dimensions of the color index array. The color of each cell is specified by the index of the corresponding element of the color index array. The color indexes are mapped from the two-dimensional cell array on a row-wise basis starting at corner \( P \) and proceeding to corner \( Q \) and so on. If an index is not present in the color table on a workstation, then the graPHIGS API uses an index value of 1 on that workstation.

When the graPHIGS API encounters an element of this type, it does a minimal simulation by drawing the transformed boundaries of the cell parallelogram using polyline color, a line width value of 1, and a line type of SOLID.

Language Bindings

C

cell_array3 (paral, colr_array)

Input Parameters

const Pparal *paral
  Cell parallelogram in MC.

const Ppat_rep *colr_array
  Color array.

FORTRAN

PCA3 (cpxa, cpya, cpzA, dimx, dimy, isc, isr, dx, dy, cola)
Input Parameters

real cpxa(3)
   x coordinates of the points P, Q, and R in MC.

real cpya(3)
   y coordinates of the points P, Q, and R in MC.

real cpza(3)
   z coordinates of the points P, Q, and R in MC.

integer dimx
   x dimension of COLIA which contains the cell array.

integer dimy
   y dimension of COLIA which contains the cell array.

integer isc
   Index of the start column of the cell array within COLIA.

integer isr
   Index of the start row of the cell array within COLIA.

integer dx
   Number of cell array columns.

integer dy
   Number of cell array rows.

integer colia(dimx,dimy)
   Color index array containing the cell array.

Errors

5      Function Requires State (PHOP,*,STOP,*)
117    One Dimension Of Color Index Array < Zero
113    Color Index Value < ZERO

Related Subroutines

• Set Polyline Color Index

FILL AREA (PHOP,*,STOP,*)

Purpose

Use Fill Area to specify a two-dimensional fill area primitive with the z coordinate assumed to be zero, and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Fill Area structure element, depending on the current edit mode.

This structure element defines the boundary of a contour which may be hollow or filled with a uniform color, a pattern, or a hatch style. The graPHIGS API displays the boundary of the primitive without an edge.

The graPHIGS API places all points specified in the x-y plane. The graPHIGS API applies interior attributes to this primitive.

Language Bindings
C

pfill_area (point_list)

Input Parameters

const Ppoint_list *point_list
List of points in MC.

FORTRAN

PFA (n, pxa, pya)

Input Parameters

integer n
Number of points.

real pxa(n)
x coordinates of points in MC.

real pya(n)
y coordinates of points in MC.

Errors

5 Function Requires State (PHOP,*STOP,*).

Related Subroutines

• Set Interior Style
• Set Interior Style Index
• Set Interior Representation
• Set Interior Index
• Set Interior Color Index

FILL AREA 3 (PHOP,*STOP,*)

Purpose

Use Fill Area 3 to specify a three-dimensional fill area 3 primitive element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Fill Area 3 structure element, depending on the current edit mode.

This structure element defines the boundary of a contour which may be hollow or filled with a uniform color, a pattern, or a hatch style. The graPHIGS API displays the boundary of the primitive without an edge.

All points specified must lie in the same plane, but the graPHIGS API does not check to verify this. The system behavior is undefined when the points are not coplanar.

The graPHIGS API applies interior attributes to this primitive.

Language Bindings
C

pfill_area3 (point_list)

Input Parameters
const Ppoint_list3 *point_list
    List of points in MC.

FORTRAN
PFA3 (n, pxa, pya, pza)

Input Parameters
integer n
    Number of points.
real pxa (n)
    x coordinates of points in MC.
real pya (n)
    y coordinates of points in MC.
real pza (n)
    z coordinates of points in MC.

Errors
5    Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Set Interior Style
- Set Interior Style Index
- Set Interior Representation
- Set Interior Index
- Set Interior Color Index

FILL AREA SET (PHOP,*,STOP,*)

Purpose
Use Fill Area Set to specify a two-dimensional fill area set primitive with the z coordinate assumed to be zero, and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Fill Area Set structure element, depending on the current edit mode.

This structure element defines the boundary of contours which may be hollow or filled with a uniform color, a pattern, or a hatch style. Each list of points defines a subarea and each subarea is implicitly closed. The graPHIGS API displays the boundary of the primitive with an edge.

The graPHIGS API places all points specified in the x-y plane. The graPHIGS API applies interior and edge attributes to this primitive.

Language Bindings
cpfill_area_set (point_list)

Input Parameters

const Ppoint_list_list *point_list_list
  List of point lists in MC.

FORTRAN

PFAS (npl, ix, px, pya)

Input Parameters

integer npl
  Number of point lists.

integer ix(npl)
  Array of end indexes for the point lists.

real px (*)
  x coordinates of points in MC.

real py (*)
  y coordinates of points in MC.

Errors

5  Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Interior Style
- Set Interior Style Index
- Set Interior Representation
- Set Interior Index
- Set Interior Color Index
- Set Edgewidth Scale Factor
- Set Edgetype
- Set Edge Color Index
- Set Edge Index

FILL AREA SET 3 (PHOP,*,STOP,*)

Purpose

Use Fill Area Set 3 to specify a three-dimensional fill area set primitive element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Fill Area Set 3 structure element, depending on the current edit mode.

This structure element defines the boundary of contours which may be hollow or filled with a uniform color, a pattern, or a hatch style. Each list of points defines a subarea and each subarea is implicitly closed. The graPHIGS API displays the boundary of the primitive with an edge.
All points specified must lie in the same plane, but the graPHIGS API does not check to verify this. The system behavior is undefined when the points are not coplanar.

The graPHIGS API applies interior and edge attributes to this primitive.

Language Bindings

C

pfill_area_set3 (point_list_list)

Input Parameters

cost Ppoint_list_list3 *point_list_list
List of point lists in MC.

FORTRAN

PFAS3 (npl, ixa, pxa, pya, pza)

Input Parameters

integer npl
Number of point lists.

integer ixa(npl)
Array of end indexes for the point lists.

real pxa (*)
x coordinates of points in MC.

real pya (*)
y coordinates of points in MC.

real pza (*)
z coordinates of points in MC.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Set Interior Style
• Set Interior Style Index
• Set Interior Representation
• Set Interior Index
• Set Interior Color Index
• Set Edgewidth Scale Factor
• Set Edgetype
• Set Edge Color Index
• Set Edge Index

GENERALIZED DRAWING PRIMITIVE (PHOP,*,STOP,*)

Purpose
Use Generalized Drawing Primitive to specify a two-dimensional generalized drawing primitive (GDP) element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Generalized Drawing Primitive structure element, depending on the current edit mode.

There are no GDP identifiers currently defined by the graPHIGS API. Therefore, when the graPHIGS API encounters this type of element, it does not display the primitive nor does it generate an error. However, GDPs are available through the GPxxxx subroutines. See The graPHIGS Programming Interface: Subroutine Reference for details.

Language Bindings

C

pgdp (point_list, gdp_id, gdp_data)

Input Parameters

const Ppoint_list *point_list
    List of points.

Pint gdp_id
    GDP identifier.

const Pgdp_data *gdp_data
    GDP data record.

FORTRAN

PGDP (n, pxa, pya, primid, ldr, datrec)

Input Parameters

integer n
    Number of points (>=0).

real pxa (*)
    x coordinates of points in MC.

real pya (*)
    y coordinates of points in MC.

integer primid
    GDP identifier.

integer ldr
    Dimension of the GDP data record array.

character*80 datrec(ldr)
    GDP data record.

Errors

5   Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• None
GENERALIZED DRAWING PRIMITIVE 3 (PHOP*, STOP*)

**Purpose**

Use Generalized Drawing Primitive 3 to specify a three-dimensional generalized drawing primitive (GDP 3) element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Generalized Drawing Primitive 3 structure element, depending on the current edit mode.

There are no GDP 3 identifiers currently defined by the graPHIGS API. Therefore, when the graPHIGS API encounters this type of element, it does not display the primitive nor does it generate an error. However, GDPs are available through the GPxxxx subroutines. See The graPHIGS Programming Interface: Subroutine Reference for details.

**Language Bindings**

**C**

`pgdp3(point_list, gdp3_id, gdp_data)`

**Input Parameters**

- `const Ppoint_list3 *point_list`
  - List of points.
- `Pint gdp3_id`
  - GDP 3 identifier.
- `const Pgdp_data3 *gdp_data`
  - GDP 3 data record.

**FORTRAN**

`PGDP3(n, pxa, pya, pza, primid, ldr, datrec)`

**Input Parameters**

- `integer n`
  - Number of points (>=0).
- `real pxa (*)`
  - x coordinates of points in MC.
- `real pya (*)`
  - y coordinates of points in MC.
- `real pza (*)`
  - z coordinates of points in MC.
- `integer primid`
  - GDP 3 identifier.
- `integer ldr`
  - Dimension of the GDP 3 data record array.
- `character*80 datrec(ldr)`
  - GDP 3 data record.

**Errors**
POLYLINE (PHOP, *, STOP, *)

Purpose

Use Polyline to create a two-dimensional polyline element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Polyline structure element, depending on the current edit mode.

This structure element defines a list of two-dimensional points \((x, y)\) (the \(z\) coordinate is assumed to be zero) that the graPHIGS API is to connect by straight lines starting with the first point and ending with the last point.

If the application specifies one or less points, then no output is generated. If two contiguous points are the same point, then the graPHIGS API generates a point of one pixel in size.

The graPHIGS API applies polyline attributes to this primitive.

Language Bindings

C

```c
ppolyline (point_list)
```

Input Parameters

```c
const Ppoint_list *point_list
```

List of points in MC.

FORTRAN

```fortran
PPL (n, pxa, pya)
```

Input Parameters

```fortran
integer n
```

Number of points (>=0).

```fortran
real pxa (n)
```

\(x\) coordinates of points in MC.

```fortran
real pya (n)
```

\(y\) coordinates of points in MC.

Errors

5 Function Requires State (PHOP, *, STOP, *)

Related Subroutines

- Set Linetype
- Set Linewidth Scale Factor
- Set Polyline Index
POLYLINE 3 (PHOP,*,STOP,*)

Purpose

Use Polyline 3 to create a three-dimensional polyline element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Polyline 3 structure element, depending on the current edit mode.

This structure element defines a list of three-dimensional points \((x, y, z)\) that the graPHIGS API is to connect by straight lines starting with the first point and ending with the last point.

If the application specifies one or less points, then no output is generated. If two contiguous points are the same point, then the graPHIGS API generates a point of one pixel in size.

The graPHIGS API applies polyline attributes to this primitive.

Language Bindings

C

\texttt{ppolyline3 (point\_list)}

Input Parameters

\texttt{const Ppoint\_list3 *point\_list}

List of points in MC.

FORTRAN

\texttt{PPL3 (n, pxa, pya, pza)}

Input Parameters

\texttt{integer n}

Number of points \((\geq 0)\).

\texttt{real pxa (n)}

\(x\) coordinates of points in MC.

\texttt{real pya (n)}

\(y\) coordinates of points in MC.

\texttt{real pza (n)}

\(z\) coordinates of points in MC.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Linetype
- Set Linewidth Scale Factor
- Set Polyline Color Index
- Set Polyline Index
POLYMARKER (PHOP,*,STOP,*)

Purpose

Use Polymarker to create a two-dimensional polymarker element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Polymarker structure element, depending on the current edit mode.

This structure element defines a list of two-dimensional points \((x, y)\) that the graPHIGS API identifies by markers and renders in Device Coordinate (DC) space parallel to the display surface.

If the primitive does not specify any points, then it is ignored.

The graPHIGS API applies polymarker attributes to this primitive.

Language Bindings

C

ppolymarker (point_list)

Input Parameters

const Ppoint_list *point_list
List of points in MC.

FORTRAN

PPM \((n, pxa, pya)\)

Input Parameters

integer \(n\)
Number of points \((\geq 0)\).

real pxa \((n)\)
\(x\) coordinate of points in MC.

real pya \((n)\)
\(y\) coordinates of points in MC.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Marker Type
- Set Marker Size Scale Factor
- Set Polymarker Color Index
- Set Polymarker Index

POLYMARKER 3 (PHOP,*,STOP,*)

Purpose
Use Polymarker 3 to create a three-dimensional polymarker element and insert it into the open structure following the element pointer or replace the element pointed at by the element pointer with a Polymarker 3 structure element, depending on the current edit mode.

This structure element defines a list of three-dimensional points \((x, y, z)\) that the graPHIGS API identifies by markers and renders in Device Coordinate (DC) space parallel to the display surface.

If the primitive does not specify any points, then it is ignored.

The graPHIGS API applies polymarker attributes to this primitive.

**Language Bindings**

**C**

```c
ppolymarker3 (point_list)
```

**Input Parameters**

```c
const Ppoint_list3 *point_list
    List of points in MC.
```

**FORTRAN**

```fortran
PPM3 (n, pxa, pya, pza)
```

**Input Parameters**

```fortran
integer n
    Number of points (>=0).
real pxa (n)
    x coordinates of points in MC.
real pya (n)
    y coordinates of points in MC.
real pza (n)
    z coordinates of points in MC.
```

**Errors**

5    Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

- Set Marker Type
- Set Marker Size Scale Factor
- Set Polymarker Color Index
- Set Polymarker Index

**TEXT (PHOP,*,STOP,*)**

**Purpose**

Use Text to insert a two-dimensional, geometric text element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Text structure element, depending on the current edit mode.
This structure element specifies a string of geometric text that the graPHIGS API draws at the specified location in the \( x, y \) plane.

The graPHIGS API treats control characters in a character string as undefined characters and displays the default for the character set. This default for the character set is the default character in the graPHIGS API character set file. For U.S. English, this is the hyphen character (EBCDIC \( X'60 \), ASCII \( X'2D \)).

**Language Bindings**

**C**

```c
ptext (text_pos, char_string)
```

**Input Parameters**

```c
const Ppoint *text_pos
    Text position in MC.

const char *char_string
    Character string.
```

**FORTRAN**

```fortran
PTX (px, py, chars)
```

**Input Parameters**

```fortran
real px
    x coordinate of text position in MC.

real py
    y coordinate of text position in MC.

character(*) chars
    Text to be displayed.
```

**FORTRAN Subset**

```fortran
PTXS (px, py, lstr, chars)
```

**Input Parameters**

```fortran
real px
    x coordinate of text position in MC.

real py
    y coordinate of text position in MC.

integer lstr
    Length of text string in bytes (\( \geq 0 \)).

character*80 chars
    Text to be displayed.
```

**Errors**

5 Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

32 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
• Inquire Text Facilities
• Set Character Expansion Factor
• Set Character Height
• Set Character Spacing
• Set Character Up Vector
• Set Text Alignment
• Set Text Color Index
• Set Text Font
• Set Text Index
• Set Text Path
• Set Text Precision

**TEXT 3 (PHOP*, STOP*)**

**Purpose**

Use Text 3 to insert a three-dimensional, geometric text element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Text 3 structure element, depending on the current edit mode.

This structure element specifies a string of geometric text that the graPHIGS API draws on the plane defined by the specified text position and reference vectors.

Two vector definitions orient a local coordinate system, within which the text is positioned. The two reference vectors and the text position define the plane in which the text is drawn. The first vector defines the x-axis of the local coordinate system. The second reference vector defines the half plane of the text in which the positive y-axis lies. The directions specified by Character Up Vector and Text Path attributes are relative to this coordinate system.

If the direction vectors fail to define a local coordinate system (i.e., one of the vectors is zero in length or the vectors are parallel), then the graPHIGS API stores the values (1,0,0) and (0,1,0) in the element.

The graPHIGS API treats control characters in a character string as undefined characters and displays the default for the character set. This default for the character set is the default character in the graPHIGS API character set file. For U.S. English, this is the hyphen character (EBCDIC '60', ASCII '2D').

**Language Bindings**

C

```c
ptext3 (text_pos, text_dir, char_string)
```

**Input Parameters**

```c
const Ppoint3 *text_pos
    Text position in MC.
const Pvec3 text_dir[2]
    Text direction vectors in MC.
const char *char_string
    Character string to be displayed.
```
FORTRAN

PTX3 \((px, py, pz, tdx, tdy, tdz, chars)\)

**Input Parameters**

- **real px**
  - \(x\) coordinate of text position in MC.
- **real py**
  - \(y\) coordinate of text position in MC.
- **real pz**
  - \(z\) coordinate of text position in MC.
- **real tdx(2)**
  - \(x\) coordinates of the text direction vectors in MC.
- **real tdy(2)**
  - \(y\) coordinates of the text direction vectors in MC.
- **real tdz(2)**
  - \(z\) coordinates of the text direction vectors in MC.
- **character*(*) chars**
  - Character string to be displayed.

**FORTRAN Subset**

PTX3S \((px, py, pzdx, tdy, tdz, lstr, chars)\)

**Input Parameters**

- **real px**
  - \(x\) coordinate of text position in MC.
- **real py**
  - \(y\) coordinate of text position in MC.
- **real pz**
  - \(z\) coordinate of text position in MC.
- **real tdx(2)**
  - \(x\) coordinates of the text direction vectors in MC.
- **real tdy(2)**
  - \(y\) coordinates of the text direction vectors in MC.
- **real tdz(2)**
  - \(z\) coordinates of the text direction vectors in MC.
- **integer lstr**
  - Length of text string in bytes \((\geq 0)\).
- **character*80 chars**
  - Character string to be displayed.

**Errors**

5 Function Requires State \((PHOP,*,STOP,*)\)

**Related Subroutines**

- Inquire Text Facilities

34 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
- Set Character Expansion Factor
- Set Character Height
- Set Character Spacing
- Set Character Up Vector
- Set Text Alignment
- Set Text Color Index
- Set Text Font
- Set Text Index
- Set Text Path
- Set Text Precision
Chapter 4. Attribute Specification

Attribute values describe the appearance of output primitives, including size, shape, style, and color.

This group of subroutines creates structure elements and requires that the structure state is Structure Open (STOP). When the graPHIGS API encounters the elements in this section at structure traversal time, it modifies the current traversal time registers.

Your application can specify some attribute values directly through a structure element or indirectly by using an index to a bundle table in the Workstation State List (WSL). During structure traversal, the current Attribute Source Flag (ASF) setting determines whether the graPHIGS API draws a primitive using an individual or bundled value of an attribute. For a complete discussion of attributes, see The graPHIGS Programming Interface: Understanding Concepts.

For attribute values supported on a specific workstation, use the appropriate Inquiry programming subroutines or see The graPHIGS Programming Interface: Technical Reference.

ADD NAMES TO SET (PHOP,*;STOP,*)

Purpose

Use Add Names to Set to insert an Add Names to Set structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with an Add Names to Set structure element, depending on the current edit mode.

During structure traversal, this structure element adds the specified class names to the current class set. The traversal default is a null name set.

Class names let an application control the eligibility of a primitive for pickability (detectability), highlighting, and invisibility by associating the primitive with a class set.

When the graPHIGS API encounters a primitive during structure traversal, the primitive belongs to the classes contained in the current class set. If the workstation does not support a specified name, then the graPHIGS API ignores the name and the name has no affect on the primitive.

Also use names to create inclusion and exclusion filters for the specified workstation. The graPHIGS API uses these filters in conjunction with the class set traversal state to determine whether pickability, highlighting, and visibility apply. The filters act independently of each other. During structure traversal, the graPHIGS API compares the current class set to the current filters.

For a complete discussion of class names and filters, see The graPHIGS Programming Interface: Understanding Concepts.

Language Bindings

C

padd_names_set(names)

Input Parameters

const Pint_list *names

Name set to be added.
FORTRAN

PADS(n, namset)

Input Parameters

integer n
   Number of names in the set.

integer namset(n)
   Name set to be added.

Errors

5   Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Inquire PHIGS Facilities
- Remove Names From Set
- Set Highlighting Filter
- Set Invisibility Filter
- Set Pick Filter

REMOVE NAMES FROM SET (PHOP,*,STOP,*)

Purpose

Use Remove Names from Set to insert a Remove Names from Set structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Remove Names from Set structure element, depending on the current edit mode.

The class set traversal state consists of a list of class names. During structure traversal, this structure element removes one or more names from the list but does not completely replace the traversal state as other attributes do.

Class names let an application control the eligibility of a primitive for pickability (detectability), highlighting, and invisibility by associating the primitive with a class set. The child structures inherit the effects of adding a class name to or removing a class name from the current class set.

When the graPHIGS API encounters a primitive during structure traversal, it uses the list of class names in the class set to determine the pickability (detectability), highlighting, and invisibility aspects. If the workstation does not support a specified name, then the graPHIGS API ignores the name and the name has no effect on the primitive.

Also use class names to create inclusion and exclusion filters for the specified workstation. The graPHIGS API uses these filters in conjunction with the class set traversal state to determine whether pickability, highlighting, and visibility apply. The filters act independently of each other. During structure traversal, the graPHIGS API compares the current class set to the current filters. When root structure traversal begins, the current class set is null.

For a complete discussion of class names and filters, see The graPHIGS Programming Interface: Understanding Concepts.

Language Bindings
C

premove_names_set (names)

Input Parameters

const Pint_list *names
    Name set to be removed.

FORTRAN

PRES (n, namset)

Input Parameters

integer n
    Number of names in the set.

integer namset(n)
    Name set to be removed.

Errors

5    Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Add Names To Set
• Inquire PHIGS Facilities

SET ANNOTATION STYLE (PHOP,*,STOP,*)

Purpose

Use Set Annotation Style to insert a Set Annotation Style structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Annotation Style structure element, depending on the current edit mode.

During structure traversal, this structure element sets the current annotation style entry in the graPHIGS traversal state list to the value specified by the annotation style parameter to render subsequent annotation text primitives. For annotation style LEAD LINE, the graPHIGS API uses the current polyline attributes to render the lead line.

The traversal default for annotation style is UNCONNECTED.

If the workstation does not support the specified annotation style or the specified style is outside the allowable range, then the annotation style defaults to UNCONNECTED.

Language Bindings

C

pset_anno_style (anno_style)

Input Parameters
**Pint anno_style**

Annotation style (1=PANNO_STYLE_UNCONNECTED, 2=PANNO_STYLE_LEAD_LINE).

**FORTRAN**

PSANS (astyle)

**Input Parameters**

`integer astyle`

Annotation style (1=PUNCON, 2=PLDLN).

**Errors**

5 Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

- Annotation Text Relative
- Annotation Text Relative 3
- Inquire Annotation Facilities

---

**SET ANNOTATION TEXT ALIGNMENT (PHOP,*,STOP,*)**

**Purpose**

Use Set Annotation Text Alignment to insert a Set Annotation Text Alignment structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Annotation Text Alignment structure element, depending on the current edit mode.

At structure traversal time, this structure element specifies the alignment the graPHIGS API uses to render all subsequent annotation text primitives.

The alignment values affect the manner in which the graPHIGS API positions the annotation text extent rectangle in relation to the text position.

The traversal default for annotation alignment is NORMAL for both horizontal and vertical alignment.

For more information concerning annotation text and annotation text attributes, see The graPHIGS Programming Interface: Understanding Concepts.

**Language Bindings**

**C**

`pset_anno_align (text_align)`

**Input Parameters**

`const Ptext_align *text_align`

Annotation text alignment.

**FORTRAN**

PSATAL (atalh, atalv,)
Input Parameters

integer atalh
   Horizontal annotation text alignment (0=PAHNOR, 1=PALEFT, 2=PACENT, 3=PARITE).

integer ataly
   Vertical annotation text alignment (0=PAVNOR, 1=PATOP, 2=PACAP, 3=PAHALF, 4=PABASE, 5=PABOTT).

Errors

5   Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Text Precision

SET ANNOTATION TEXT CHARACTER HEIGHT (PHOP,*,STOP,*)

Purpose

Use Set Annotation Text Character Height to insert a Set Annotation Text Character Height structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Annotation Text Character Height structure element, depending on the current edit mode.

The application specifies the annotation text character height with respect to the annotation text local coordinate system; that is a two-dimensional coordinate system parallel to the NPC (Normalized Projection Coordinates) x-y plane. The graPHIGS API multiplies the absolute value of the specified height by the scale factor of the current workstation transformation and then maps the result to the closest available height on the workstation.

The traversal default value for annotation text character height is 0.01.

For more information concerning annotation text and annotation text attributes, see The graPHIGS Programming Interface: Understanding Concepts.

Language Bindings

C

pset_anno_char_ht (char_ht)

Input Parameters

Pfloat char_ht
   Annotation text character height.

FORTRAN

PSATCH (atchh)

Input Parameters

real atchh
   Annotation text character height.

Errors
Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Inquire Annotation Facilities

SET ANNOTATION TEXT CHARACTER UP VECTOR (PHOP,*,STOP,*)

Purpose

Use Set Annotation Text Character Up Vector to insert a Set Annotation Text Character Up Vector structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Annotation Text Character Up Vector structure element, depending on the current edit mode.

At structure traversal time, this structure element specifies the y-axis direction of the text coordinate system for characters in a text string that the graPHIGS API uses to render all subsequent annotation text primitives. When rendering annotation text primitives, the graPHIGS API uses the annotation up vector along with a default annotation base vector set at right angles in the clockwise direction to the annotation up vector.

The traversal default value for annotation up vector is 0.0, 1.0 and for annotation base vector the traversal default value is 1.0, 0.0.

If the annotation up vector is invalid, then the vector value defaults to a value of 0.0, 1.0, and a base vector value of 1.0, 0.0.

The graPHIGS API normalizes the specified vector. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vector, not the original vector specified by this subroutine.

For more information concerning annotation text and annotation text attributes, see The graPHIGS Programming Interface: Understanding Concepts.

Language Bindings

C

pset_anno_char_up_vec (char_up_vec)

Input Parameters

const Pvec *char_up_vec
  Annotation text character up vector.

FORTRAN

PSATCU (atchux, atchuy)

Input Parameters

real atchux
  x offset of the annotation text character up vector.

real atchuy
  y offset of the annotation text character up vector.
Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Set Text Precision

SET ANNOTATION TEXT PATH (PHOP,*,STOP,*)

Purpose
Use Set Annotation Text Path to insert a Set Annotation Text Path structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Annotation Text Path structure element, depending on the current edit mode.

This structure element specifies the writing direction of characters in a text string relative to the Annotation Up Vector. At structure traversal time, the graPHIGS API uses this path value to render all subsequent annotation text primitives.

The traversal default for annotation path is \textit{RIGHT}.

If the workstation does not support the specified path value or the specified value is outside the allowable range, then the annotation path value defaults to \textit{RIGHT}.

For more information concerning annotation text and annotation text attributes, see 	extit{The graPHIGS Programming Interface: Understanding Concepts}.

Language Bindings

C
\begin{verbatim}
pset_anno_path (text_path)
\end{verbatim}

Input Parameters
\begin{verbatim}
Pt_text_path text_path
\end{verbatim}
Annotation text path (0=PATH_RIGHT, 1=PATH_LEFT, 2=PATH_UP, 3=PATH_DOWN).

FORTRAN
\begin{verbatim}
PSATP (atp)
\end{verbatim}

Input Parameters
\begin{verbatim}
integer atp
\end{verbatim}
Annotation text path (0=PRIGHT, 1=PLEFT, 2=UP, 3=DOWN).

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Set Text Precision
SET CHARACTER EXPANSION FACTOR (PHOP,*,STOP,*)

Purpose

Use Set Character Expansion Factor to insert a Set Character Expansion Factor structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Character Expansion Factor structure element, depending on the current edit mode.

At structure traversal time, this structure element specifies the character expansion factor that the graPHIGS API uses to render all subsequent text primitives when the character expansion aspect source flag value is set to INDIVIDUAL (SET INDIVIDUAL ASF (PHOP,*,STOP,*)).

The value is a fraction of the width/height ratio that the font designer specified. A value of 1.0 reproduces the font designer’s width/height ratio.

The traversal default value for character expansion factor is 1.0.

When the graPHIGS API encounters an element of this type, it uses the absolute value of the specified character expansion factor. If the workstation does not support a continuous range of character expansion factors, then the graPHIGS API uses the closest supported value.

Language Bindings

C

pset_char_expan (char_expan)

Input Parameters

Pfloat char_expan

Character expansion factor.

FORTRAN

PSCHXP (chxp)

Input Parameters

real chxp

Character expansion factor.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Inquire Annotation Facilities
• Inquire Predefined Text Representation
• Inquire Text Facilities
• Set Individual ASF
• Set Text Precision

SET CHARACTER HEIGHT (PHOP,*,STOP,*)

Purpose

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Use Set Character Height to insert a Set Character Height structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Character Height structure element, depending on the current edit mode.

This structure element specifies the character height in Modeling Coordinate (MC) space that the graPHIGS API uses when rendering subsequent geometric text primitives.

The traversal default value for character height is 0.01.

When the graPHIGS API encounters an element of this type, it uses the absolute value of the specified character height. If the workstation does not support a continuous range of character heights, then the graPHIGS API uses the closest supported value.

**Language Bindings**

**C**

`pset_char_ht (char_ht)`

**Input Parameters**

`Pfloat char_ht`

Character height.

**FORTRAN**

`PSCHH (chh)`

**Input Parameters**

`real chh`

Character height.

**Errors**

5 Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

- Inquire Text Facilities

---

### SET CHARACTER SPACING (PHOP,*,STOP,*)

**Purpose**

Use Set Character Spacing to insert a Set Character Spacing structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Character Spacing structure element, depending on the current edit mode.

At structure traversal time, this structure element specifies the additional amount of space that the graPHIGS API inserts between characters to render all subsequent text primitives when the character spacing aspect source flag value is set to *INDIVIDUAL* (Set Individual ASF).

This value is expressed as a fraction of the height.

The traversal default value for character spacing is 0.0.
Language Bindings

C

pset_char_space (char_space)

Input Parameters

Pfloat char_space
Character spacing.

FORTRAN

PSCHSP (chsp)

Input Parameters

real chsp
Character spacing.

Errors

5 Function Requires State (PHOP,*;STOP,*)

Related Subroutines

- Inquire Predefined Text Representation
- Set Individual ASF
- Set Text Precision

SET CHARACTER UP VECTOR (PHOP,*;STOP,*)

Purpose

Use Set Character Up Vector to insert a Set Character Up Vector structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Character Up Vector structure element, depending on the current edit mode.

During structure traversal, this structure element sets the current character up vector in the graPHIGS traversal state list to the specified value. The base vector entry is reset to the vector that is obtained by rotating the up vector 90 degrees clockwise.

The character up vector specifies the direction of the font coordinate y-axis within the text reference coordinate system. The character base vector specifies the direction of the font coordinate x-axis within the text reference coordinate system.

At structure traversal time, this structure element specifies the y-axis direction of the text coordinate system for characters in a text string that the graPHIGS API uses to render all subsequent geometric text primitives. The character up vector is a two-dimensional vector on the text plane specified by the text primitive. When rendering text primitives, the graPHIGS API uses the character up value along with a default annotation base vector set at right angles in the clockwise direction to the character up value.

The traversal default value for character up vector is 0.0, 1.0 and the traversal default value for character base vector is 1.0, 0.0.
If the character up vector is invalid, then the up vector value defaults to 0.0, 1.0 and the base vector value defaults to 1.0, 0.0.

The graPHIGS API normalizes the specified vector. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vector, not the original vector specified by this subroutine.

Language Bindings

C


Input Parameters

const Pvec *char_up_vec
Character up vector.

FORTRAN

PSCHUP (chux, chuy)

Input Parameters

real chux
x offset of character up vector.

real chuy
y offset of character up vector.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

* Set Text Precision

SET EDGE COLOR INDEX (PHOP,*,STOP,*)

Purpose

Use Set Edge Color Index to insert a Set Edge Color Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Edge Color Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s rendering color table that defines the color the graPHIGS API will use to render the edges of all output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this index to render the edges of output primitives when the edge color aspect source flag value is set to INDIVIDUAL (Set Individual ASF) and the edge flag is set to ON (Set Edge Flag).

The traversal default for edge color is a color index value of 1.

If the workstation does not support the specified color index value or the specified index is outside the color table limit, then the color index defaults to a value of 1.
Language Bindings

C

\texttt{pset\_edge\_colr\_ind (edge\_colr\_ind)}

Input Parameters

\textit{Pint edge\_colr\_ind}

Edge color index.

FORTRAN

\texttt{PSEDCI (coli)}

Input Parameters

\textit{integer coli}

Edge color index.

Errors

5 Function Requires State (PHOP,*,STOP,* )

113 Color Index Value $< \text{ZERO}$

Related Subroutines

\begin{itemize}
  \item Inquire Edge Representation
  \item Inquire Predefined Edge Representation
  \item \texttt{Set Individual ASF}
\end{itemize}

\underline{SET EDGE FLAG (PHOP,*,STOP,* )}

Purpose

Use \textit{Set Edge Flag} to insert a \textit{Set Edge Flag} structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a \textit{Set Edge Flag} structure element, depending on the current edit mode.

This structure element indicates whether or not the graPHIGS API draws the edge of subsequent polygon primitives during structure traversal. The graPHIGS API uses the specified value if the aspect source flag value is set to \textit{INDIVIDUAL} (Set Individual ASF).

The traversal default for edge flag is \textit{OFF}.

If the workstation does not support the specified edge flag value or if the specified value is outside the allowable range, then the edge flag defaults to \textit{OFF}.

Language Bindings

C

\texttt{pset\_edge\_flag (edge\_flag)}

Input Parameters
Pedge_flag edge_flag
Edge flag (0=PEDGE_OFF, 1=PEDGE_ON).

FORTRAN
PSEDFG (edflag)

Input Parameters
integer edflag
Edge flag (0=OFF, 1=ON).

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Inquire Edge Facilities
• Inquire Predefined Edge Representation
• Set Individual ASF

SET EDGE INDEX (PHOP,*,STOP,*)

Purpose
Use Set Edge Index to insert a Set Edge Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Edge Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation's edge bundle table. The entry contains attribute settings for edge flag, edge line type, edge scale factor, and edge color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent fill area set primitives for those attributes which have an aspect source flag value set to BUNDLED (Set Individual ASF).

The traversal default value for edge index is 1.

If the workstation does not support the specified index or the specified index is outside the edge table size, then the edge index defaults to a value of 1.

Language Bindings

C
pset_edge_ind (edge_ind)

Input Parameters
Pint edge_ind
Edge index (>=1).

FORTRAN
PSEDI (edi)

Input Parameters
integer edi
   Edge index (≥ 1).

Errors
5   Function Requires State (PHOP,* ,STOP,*)
100  Bundle Index Value Is Less Than One

Related Subroutines
  • Inquire Edge Representation
  • Inquire Workstation State Table Lengths
  • Set edge Representation

SET EDGETYPE (PHOP,* ,STOP,* )

Purpose
Use Set Edgetype to insert a Set Edgetype structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Edgetype structure element, depending on the current edit mode.

This structure element specifies an index into a workstation line type table that contains line types. The graPHIGS API uses this index to render the edges of all subsequent output primitives if the corresponding edge flag is set to ON (Set Edge Flag). At structure traversal time, the graPHIGS API uses this line type to render the edges of output primitives when the line type of an edge aspect source flag is set to INDIVIDUAL (Set Individual ASF). Possible edge line types include: 1=SOLID, 2=DASHED, 3=DOTTED, and 4=DASHED-DOTTED.

The traversal default for edge type is SOLID.

If the workstation does not support the specified index or the specified index is outside the allowable range, then the edge index defaults to SOLID (edgetype).

Language Bindings

C

pset_edgetype (edgetype)

Input Parameters

Pint edgetype
   Edge type (1=PLINE_SOLID, 2=PLINE_DASH, 3=PLINE_DOT, 4=PLINE_DASH_DOT).

FORTRAN

PSEDT (edtype)

Input Parameters

integer edtype
   Edge type (1=PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Inquire Edge Representation
- Inquire Predefined Edge Representation
- Set edge Representation
- Set Individual ASF

SET EDGEWIDTH SCALE FACTOR (PHOP,*,STOP,*)

Purpose

Use Set Edgewidth Scale Factor to insert a Set Edgewidth Scale Factor structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Edgewidth Scale Factor structure element, depending on the current edit mode.

This structure element specifies a value that the graPHIGS API uses to determine how wide to draw the edges of subsequent output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this scale factor to determine the width of the edge when the edgewidth scale factor aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The edge scale factor element specifies the edge's width as a fraction of the nominal edgewidth. The device support multiplies this scale factor times the nominal width of a line on the corresponding device to determine the requested width. The graPHIGS API maps the calculated value to the closest width available on the device. A scale factor value of 1.0, which is the traversal default, generates a nominal size line on any workstation.

Language Bindings

C

pset_edgewidth (edgewidth)

Input Parameters

float edgewidth
   Edgewidth scale factor.

FORTRAN

PSEWSC (ewidth)

Input Parameters

real ewidth
   Edgewidth scale factor.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Set Edge Flag
- Inquire Edge Representation
· Inquire Predefined Edge Representation
· Set edge Representation
· Set Individual ASF

SET HLHSR IDENTIFIER (PHOP,*,STOP,*)

Purpose

Use Set HLHSR Identifier to insert a Set HLHSR Identifier structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set HLHSR Identifier structure element, depending on the current edit mode. During structure traversal, the graPHIGS API sets the current Hidden Line/Hidden Surface Removal (HLHSR) identifier entry of the graPHIGS traversal state list to the specified parameter.

The application uses this value when creating subsequent output primitives in a view with a HLHSR mode other than OFF.

If the workstation does not support the specified HLHSR identifier or the specified identifier is outside the allowable range, then the HLHSR identifier defaults to a value of 0. In a view with HLHSR mode set to OFF, this value is ignored and has no affect on the visualization of primitives.

HLHSR processing is often implemented by use of a z-buffer and a frame buffer. The following table summarizes the effect of the various HLHSR identifiers on the z-buffer and the frame buffer:

Table 1. HLHSR Processing: Summary of when the frame buffer and the z-buffer are updated.

<table>
<thead>
<tr>
<th>HLHSR Identifier</th>
<th>z-buffer</th>
<th>Frame buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Z_{prim} \geq Z_{buf}</td>
<td>Z_{prim} \geq Z_{buf}</td>
</tr>
<tr>
<td>1</td>
<td>Z_{prim} &lt; Z_{buf}</td>
<td>Never</td>
</tr>
<tr>
<td>2</td>
<td>Always</td>
<td>Always</td>
</tr>
<tr>
<td>3</td>
<td>Never</td>
<td>Z_{prim} \geq Z_{buf}</td>
</tr>
<tr>
<td>4</td>
<td>Z_{prim} \geq Z_{buf}</td>
<td>Z_{prim} \geq Z_{buf}</td>
</tr>
<tr>
<td>5</td>
<td>Front-facing Areas</td>
<td>Back-facing Areas</td>
</tr>
<tr>
<td>6</td>
<td>Z_{prim} &gt; Z_{buf}</td>
<td>Never</td>
</tr>
<tr>
<td>7</td>
<td>Z_{prim} = Z_{buf}</td>
<td>Z_{prim} = Z_{buf}</td>
</tr>
<tr>
<td>8</td>
<td>Less than</td>
<td>Z_{prim} \geq Z_{buf}</td>
</tr>
<tr>
<td>9</td>
<td>Z_{prim} &lt; Z_{buf}</td>
<td>Z_{prim} &lt; Z_{buf}</td>
</tr>
<tr>
<td>10</td>
<td>Z_{prim} \leq Z_{buf}</td>
<td>Z_{prim} \leq Z_{buf}</td>
</tr>
</tbody>
</table>

Note: The actual update of the z-buffer and/or the frame buffer may be prohibited by the use of the z-buffer protect mask and the frame buffer protect mask.

Language Bindings

C

pset_hlhsr_id (hlhsr_id)

Input Parameters

Pint hlhsr_id

HLHSR identifier.
FORTRAN

PSHRID(hrid)

Input Parameters

integer hrid
    HLHSR identifier.

Errors

5    Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Inquire HLHSR Identifier Facilities
• Inquire HLHSR Mode Facilities
• Set HLHSR Mode

SET INDIVIDUAL ASF (PHOP,*,STOP,*

Purpose

Use Set Individual ASF to insert a Set Individual ASF (Attribute Source Flag) structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Individual ASF structure element, depending on the current edit mode.

At structure traversal time, the current ASF setting determines the BUNDLED or INDIVIDUAL attributes that the graPHIGS API uses to draw an output primitive.

The traversal default for all attributes is INDIVIDUAL.

Attribute identifiers are:

<table>
<thead>
<tr>
<th>Linetype</th>
<th>Character spacing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linewidth scale factor</td>
<td>Text color index</td>
</tr>
<tr>
<td>Polyl ine color index</td>
<td>Interior style</td>
</tr>
<tr>
<td>Marker type</td>
<td>Interior style index</td>
</tr>
<tr>
<td>Marker size scale factor</td>
<td>Interior color index</td>
</tr>
<tr>
<td>Polymarker color index</td>
<td>Edge flag</td>
</tr>
<tr>
<td>Text font</td>
<td>Edge linetype</td>
</tr>
<tr>
<td>Text precision</td>
<td>Edgew idth scale factor</td>
</tr>
<tr>
<td>Character expansion factor</td>
<td>Edge color index</td>
</tr>
</tbody>
</table>

If any attribute identifier in the list is invalid, then the graPHIGS API ignores that entry. If any attribute source flag is invalid, then it defaults to INDIVIDUAL.

Language Bindings

C

pset_indiv_asf (asf_id, asf_source)

Input Parameters
Set Interior Color Index (PHOP,*,STOP,*)

Purpose

Use Set Interior Color Index to insert a Set Interior Color Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Interior Color Index structure element, depending on the current edit mode. This structure element specifies an entry in the workstation’s rendering color table. The graPHIGS API uses the color values contained in the color table to fill all subsequent area defining primitives if the interior style is set to HOLLOW (Set Interior Style) and edge is OFF (Set Edge Flag) or the interior style is set to SOLID or HATCH.

At structure traversal time, the graPHIGS API uses this index to render the interiors when the interior color aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default for interior color is a color index value of 1.

If the workstation does not support the specified interior color index or the specified index is outside the color table limit, then the interior color index defaults to a value of 1.

Language Bindings

C

pset_int_colr_ind (int_colr_ind)
Input Parameters

Pint int_colr_ind
   Interior color index (>=0).

FORTRAN

PSICI (coli)

Input Parameters

integer coli
   Interior color index (>=0).

Errors

5   Function Requires State (PHOP,*,STOP,*)
113  Color Index Value < ZERO

Related Subroutines

• Inquire Workstation State Table Lengths
• Inquire Interior Representation
• Set Interior Representation

SET INTERIOR INDEX (PHOP,*,STOP,*)

Purpose

Use Set Interior Index to insert a Set Interior Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Interior Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s interior bundle table. The entry contains attribute settings for interior style, interior style index, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent fill area and fill area set primitives for those attributes when the aspect source flag is set to BUNDELED (Set Individual ASF).

The traversal default value for interior index is 1.

If the workstation does not support the specified interior index or the specified index is outside the interior table size, then the interior index defaults to a value of 1.

Language Bindings

C

pset_int_ind (int_ind)

Input Parameters

Pint int_ind
   Interior index (>=1).
FORTRAN
PSII (ii)

Input Parameters

integer ii
    Interior index (>=1).

Errors

5    Function Requires State (PHOP,*,STOP,*)
100  Bundle Index Value Is Less Than One

Related Subroutines

• Inquire Interior Representation
• Inquire Workstation State Table Lengths
• Set Interior Representation

SET INTERIOR STYLE (PHOP,*,STOP,*)

Purpose

Use Set Interior Style to insert a Set Interior Style structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Interior Style structure element, depending on the current edit mode.

At structure traversal time, this structure element specifies the way the graPHIGS API draws the interior of a polygon when rendering all subsequent fill area and fill area set primitives. At structure traversal time, the graPHIGS API uses this style value when the interior style aspect source flag is set to INDIVIDUAL (Set Individual ASF). Transformations do not affect interior styles of HATCH and PATTERN. For interior style HOLLOW, the bounding polyline is SOLID and the line width is the nominal line width for the workstation.

The traversal default for interior style is HOLLOW.

If the workstation does not support the specified interior style value or the specified value is outside the allowable range, then the interior style defaults to HOLLOW.

Language Bindings

C

pset_int_style (int_style)

Input Parameters

Pint_style int_style
    Interior style (0=PSTYLE_HOLLOW, 1=PSTYLE_SOLID, 2=PSTYLE_PAT, 3=PSTYLE_HATCH, 4=PSTYLE_EMPTY).

FORTRAN
PSIS (ints)
Input Parameters

integer ints
    Interior style (0=PHOLLO, 1=PSOLID, 2=PPATTR, 3=PHATCH, 4=PISEMP).

Errors

5       Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Inquire Interior Facilities
- Inquire Interior Representation
- Inquire Predefined Interior Representation
- Set Individual ASF
- Set Interior Style Index

SET INTERIOR STYLE INDEX (PHOP,*,STOP,*)

Purpose

Use Set Interior Style Index to insert a Set Interior Style Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Interior Style Index structure element, depending on the current edit mode.

This structure element specifies an index into the workstation hatch table if the current interior style is HATCH (Set Interior Style) or it specifies an index into the workstation pattern table if the current interior style is PATTERN. At structure traversal time, the graPHIGS API uses this value when rendering all subsequent area defining primitives when the interior style index aspect source flag is set to INDIVIDUAL. (Set Individual ASF).

If the current interior style is not PATTERN or HATCH, then the graPHIGS API ignores this structure element and increments the element counter.

The traversal default value for interior style index is 1.

If the workstation does not support the specified interior style index value or the specified index is outside the table limit, then the interior style index defaults to a value of 1.

All workstations have available registered hatch styles 1-6.
- Horizontal equally spaced parallel lines
- Vertical equally spaced parallel lines
- Positive slope equally spaced parallel lines
- Negative slope equally spaced parallel lines
- Horizontal/vertical crosshatch
- Positive slope/negative slope crosshatch. Transformations do not affect interior styles HATCH and PATTERN.

Language Bindings

C

pset_int_style_ind (int_style_ind)

Input Parameters
\textit{Pint int_style_ind}

Interior style index.

\textbf{FORTRAN}

\textbf{PSISI} (istyl)

\textbf{Input Parameters}

\textit{integer istyl}

Interior style index.

\textbf{Errors}

5 Function Requires State (PHOP,*;STOP,*).

\textbf{Related Subroutines}

- Inquire Interior Facilities
- Inquire Interior Representation
- Inquire Predefined Interior Representation
- Set Individual ASF
- Set Interior Representation
- Set Interior Style

\textbf{SET LINETYPE (PHOP,*,STOP,*)}

\textbf{Purpose}

Use \textit{Set Linetype} to insert a Set Linetype structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Linetype structure element, depending on the current edit mode.

This structure element specifies an index into a workstation line type table that contains line types. The graPHIGS API uses this index to render all subsequent output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this line type to render the output primitives when the line type aspect source flag is set to \textit{INDIVIDUAL} (Set Individual ASF).

The traversal default for line type is \textit{SOLID}.

If the workstation does not support the specified linetype entry or the specified entry is outside the allowable range, then the linetype defaults to \textit{SOLID}.

\textbf{Language Bindings}

\textbf{C}

\textit{pset_linetype (linetype)}

\textbf{Input Parameters}

\textit{Pint linetype}

Line type (1=\textit{PLINE_SOLID}, 2=\textit{PLINE_DASH}, 3=\textit{PLINE_DOT}, 4=\textit{PLINE_DASH_DOT}).
FORTRAN

PSLN (ltype)

Input Parameters

integer ltype
    Line type (1=PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).

Errors

5 Function Requires State (PHOP,*,STOP,*

Related Subroutines

• Inquire Polyline Facilities
• Inquire Polyline Representation
• Set Individual ASF
• Set Polyline Representation

SET LINEWIDTH SCALE FACTOR (PHOP,*,STOP,*)

Purpose

Use Set Linewidth Scale Factor to insert a Set Linewidth Scale Factor structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Linewidth Scale Factor structure element, depending on the current edit mode.

The Linewidth Scale Factor specifies the width of the line as a fraction of the nominal. The device support multiplies this scale factor by the nominal line width on the corresponding device to determine the requested width. The graPHIGS API maps the calculated value to the closest width available on the device. A scale factor value of 1.0 generates a nominal size line on any workstation. At structure traversal time, the graPHIGS API uses this scale factor when the line width scale factor aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default value for linewidth scale factor is 1.0.

Language Bindings

C

pset_linewidth (linewidth)

Input Parameters

Pfloat linewidth
    Line width scale factor.

FORTRAN

PSLWSC (lwidth)

Input Parameters

real lwidth
    Line width scale factor.
Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Inquire Polyline Facilities
- Inquire Polyline Representation
- Set Individual ASF
- Set Polyline Representation

SET MARKER SIZE SCALE FACTOR (PHOP,*,STOP,*)

Purpose
Use Set Marker Size Scale Factor to insert a Set Marker Size Scale Factor structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Marker Size Scale Factor structure element, depending on the current edit mode.

This structure element specifies the marker’s size as a fraction of the nominal marker size. The device support multiplies this scale factor by the nominal size of markers on the corresponding device to determine the requested size. The graPHIGS API maps the calculated value to the closest size available on the device. A scale factor value of 1.0 generates a nominal size marker on any workstation. At structure traversal time, the graPHIGS API uses this marker size scale factor when the marker size scale factor aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default value for marker size scale factor is 1.0.

Language Bindings

C

pset_marker_size (marker_size)

Input Parameters

Pfloat marker_size
Marker size scale factor.

FORTRAN

PSMKSC (mszsf)

Input Parameters

real mszsf
Marker size scale factor.

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Inquire Polymarker Facilities
- Inquire Polymarker Representation
- Set Individual ASF
• Set Polyline Representation

SET MARKER TYPE (PHOP,*,STOP,*)

Purpose

Use Set Marker Type to insert a Set Marker Type structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Marker Type structure element, depending on the current edit mode.

This structure element specifies an index into a workstation markertype table that contains marker types that the graPHIGS API uses to render all subsequent polymarker primitives. At structure traversal time, the graPHIGS API uses this marker type to render the polymarker primitives when the marker type aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default for marker type is ASTERISK.

If the workstation does not support the specified marker type entry or the specified entry is outside the allowable range, then the marker type entry defaults to ASTERISK (marker).

Language Bindings

C

pset_marker_type (marker_type)

Input Parameters

Pint marker_type

  Marker type (1=PMARKER_DOT, 2=PMARKER_PLUS, 3=PMARKER_ASTERISK, 4=PMARKER_CIRCLE, 4=PMARKER_CROSS).

FORTRAN

PSMK (mtype)

Input Parameters

integer mtype

  Marker type (1=PPOINT, 2=PPLUS, 3=PAST, 4=POMARK, 5=PXMARK).

Errors

5  Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Inquire Polymarker Facilities
• Inquire Polymarker Representation
• Set Individual ASF
• Set Polyline Representation

SET PATTERN REFERENCE POINT (PHOP,*,STOP,*)

Purpose
Use Set Pattern Reference Point to insert a Set Pattern Reference Point structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Pattern Reference Point structure element, depending on the current edit mode.

This structure element specifies a two-dimensional pattern reference point. The z coordinate is assumed to be zero. The pattern reference vectors are assumed to be (1,0,0) and (0,1,0). At structure traversal time, the graPHIGS API uses this pattern reference point and assumed pattern reference vectors to display fill area and fill area set primitives when the currently selected interior style is set to PATTERN (Set Interior Style).

The traversal default value for pattern reference point is (0.0,0.0,0.0). The traversal default values for pattern reference vectors are (1,0,0) and (0,1,0).

**Note:** The graPHIGS API currently ignores this structure element at structure traversal time.

### Language Bindings

#### C

```c
pset_pat_ref_point (pat_ref_point)
```

**Input Parameters**

```c
const Ppoint *pat_ref_point
```
Pattern reference point in MC.

#### FORTRAN

```fortran
PSPARF (rfx, rfy)
```

**Input Parameters**

```fortran
real rfx
```
  x coordinate of the pattern reference point in MC.

```fortran
real rfy
```
  y coordinate of the pattern reference point in MC.

#### Errors

5  Function Requires State (PHOP,*,STOP,*)

#### Related Subroutines

- Set Pattern Reference Point And Vectors
- Set Pattern Size

### SET PATTERN REFERENCE POINT AND VECTORS (PHOP,*,STOP,*)

#### Purpose

Use Set Pattern Reference Point and Vectors to insert a Set Pattern Reference Point and Vectors structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Pattern Reference Point and Vectors structure element, depending on the current edit mode.
This structure element specifies a three-dimensional pattern reference point and pattern reference vectors. At structure traversal time, the graPHIGS API uses this pattern reference point and pattern reference vectors to display fill area and fill area set primitives when the currently selected interior style is set to PATTERN (Set Interior Style). If one of the pattern reference vectors is zero in length or the vectors are parallel, then the graPHIGS API ignores this structure element.

The traversal default value for pattern reference point is (0,0,0). The traversal default values for pattern reference vectors are (1,0,0) and (0,1,0).

**Note:** The graPHIGS API currently ignores this structure element at structure traversal time.

### Language Bindings

#### C

```c
pset_pat_ref_point_vecs (pat_ref_point, pat_ref_vec)
```

**Input Parameters**

- `const Ppoint3 *pat_ref_point`: Pattern reference point in MC.

#### FORTRAN

```fortran
PSPRPV (rfx, rfy, rfz, rfvx, rfvy, rfvz)
```

**Input Parameters**

- `real rfx`: x coordinate of the pattern reference point in MC.
- `real rfy`: y coordinate of the pattern reference point in MC.
- `real rfz`: z coordinate of the pattern reference point in MC.
- `real pfvx(2)`: x coordinates of the pattern reference vectors in MC.
- `real pfvy(2)`: y coordinates of the pattern reference vectors in MC.
- `real rfvz(2)`: z coordinates of the pattern reference vectors in MC.

**Errors**

- `5` Function Requires State (PHOP,*,STOP,*).

**Related Subroutines**

- Set Pattern Reference Point
- Set Pattern Size
SET PATTERN SIZE (PHOP,*,STOP,*)

Purpose

Use Set Pattern Size to insert a Set Pattern Size element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Pattern Size structure element, depending on the current edit mode.

This structure element specifies the pattern size dimensions. At structure traversal time, the graPHIGS API uses the specified pattern size in conjunction with the pattern reference point and pattern reference vectors to display fill area and fill area set primitives when the currently selected interior style is set to PATTERN (Set Interior Style). The graPHIGS API uses only the magnitudes of the pattern size components. If either of the components is zero, then the graPHIGS API ignores this structure element.

The traversal default value for pattern size is (1.0,1.0).

Note: The graPHIGS API currently ignores this structure element at structure traversal time.

Language Bindings

C

pset_pat_size (pat_size)

Input Parameters

const Pfloat_size *pat_size
Pattern size in MC.

FORTRAN

PSPA (szx, szy)

Input Parameters

real szx
x dimension of pattern size in MC.

real szy
y dimension of pattern size in MC.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Set Pattern Reference Point
- Set Pattern Reference Point And Vectors

SET PICK IDENTIFIER (PHOP,*,STOP,*)

Purpose

Use Set Pick Identifier to insert a Set Pick Identifier structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Pick Identifier structure element, depending on the current edit mode.
The graPHIGS API associates the pick identifier with all subsequent primitives and returns the pick identifier in each entry of a pick path. The returned pick identifier represents the pick identifier that was current when the application processed the corresponding structure element.

The traversal default for pick identifier is no pick identifier.

Language Bindings

C

pset_pick_id (pick_id)

Input Parameters

Pint pick_id
Pick identifier.

FORTRAN

PSPKID (pkid)

Input Parameters

integer pkid
Pick identifier.

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• None

SET POLYLINE COLOR INDEX (PHOP,*,STOP,*)

Purpose

Use Set Polyline Color Index to insert a Set Polyline Color Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Polyline Color Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s rendering color table that defines the color values that the graPHIGS API uses to render all output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this index to render the output primitives when the polyline color aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default for polyline color is a color index value of 1.

If the workstation does not support the specified polyline color index or the specified index is outside the color table limit, then the polyline color index defaults to a value of 1.

Language Bindings

C

pset_line_colr_ind (line_colr_ind)
Input Parameters

Pint line_colr_ind
Polyline color index.

FORTRAN

PSPLCI (coli)

Input Parameters

integer coli
Polyline color index.

Errors

5  Function Requires State (PHOP,*;STOP,*)
113  Color Index Value < ZERO

Related Subroutines

• Inquire Polyline Facilities
• Inquire Polyline Representation
• Set Individual ASF
• Set Polyline Representation

SET POLYLINE INDEX (PHOP,*;STOP,*)

Purpose

Use Set Polyline Index to insert a Set Polyline Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Polyline Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s polyline bundle table. The entry contains attribute settings for line type, line width scale factor, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polyline primitives for the attributes that have an aspect source flag set to BUNDLED (Set Individual ASF).

The traversal default value for polyline index is 1.

If the workstation does not support the specified polyline index or the specified index is outside the polyline bundle table size, then the polyline index defaults to a value of 1.

Language Bindings

C

pset_line_ind (line_ind)

Input Parameters

Pint line_ind
Polyline index (>=1).
FORTRAN

PSPLI (pli)

Input Parameters

integer pli
   Polyline index (>=1).

Errors

5     Function Requires State (PHOP,*,STOP,*)
100   Bundle Index Value Is Less Than One

Related Subroutines

• Inquire Polyline Representation
• Inquire Workstation State Table Lengths
• Set Polyline Representation

SET POLYMARKER COLOR INDEX (PHOP,*,STOP,*)

Purpose

Use Set Polymarker Color Index to insert a Set Polymarker Color Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Polymarker Color Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s rendering color table that defines the color values that the graPHIGS API uses to render all polymarker primitives. At structure traversal time, the graPHIGS API uses this index to render the polymarker primitives when the polymarker color aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default for polymarker color is a color index value of 1.

If the workstation does not support the specified polymarker color or the specified index is outside the color table limit, then the polymarker color index defaults to a value of 1.

Language Bindings

C

pset_marker_colr_ind (marker_colr_ind)

Input Parameters

Pint marker_colr_ind
   Polymarker color index (>=0).

FORTRAN

PSPMCI (coli)

Input Parameters
integer coli
Polymarker color index (>=0).

Errors
5   Function Requires State (PHOP,*,STOP,*),
113  Color Index Value < ZERO

Related Subroutines
• Inquire Polymarker Facilities
• Inquire Polymarker Representation
• Set Individual ASF
• Set Polyline Representation

SET POLYMARKER INDEX (PHOP,*,STOP,*)

Purpose
Use Set Polymarker Index to insert a Set Polymarker Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Polymarker Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s polymarker bundle table. The entry contains attribute settings for marker type, marker size scale factor, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polymarker primitives for the attributes that have an aspect source flag set to BUNDLED (Set Individual ASF).

The traversal default value for polymarker index is 1.

If the workstation does not support the specified polymarker index of the specified index is outside the polymarker bundle table size, then the polymarker index defaults to a value of 1.

Language Bindings

C

pset_marker_ind (marker_ind)

Input Parameters
Pint marker_ind
   Polymarker index (>=1).

FORTRAN

PSPMI (pmi)

Input Parameters
integer pmi
   Polymarker index (>=1).

Errors
5   Function Requires State (PHOP,*,STOP,*),
Related Subroutines

- Inquire Polymarker Representation
- Inquire Workstation State Table Lengths
- Set Polyline Representation

---

### SET TEXT ALIGNMENT (PHOP,*,STOP,*)

**Purpose**

Use Set Text Alignment to insert a Set Text Alignment structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Text Alignment structure element, depending on the current edit mode.

At structure traversal time, the graPHIGS API uses the specified alignment in this structure element to render all subsequent geometric text primitives. This setting affects the manner in which the graPHIGS API positions the geometric text extent rectangle in relation to the text position.

The traversal default for geometric text horizontal and vertical alignment is `NORMAL`.

If the workstation does not support the specified text alignment or the specified value is outside the allowable range, then the text alignment defaults to `NORMAL` for both horizontal and vertical text alignment.

**Language Bindings**

**C**

```
pset_text_align (text_align)
```

**Input Parameters**

```
const Ptext_align *text_align
```

Text alignment.

**FORTRAN**

```
PSTXAL (txalh, txalv)
```

**Input Parameters**

```
integer txalh
```

Horizontal text alignment (0=PAHNor, 1=PALEFT, 2=PACENT, 3=PARITE).

```
integer txalv
```

Vertical text alignment (0=PAVNOR, 1=PATOP, 2=PACAP, 3=PAHALF, 4=PABASE, 5=PABOTT).

**Errors**

5 Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

- Set Text Precision
SET TEXT COLOR INDEX (PHOP,*,STOP,*)

Purpose

Use Set Text Color Index to insert a Set Text Color Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Text Color Index structure element, depending on the current edit mode.

This structure element specifies an entry in the workstation’s rendering color table that defines the color the graPHIGS API uses to render all subsequent annotation and geometric text primitives. At structure traversal time, the graPHIGS API uses this index to render the text primitives when the text color aspect source flag is set to INDIVIDUAL (Set Individual ASF).

The traversal default for text color is a color index value of 1.

If the workstation does not support the specified text color index or the specified index is outside the color table limit, then the text color index defaults to a value of 1.

Language Bindings

C

pset_text_colr_ind (text_colr_ind)

Input Parameters

Pint text_colr_ind
  Text color index (>=0).

FORTRAN

PSTXCI (coli)

Input Parameters

integer coli
  Text color index (>=0).

Errors

5    Function Requires State (PHOP,*,STOP,*)
113   Color Index Value < ZERO

Related Subroutines

• Inquire Predefined Text Representation
• Inquire Text Representation
• Set Individual ASF
• Set Text Representation

SET TEXT FONT (PHOP,*,STOP,*)

Purpose
Use **Set Text Font** to insert a **Set Text Font** structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a **Set Text Font** structure element, depending on the current edit mode.

At structure traversal time, the graPHIGS API uses the specified font identifier in this structure element to render all subsequent annotation and geometric text primitives when the text font aspect source flag is set to `INDIVIDUAL` (Set Individual ASF).

The traversal default for annotation and geometric text font is font 1.

The graPHIGS API implicitly makes available fonts 1 and 2 of the US English character set when an application issues an ISO PHIGS Open Workstation subroutine call (Open Workstation). For an illustration of those fonts refer to *The graPHIGS Programming Interface: Technical Reference*. In *The graPHIGS Programming Interface: Technical Reference*, the U.S. English character set is referred to as Character Set 1. This is the number associated with this character set when using the GPxxxx interface.

**Language Bindings**

**C**

```c
pset_text_font (font)
```

**Input Parameters**

`Pint font`

Text font.

**FORTRAN**

```fortran
PSTXFN (font)
```

**Input Parameters**

`integer font`

Text font.

**Errors**

5 Function Requires State (PHOP,* ,STOP,*)

**Related Subroutines**

- Inquire Predefined Text Representation
- Inquire Text Representation
- Set Individual ASF
- Set Text Representation

---

**SET TEXT INDEX (PHOP,* ,STOP,* )**

**Purpose**

Use **Set Text Index** to insert a **Set Text Index** structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a **Set Text Index** structure element, depending on the current edit mode.
This structure element specifies an entry in the workstation’s text bundle table. The entry contains attribute settings for text font, text precision, character expansion factor, character spacing, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent annotation and geometric text primitives for the attributes that have an aspect source flag set to BUNDLED (Set Individual ASF).

The traversal default value for text index is 1.

If the workstation does not support the specified text index or the specified index is outside the text bundle table size, then the text index defaults to a value of 1.

Language Bindings

C

```c
pset_text_ind (text_ind)
```

Input Parameters

Pint text_ind

Text index (>=1).

FORTRAN

PSTXI (txi)

Input Parameters

```
ext integer txi
```

Text index (>=1).

Errors

5   Function Requires State (PHOP,*,STOP,*)

100  Bundle Index Value Is Less Than One

Related Subroutines

- Inquire Predefined Text Representation
- Inquire Text Representation
- Inquire Workstation State Table Lengths
- Set Text Representation

SET TEXT PATH (PHOP,*,STOP,*)

Purpose

Use Set Text Path to insert a Set Text Path structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Text Path structure element, depending on the current edit mode.

This structure element specifies the writing direction of the text string relative to the Character Up Vector. At structure traversal time, the graPHIGS API uses this path value to render all subsequent geometric text primitives.

The traversal default for text path is RIGHT.
If the workstation does not support the specified text path or the specified value is outside the allowable range, then the text path defaults to RIGHT.

Language Bindings

C

pset_text_path (text_path)

Input Parameters

Ptext_path text_path
   Text path (0=PPATH_RIGHT, 1=PPATH_LEFT, 2=PPATH_UP, 3=PPATH_DOWN).

FORTRAN

PSTXP (txp)

Input Parameters

integer txp
   Text path (0=PRIGHT, 1=PLEFT, 2=PUP, 3=PDOWN).

Errors

5    Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Set Text Precision

SET TEXT PRECISION (PHOP,*,STOP,*)

Purpose

Use Set Text Precision to insert a Set Text Precision structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Text Precision structure element, depending on the current edit mode.

The text precision specifies which attributes apply to annotation and geometric text primitives and the manner in which the graPHIGS API uses them. At structure traversal time, the graPHIGS API uses this precision when the text precision aspect source flag is set to INDIVIDUAL (Set Individual ASF).

Text precisions available are STRING, CHARACTER, and STROKE. The traversal default for text precision is STRING.

If the workstation does not support the specified text precision, then the graPHIGS API uses the highest available precision instead.
The following figure describes the attributes and precision for geometric text:

<table>
<thead>
<tr>
<th>Geometric Text</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEXT Font</td>
<td></td>
</tr>
<tr>
<td>CHARACTER EXPANSION FACTOR</td>
<td></td>
</tr>
<tr>
<td>CHARACTER SPACING</td>
<td></td>
</tr>
<tr>
<td>COLOR</td>
<td></td>
</tr>
<tr>
<td>CHARACTER HEIGHT</td>
<td></td>
</tr>
<tr>
<td>CHARACTER UP VECTOR</td>
<td></td>
</tr>
<tr>
<td>TEXT PATH</td>
<td></td>
</tr>
<tr>
<td>TEXT ALIGNMENT</td>
<td></td>
</tr>
<tr>
<td>CHARACTER UP AND BASE VECTORS</td>
<td></td>
</tr>
</tbody>
</table>

r

c

<table>
<thead>
<tr>
<th>STRING</th>
<th>Y **</th>
<th>N</th>
<th>N</th>
<th>Y</th>
<th>Y</th>
<th>Y</th>
<th>N</th>
<th>N</th>
<th>N</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARACTER</td>
<td>Y **</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>STROKE</td>
<td>Y **</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

The following keywords are used above to designate which attributes will be processed for a particular precision:

Y - The attribute is applied for this precision.
N - The attribute is not applied for this precision.
** - The requested font will be applied if it is available on the requested workstation. Otherwise, the workstation will default to an alternate font.

The following figure describes the attributes and precision for annotation text:

<table>
<thead>
<tr>
<th>Annotation Text</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEXT Font</td>
<td></td>
</tr>
<tr>
<td>CHARACTER EXPANSION FACTOR</td>
<td></td>
</tr>
<tr>
<td>CHARACTER SPACING</td>
<td></td>
</tr>
<tr>
<td>COLOR</td>
<td></td>
</tr>
<tr>
<td>ANNOTATION HEIGHT SCALE FACTOR</td>
<td></td>
</tr>
<tr>
<td>ANNOTATION UP VECTOR</td>
<td></td>
</tr>
<tr>
<td>ANNOTATION PATH</td>
<td></td>
</tr>
<tr>
<td>ANNOTATION ALIGNMENT</td>
<td></td>
</tr>
<tr>
<td>ANNOTATION HEIGHT</td>
<td></td>
</tr>
</tbody>
</table>

r

c

<table>
<thead>
<tr>
<th>STRING</th>
<th>Y 4</th>
<th>N</th>
<th>N</th>
<th>Y</th>
<th>Y</th>
<th>Y 1</th>
<th>N</th>
<th>N</th>
<th>N</th>
<th>N</th>
<th>Y 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARACTER</td>
<td>Y 4</td>
<td>Y 1</td>
<td>Y 2</td>
<td>Y</td>
<td>Y 1</td>
<td>Y 2</td>
<td>Y 2</td>
<td>Y 2</td>
<td>Y 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STROKE</td>
<td>Y 4</td>
<td>Y 3</td>
<td>Y 3</td>
<td>Y</td>
<td>Y 3</td>
<td>Y 3</td>
<td>Y 3</td>
<td>Y 3</td>
<td>Y 3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following keywords are used above to designate which attributes will be processed for a particular precision:

Y - The attribute is applied for this precision.
N - The attribute is not applied for this precision.

The following keywords are used above to designate which attributes will be processed for a particular precision:

Y - The attribute is applied for this precision.
N - The attribute is not applied for this precision.

The following numbers are used above to describe how precisely an attribute will be applied:

1 - The attribute is applied as closely as possible for the entire text string.
2 - Whether these attributes are applied is workstation dependent. See The graPHIGS Programming Interface: Technical Reference for more information.
3 - The attribute is applied on a stroke-by-stroke basis, that is, exactly.
The requested font will be applied if the font is available in the requested workstation; otherwise, the workstation will default to an alternate font.

Language Bindings

C

`pset_text_prec (prec)`

Input Parameters

`Ptext_prec prec`

Text precision (0=PPREC_STRING, 1=PPREC_CHAR, 2=PPREC_STROKE).

FORTRAN

`PSTXPR (prec)`

Input Parameters

`integer prec`

Text precision (0=PSTRP, 1=PCHARP, 2=PSTRKP).

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Inquire Annotation Facilities
- Inquire Predefined Text Representation
- Inquire Text Facilities
- Inquire Text Representation
- Set Individual ASF
- Set Text Representation

SET VIEW INDEX (PHOP,*,STOP,*)

Purpose

Use Set View Index to insert a Set View Index structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set View Index structure element, depending on the current edit mode.

This structure elements specifies an entry in the workstation’s text bundle table. The entry contains attribute settings for the view orientation matrix, view mapping matrix, viewport boundaries, and viewport clipping indicators. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent output primitives.

The traversal default for the view index is entry 0 of the workstation’s view table.

Language Bindings
C

`pset_view_ind (view_ind)`

Input Parameters

`Pint view_ind`
View index (>=0).

FORTRAN

`PSVWI (viewi)`

Input Parameters

`integer viewi`
View index (>=0).

Errors

5 Function Requires State (PHOP,*,STOP,*)
114 View Index Value < ZERO

Related Subroutines

- Set View Representation
- Set View Representation 3
Chapter 5. Miscellaneous Structure Element Subroutines

This section describes subroutines which generate structure elements which are *not* related to primitives or primitive attributes. The subroutines in this section generate elements which cause structure execution at traversal time, or elements which are used to store application specific information, or generalized structure elements.

**APPLICATION DATA (PHOP,*,STOP,*)**

**Purpose**

Use Application Data to insert an application data structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with an Insert Application Data structure element, depending on the current edit mode.

This subroutine allows the insertion of application specific data into a structure element. The graPHIGS API ignores this data during structure traversal.

**Language Bindings**

**C**

```c
pappl_data(data)
```

**Input Parameters**

```c
const Pdata *data
```

Application data.

**FORTRAN**

```fortran
PAP(ldr, datrec)
```

**Input Parameters**

```fortran
integer ldr
```

Dimension of data record array.

```fortran
character*80 datrec(ldr)
```

Data record.

**Errors**

5 Function Requires State (PHOP,*,STOP,*)

**Related Subroutines**

- None

**EXECUTE STRUCTURE (PHOP,*,STOP,*)**

**Purpose**
Use Execute Structure to insert an Execute Structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Execute Structure element, depending on the current edit mode. If the specified structure does not exist, Execute Structure creates a new empty structure.

Traversal of the structure in which the Execute Structure element exists causes invocation of the target structure as soon as the Execute Structure element is encountered. Although the graPHIGS API does not allow recursive structure networks, no error is generated by the creation of such a network. When the application attempts to execute the open structure, the graPHIGS API generates an implementation error (-125). This is a graPHIGS API restriction. The behavior of the graPHIGS API when traversing a recursive structure network is undefined.

Language Bindings

C

pexec_struct (struct_id)

Input Parameters

Pint struct_id
    Structure identifier.

FORTRAN

PEXST (strid)

Input Parameters

integer strid
    Structure identifier.

Errors

5    Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- None

GENERALIZED STRUCTURE ELEMENT (PHOP,*,STOP,*)

Purpose

Use Generalized Structure Element to insert a Generalized Structure Element (GSE) into the open structure following the element pointer or to replace the element pointed at by the element pointer with a GSE, depending on the current edit mode.

The graPHIGS API currently does not support any GSEs through this subroutine. The graPHIGS API puts any elements generated by this subroutine into the open structure but ignores them at structure traversal time. Use the appropriate GPxxxx subroutine to generate a desired GSE supported by the graPHIGS API. See The graPHIGS Programming Interface: Technical Reference, for a list of the GSEs supported by the graPHIGS API. Also, because GSE support is workstation dependent, use the graPHIGS API Inquire List of Available GSEs (GPQGSE) subroutine to determine the specific GSEs supported by an open workstation.

Language Bindings
C

gse (id, gse_data)

Input Parameters

Pint id  
   GSE identifier.
const Pgse_data *gse_data  
   GSE data record.

FORTRAN

PGSE (gseid, ldr, datrec)

Input Parameters

integer gseid  
   GSE identifier.
integer ldr  
   Dimension of data record array
character*80 datrec(ldr)  
   GSE data record.

Errors

5   Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• Inquire Generalized Structure Element Facilities
Chapter 6. Structure Operation Subroutines

The subroutines included in this section let your application program manipulate structure content. Operations performed by these subroutines include:

- creating/deleting a structure
- creating structure hierarchies
- opening a structure for modification
- editing structure content

After opening a structure, the element pointer normally points to the last element in the structure. In order to modify a structure, the application must reposition the element pointer to a desired element. If the edit mode is set for insertion, then the graPHIGS API inserts an element in the open structure following the element pointer. Otherwise, if the edit mode is set for replacement, then an element replaces the element at the current element pointer.

If you plan to use both GPxxxx and ISO PHIGS subroutine calls, be aware of the compatibility issues as outlined in Chapter 22. "graPHIGS API Extensions and Compatibility with the ISO PHIGS Standard.”

CHANGE STRUCTURE IDENTIFIER (PHOP, *, *, *)

Purpose

Use Change Structure Identifier to change the identifier of a structure (called the original structure) to a specified structure identifier (called the resulting structure). This subroutine does not affect execute structure-type elements that reference the original structure.

If the identifier of the original structure is the same as the identifier of the resulting structure and if the structure exists, then no action occurs. If the structure does not exist, however, then the graPHIGS API creates an empty structure with the identifier of the resulting structure.

If the original structure does not exist, then the graPHIGS API empties the resulting structure. If the original structure does exist, then the contents of the original structure replace the contents of the resulting structure and the graPHIGS API empties the original structure. If the original structure references the resulting structure, then the graPHIGS API generates an implementation error (-129) and no action occurs.

At the completion of this subroutine, the graPHIGS API deletes the original structure; unless, the original structure is the open structure, is referenced by any other structure, or is posted to a workstation.

If the original structure is the open structure, then the graPHIGS API sets the current element pointer to zero. If the resulting structure is the open structure, then the graPHIGS API sets the current element pointer to point to the last element in the structure.

Language Bindings

C

pchange_struct_id (orig_struct_id, result struct_id)

Input Parameters

Pint orig struct_id
   Original structure identifier.
Pint result_struct_id
   Resulting structure identifier.

FORTRAN

PCSTID (oldsid, newsid)

Input Parameters

integer oldsid
   Original structure identifier.

integer newsid
   Resulting structure identifier.

Errors

2    Function Requires State (PHOP,*,*,*

Related Subroutines

• Empty Structure
• Execute Structure
• Post Structure

CHANGE STRUCTURE IDENTIFIER AND REFERENCES (PHOP,*,*,*

Purpose

Use Change Structure Identifier and References to change all execute structure-type elements which reference a structure (called the original structure) with elements which reference a specified structure (called the resulting structure). This subroutine changes the identifier of the original structure to be that of the resulting structure. The effect of this subroutine is as though the application called the Change Structure References subroutine followed by a call to the Change Structure Identifier subroutine.

If the original structure references the resulting structure, then the graPHIGS API generates an implementation error (-129) and no action occurs. This error is generated to prevent an application from causing recursive traversal of a structure.

Language Bindings

C

pchange_struct_id.refs (orig_struct_id, result_struct_id)

Input Parameters

Pint orig_struct_id
   Original structure identifier.

Pint result_struct_id
   Resulting structure identifier.

FORTRAN

PCSTIR (oldsid, newsid)
Input Parameters

integer oldsid
   Original structure identifier.

integer newsid
   Resulting structure identifier.

Errors

2   Function Requires State (PHOP,*,*,*)

Related Subroutines

• Change Structure Identifier
• Change Structure References

CHANGE STRUCTURE REFERENCES (PHOP,*,*,*)

Purpose

Use Change Structure References to change all execute structure-type elements which reference a structure (called the original structure) with elements which reference a specified structure (called the resulting structure). This subroutine does not affect any references to the resulting structure that existed before the call.

If the identifier of the original structure and the identifier of the resulting structure are identical, then no action occurs. If the resulting structure references the original structure, then the graPHIGS API generates an implementation error (-129) and no action occurs. This error is generated to prevent an application from causing recursive traversal of a structure.

If there are references to the original structure and the resulting structure does not exist, then the graPHIGS API creates an empty structure with the identifier of the resulting structure. If the original structure does not exist or if there are no references to the original structure, then no action occurs.

If the resulting structure is posted to a workstation, then the resulting structure remains posted and the original structure, if posted to the workstation, is unposted. If the original structure is posted to a workstation but the resulting structure is not, then the graPHIGS API posts the resulting structure to that workstation and unposts the original structure from the workstation.

Language Bindings

C

pchange_struct_refs( orig_struct_id, result_struct_id)

Input Parameters

Pint orig_struct_id
   Original structure identifier.

Pint result_struct_id
   Resulting structure identifier.

FORTRAN

PCSTRF (oldsid, newsid)
Input Parameters

- **integer oldsid**: Original structure identifier.
- **integer newsid**: Resulting structure identifier.

Errors

- **2**: Function Requires State (PHOP,*,*,*)

Related Subroutines

- Execute Structure
- Post Structure
- Unpost Structure

---

**CLOSE STRUCTURE (PHOP,*,STOP,*)**

**Purpose**

Use Close Structure to close a structure. The current structure state is set to Structure Closed (STCL).

Conditional editing is stopped if it had been started by the graPHIGS API extension Conditional Editing (GPCEDT. See “Structure Operations” in *The graPHIGS Programming Interface: Subroutine Reference*).

Once closed, your application cannot insert or replace structure elements in the structure until the structure is opened.

**Language Bindings**

**C**

```c
pclose_struct()
```

**FORTRAN**

```fortran
PCLST
```

Errors

- **5**: Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Open Structure

---

**COPY ALL ELEMENTS FROM STRUCTURE (PHOP,*,STOP,*)**

**Purpose**

Use Copy All Elements From Structure to copy the elements from the specified structure into the open structure following the current element pointer. The element pointer moves to the last element copied into the open structure.

If the structure you copy references the open structure (by the use of an execute structure-type element), then the graPHIGS API issues an implementation error message (-125) and does not perform the copy.
Language Bindings

C
pcopy_all elems struct (struct_id)

Input Parameters
Pint struct_id
  Structure identifier.

FORTRAN
PCELST (strid)

Input Parameters
integer strid
  Structure identifier.

Errors
5  Function Requires State (PHOP,*,STOP,*)

Related Subroutines
  • Execute Structure

DELETE ALL STRUCTURES (PHOP,*,*,*)

Purpose
Use Delete All Structures to delete all existing structures from the structure store. This subroutine is equivalent to invoking the Delete Structure subroutine for each structure in the structure store.

Language Bindings

C
pdel_all structs()

FORTRAN
PDAS

Errors
2  Function Requires State (PHOP,*,*,*)

Related Subroutines
  • Delete Structure

DELETE ELEMENT (PHOP,*,STOP,*)

Purpose
Use Delete Element to delete the element indicated by the element pointer from the open structure. The element pointer moves to the element immediately preceding the deleted element. If the element pointer is zero, then the graPHIGS API deletes nothing and the element pointer does not move.

Language Bindings

C

pdel_elem ()

FORTRAN

PDEL

Errors

5 Function Requires State (PHOP,*;STOP,*)

Related Subroutines

• None

DELETE ELEMENT RANGE (PHOP,*;STOP,*)

Purpose

Use Delete Element Range to delete all structure elements between and including the elements indicated by the specified element numbers.

After deletion, the element pointer moves to the element immediately preceding any deleted elements. If both values point beyond the last element of the structure or both values are less than zero, then the graPHIGS API does not delete any elements and the element pointer remains the same. If one of the values is less than zero, then the element pointer defaults to a value of zero. If one of the values is greater than the number of elements in the open structure, then the graPHIGS API uses the element number of the last element in the open structure instead.

Language Bindings

C

pdel_elem_range (elem_ptr1_value, elem_ptr2_value)

Input Parameters

Pint elem_ptr1_value
  Element pointer 1 value.

Pint elem_ptr2_value
  Element pointer 2 value.

FORTRAN

PDELRA (ep1, ep2)

Input Parameters
integer ep1
Element position 1 value.

integer ep2
Element position 2 value.

Errors
5 Function Requires State (PHOP,*,STOP,*).

Related Subroutines
• None

DEDELETE ELEMENTS BETWEEN LABELS (PHOP,*,STOP,*).

Purpose
Use Delete Elements Between Labels to delete all elements from the open structure between, but not including, the two specified labels. After deletion, the element pointer moves to the element specified by the label 1 identifier.

Starting from the current position of the element pointer, the graPHIGS API searches for label identifier 1. If the current position of the element pointer points to an occurrence of the label identifier 1, then the search starts at the next element. The search for label identifier 2 starts at the element following the label identifier 1 element. If neither label is found when the end of the structure is reached, then the graPHIGS API generates an error, does not perform the deletion, and does not change the current element pointer.

Language Bindings

C

pdel_elems_labels (label1_id, label2_id)

Input Parameters

Pint label1_id
Label 1 identifier.

Pint label2_id
Label 2 identifier.

FORTRAN

PDELLB (label1, label2)

Input Parameters

integer label1
Label 1 identifier.

integer label2
Label 2 identifier.

Errors
5 Function Requires State (PHOP,*,STOP,*).

206 Label(s) Not Between Element Pointer And End Of Structure.
Related Subroutines
• Label

DELETE STRUCTURE (PHOP,*,*,*)

Purpose
Use Delete Structure to delete the specified structure, its identifier, and its contents. When you use this subroutine to delete a closed structure, the graPHIGS API removes all references to the deleted structure (execute structure-type elements) from all existing structures. The deleted structure is also unposted from all workstations.

When you delete a structure that is an open structure, the structure remains open and is emptied, but the graPHIGS API does not remove references to the deleted structure. The result is as though you had issued the following subroutine calls:
• Close Structure
• Delete Structure (structure identifier)
• Open Structure (structure identifier)
  If the specified structure does not exist, then no action occurs.
  If it is necessary to empty the contents of a specified structure and at the same time maintain its association with all workstations, use the Empty Structure subroutine (the structure remains posted).

Language Bindings

C
pdel_struct (struct_id)

Input Parameters
Pint struct_id
Structure identifier.

FORTRAN
PDST (strid)

Input Parameters
integer strid
Structure identifier.

Errors
2 Function Requires State (PHOP,*,*,*)

Related Subroutines
• Empty Structure
• Execute Structure
• Unpost Structure

DELETE STRUCTURE NETWORK (PHOP,*,*,*)

Purpose
Use Delete Structure Network to delete the specified structure and to delete structures referenced, either directly or indirectly, by the specified structure, depending on the value of the reference handling flag:

- If the flag is set to KEEP, then this subroutine does not delete the structures in the network that are referenced by structures outside the structure network. However, this subroutine deletes all other structures in the network hierarchy as if the application called the Delete Structure subroutine for each of them.
- If the flag is set to DELETE, then this subroutine deletes all the structures in the network as if the application called the Delete Structure subroutine for each of the structures regardless of whether they were referenced by a structure outside the network hierarchy or not.

The graPHIGS API also deletes all execute structure-type elements referencing any of the deleted structures.

**Language Bindings**

**C**

```
pdel_struct_net (struct_id, ref_flag)
```

**Input Parameters**

- `Pint struct_id`
  Structure identifier.
- `Pref_flag ref_flag`
  Reference handling flag (0=PFLAG_DEL, 1=PFLAG_KEEP).

**FORTRAN**

```
PDSN (strid, refhnf).
```

**Input Parameters**

- `integer strid`
  Structure identifier.
- `integer refhnf`
  Reference handling flag (0=PDELE, 1=PKEEP).

**Errors**

2 Function Requires State (PHOP,*,*,*)

**Related Subroutines**

- Delete Structure

---

**EMPTY STRUCTURE (PHOP,*,*,*)**

**Purpose**

Use Empty Structure to empty the contents of the specified structure.

References to this now empty structure remain intact. If the specified structure is the open structure, then the graPHIGS API sets the element pointer to zero. If the specified structure does not exist, then the graPHIGS API creates a new empty structure.
Language Bindings

C

pempty_struct (struct_id)

Input Parameters

Pint struct_id
Structure identifier.

FORTRAN

PEMST (strid)

Input Parameters

integer strid
Structure identifier.

Errors

2  Function Requires State (PHOP,*,*,*)

Related Subroutines

•  None

---

LABEL (PHOP,*,STOP,*)

Purpose

Use Label to insert a Label structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Label structure element, depending on the current edit mode.

This structure element defines a label that the application uses to reference and modify structure elements.

Language Bindings

C

plabel (label_id)

Input Parameters

Pint label_id
Label identifier.

FORTRAN

PLB (label)

Input Parameters
integer label
Label identifier.

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Delete Elements Between Labels
• Set Element Pointer At Label

OFFSET ELEMENT POINTER (PHOP,*,STOP,*)

Purpose
Use Offset Element Pointer to move the element pointer to a new element relative to the current element pointer.

This subroutine adds the pointer offset value to the element pointer. It can either be positive, which moves the element pointer forward, or negative, which moves the element pointer backward. If the resultant value is less than zero, then the element pointer defaults to a value of zero. If the resultant value is greater than the number of elements in the open structure, then the graPHIGS API sets the pointer to the last element.

Language Bindings

C

coffset_elem_ptr (elem_ptr_offset)

Input Parameters

Pint elem_ptr_offset
Element pointer offset.

FORTRAN

POSEP (epo)

Input Parameters

integer epo
Element pointer offset.

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Inquire Element Pointer

OPEN STRUCTURE (PHOP,*,STCL,*)

Purpose
Use Open Structure to open a structure. The structure state is set to Structure Open (STOP).
When the application opens a structure, the graPHIGS API sets the element pointer to the last element of the currently specified structure. If the specified structure does not exist, then the graPHIGS API creates an empty structure and sets the element pointer to zero.

The graPHIGS API sets no limit on the number of simultaneously definable structures.

**Language Bindings**

**C**

`popen_struct (struct_id)`

**Input Parameters**

`Pint struct_id`

Structure identifier.

**FORTRAN**

`POPST (strid)`

**Input Parameters**

`integer strid`

Structure identifier.

**Errors**

6 Function Requires State (PHOP,*,STCL,*)

**Related Subroutines**

- Close Structure

---

**SET EDIT MODE (PHOP,*,*,*)**

**Purpose**

Use Set Edit Mode to set the current edit mode entry of the graPHIGS API state list to INSERT or REPLACE. If the application does not use this subroutine, then the edit mode defaults to INSERT.

When the application moves an element into an open structure, the edit mode determines how the element is to be placed into the structure:

- If the edit mode is set to REPLACE, then the graPHIGS API deletes the element at the element pointer. The incoming element takes its place. The graPHIGS API does not move the element pointer (so if another element is to be placed into the structure, then it replaces the last element that was just placed into the structure). If the element pointer is set to element 0, then the graPHIGS API inserts the incoming element into the structure and sets the element pointer to element 1.

- If the edit mode is set to INSERT, then the graPHIGS API inserts the incoming element into the open structure after the element at the element pointer. The graPHIGS API then increments the element pointer by 1 to point to the new element.

The edit mode does not affect elements copied into the open structure by the Copy All Elements From Structure subroutine.

**Language Bindings**
C

*pset_edit_mode* (*edit_mode*)

**Input Parameters**

*Pedit_mode edit_mode*

Edit mode (*0=PEDIT_INSERT, 1=PEDIT_REPLACE*).

**FORTRAN**

*PSEDM* (*editmo*)

**Input Parameters**

*integer editmo*

Edit mode (*0=PINSRT, 1=PREPLC*).

**Errors**

2 Function Requires State (*PHOP,*,*,*)

**Related Subroutines**

- Inquire Edit Mode

---

**SET ELEMENT POINTER (PHOP,*,STOP,*)**

**Purpose**

Use Set Element Pointer to set the element pointer to the specified element number.

If the value is less than zero, then the element pointer defaults to a value of zero. If the value is greater than the number of elements in the open structure, then the graPHIGS API sets the pointer to the last element.

**Language Bindings**

C

*pset_elem_ptr* (*elem_ptr_value*)

**Input Parameters**

*Pint elem_ptr_value*

Element pointer value.

**FORTRAN**

*PSEP* (*ep*)

**Input Parameters**

*integer ep*

Element pointer value.

**Errors**
Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Inquire Element Pointer

SET ELEMENT POINTER AT LABEL (PHOP,*,STOP,*)

Purpose
Use Set Element Pointer at Label to set the element pointer to the specified label identifier element within the open structure.

Starting at the position following the current element pointer, the graPHIGS API searches for the first occurrence of the specified label. If the element pointer is already positioned at an occurrence of the specified label, then the search starts at the next element. If the graPHIGS API does not find the label when the end of the structure is reached, then the graPHIGS API generates an error and does not change the position of the current element pointer.

Language Bindings

C
pset_elem_ptr_label (label_id)

Input Parameters
Pint label_id
Label identifier.

FORTRAN
PSEPLB (ep)

Input Parameters
integer label
Label identifier.

Errors
5 Function Requires State (PHOP,*,STOP,*)
205 Label Not Between Element Pointer And End Of Structure

Related Subroutines
- Inquire Element Pointer
- Label
Chapter 7. Workstation Table Settings

When your application open a workstation, the graPHIGS API automatically creates workstation tables to describe the workstation. Each workstation table has default settings. Use the subroutines in this section to modify some of the table entries according to your application’s specifications.

Most workstation table indexes begin with 1. The following exceptions begin with 0:
• color tables
• view tables

Some table entries may not be modified.

The subroutines in this section do not create structure elements or modify structure content. The changed table values only take effect after you update the workstation. The application may inquire the default table settings by issuing the appropriate inquiry programming calls. For a listing of the default tables for each supported workstation type, see The graPHIGS Programming Interface: Technical Reference.

SET COLOR MODEL (PHOP, WSOP, *, *)

Purpose
Use Set Color Model to set the current color model for the specified workstation to the given color model. Possible color models include: 1=RGB, 2=CIELUV, and 3=HSV. The CMY color model is supported via GP.xxxx subroutines.

Use this subroutine to specify a color model. The graPHIGS API uses this color model to interpret the color parameters for color definition and inquiries pertaining to the specified workstation.

Language Bindings

C

pset_colr_model (ws_id, colr_model)

Input Parameters

Pint ws_id
  Workstation identifier.

Pint colr_model
  Color model (1=PMODEL_RGB, 2=PMODEL_CIELUV, 3=PMODEL_HSV).

FORTRAN

PSCMD (wkid, cmodel)

Input Parameters

integer wkid
  Workstation identifier.

integer cmodel
  Color model (1=PRGB, 2=PCIE, 3=PHSV).

Errors

© Copyright IBM Corp. 1994, 2002 95
3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability
110 Specified Color Model Not Available On Workstation

Related Subroutines
• Inquire Color Model

SET COLOR REPRESENTATION (PHOP,WSOP,*,*)

Purpose

Use Set Color Representation to set the specified color values at the specified color table entry of the workstation’s color table. All currently supported color models require three floating-point components. Each component must be in the range [0.0, 1.0].

If the color model is Hue-Saturation-Value (HSV), then the graPHIGS API uses the first color component (hue) as a fraction of the total range available (that is, zero to one is used to represent zero degrees to 360 degrees). If the second color component is zero, then the graPHIGS API ignores the first color component.

Language Bindings

C

pset_colr_rep (ws_id, colr_ind, colr_rep)

Input Parameters

Pint ws_id
Workstation identifier.

Pint colr_ind
Color index (>=0).

Pcolr_rep *colr_rep
Color representation.

FORTRAN

PSCR (wkid, ci, nccs, cspec)

Input Parameters

integer wkid
Workstation identifier.

integer ci
Color index (>=0).

integer nccs
Number of components of color specification.

real cspec(*)
Color specification.
Errors
3    Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
59   Specified Workstation Does Not Have Output Capability
113  Color Index Value < ZERO
103  Exceeded Maximum Number Of Workstation Bundle Table Entries
118  Color Component Is Out Of Range

Related Subroutines
• Inquire Color Facilities
• Inquire Color Representation
• Inquire Color Model

SET EDGE REPRESENTATION (PHOP,WSOP,*,*)

Purpose
Use Set Edge Representation to set the given attribute values in the specified entry of the edge bundle table for the specified workstation.

Language Bindings

C

pset_edge_rep (ws_id, edge_ind, edge_bundle)

Input Parameters
Pint ws_id
    Workstation identifier.

Pint edge_ind
    Edge bundle index (>=1).

const Pedge_bundle *edge_bundle
    Edge representation.

FORTRAN

PSEDR (wkid, edi, edflag, edtype, ewidth, col)

Input Parameters

integer wkid
    Workstation identifier.

integer edi
    Edge bundle index (>=1).

integer edflag
    Edge flag (0=POFF, 1=PON).

integer edtype
    Edge type (1=PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).
real ewidth
   Edge width scale factor.

integer coli
   Edge color index (>=0).

Errors
3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability
100 Bundle Index Value Is Less Than One
103 Exceeded Maximum Number Of Workstation Bundle Table Entries
113 Color Index Value < ZERO
107 Specified Edgetype Not Available On Workstation

Related Subroutines
• Inquire Edge Facilities
• Inquire Edge Representation
• Set Color Model
• Set Edge Color Index
• Set Edge Index
• Set Edgetype
• Set Edgewidth Scale Factor
• Set Individual ASF

SET HIGHLIGHTING FILTER (PHOP,WSOP,*,*)

Purpose
Use Set Highlighting Filter to set the inclusion and exclusion highlighting filters for the specified workstation. The new filters take effect when you update the workstation.

The filters consist of class names which indicate which classes the graPHIGS API includes or excludes from highlighting. The same class names may exist in both the inclusion and exclusion filter. If a class name is in both filters when you update the workstation, then the graPHIGS API excludes the class. For more information on classes and class names, see The graPHIGS Programming Interface: Understanding Concepts. Use the Inquire PHIGS Facilities subroutine to inquire the number of available names for name sets.

Language Bindings

C

pset_highl_filter (ws_id, filter)

Input Parameters
Pint ws_id
   Workstation identifier.
const Pfilter *filter
    Highlighting filter.

FORTRAN

PSHLFT (wkid, isn, is, esn, es)

### Input Parameters

- **integer wkid**: Workstation identifier.
- **integer isn**: Number of names in the inclusion set (>=0).
- **integer is(isn)**: Inclusion set.
- **integer esn**: Number of names in the exclusion set (>=0).
- **integer es(esn)**: Exclusion set.

### Errors

- **3**: Function Requires State (PHOP,WSOP,*,*)
- **54**: Specified Workstation Is Not Open
- **59**: Specified Workstation Does Not Have Output Capability

### Related Subroutines

- Add Names To Set
- Inquire Highlighting Filter
- Inquire PHIGS Facilities
- Remove Names From Set

---

**SET HLHSR MODE (PHOP,WSOP,*,*)**

### Purpose

Use Set HLHSR Mode to set the requested Hidden Line/Hidden Surface Removal (HLHSR) mode in the workstation state list to the value specified.

If the dynamic modification accepted for HLHSR mode in the workstation description table is set to Immediate (IMM), or if the display surface empty entry in the workstation state list is set to *EMPTY*, then the graPHIGS API sets the current HLHSR mode in the workstation state list to the specified value and sets the HLHSR update state to *NOTPENDING*. Otherwise, the graPHIGS API sets the HLHSR update state to *PENDING* and does not change the current HLHSR mode.

### Language Bindings

- **C**

  `pset_hlhsr_mode (ws_id, hlhsr_mode)`

### Input Parameters
**Pint ws_id**
Workstation identifier.

**Pint hlhsr_mode**
HLHSR mode (0=OFF, 1=ON THE FLY).

**FORTRAN**

**PSHRM (wkid, hrm)**

**Input Parameters**

- **integer wkid**
  Workstation identifier.

- **integer hrm**
  HLHSR mode (0=OFF, 1=ON THE FLY).

**Errors**

- **3** Function Requires State (PHOP,WSOP,*,*).
- **54** Specified Workstation Is Not Open
- **59** Specified Workstation Does Not Have Output Capability
- **111** Specified HLHSR Mode Not Available On Workstation

**Related Subroutines**
- Inquire HLHSR Identifier Facilities
- Inquire HLHSR Mode Facilities
- Set HLHSR Identifier

---

**SET INTERIOR REPRESENTATION (PHOP,WSOP,*,* )**

**Purpose**

Use Set Interior Representation to set the given attribute values in the specified table entry of the specified workstation.

The *EMPTY* and *HOLLOW* interior styles display nothing for the interior. If the edge flag is set to *OFF* and the interior style is *EMPTY*, then the graPHIGS API generates no visual output. The interior is detectable when the graPHIGS API encounters a primitive with an interior style of *EMPTY* and the primitive is eligible for picking, as determined by its visibility and detectability.

If the edge flag is *OFF* and the interior style is *HOLLOW*, then the graPHIGS API draws the boundary. When the graPHIGS API encounters a primitive with an interior style of *HOLLOW* only the boundary of the primitive is eligible for picking, as determined by its visibility and detectability.

**Language Bindings**

**C**

```c
pset_int_rep (ws_id, int_ind, int_bundle)
```

**Input Parameters**
\textit{Pint ws\_id}

Workstation identifier.

\textit{Pint int\_ind}

Interior bundle index (>=1).

\textit{const Pint\_bundle \*int\_bundle}

Interior representation.

\textbf{FORTRAN}

\textit{PSIR (wkid, ii, ints, styli, coli)}

\textbf{Input Parameters}

\textit{integer wkid}

Workstation identifier.

\textit{integer ii}

Interior index (>=1).

\textit{integer ints}

Interior style (0=PHOLLO, 1=PSOLID, 2=PPATTR, 3=PHATCH, 4=PISEMP).

\textit{integer styli}

Interior style index (>=1).

\textit{integer coli}

Interior color index (>=0).

\textbf{Errors}

3 Function Requires State (PHOP,WSOP,*,*)

54 Specified Workstation Is Not Open

59 Specified Workstation Does Not Have Output Capability

100 Bundle Index Value Is Less Than One

103 Exceeded Maximum Number Of Workstation Bundle Table Entries

108 Specified Interior Style Not Available On Workstation

112 Pattern Index Value < ONE

113 Color Index Value < ZERO

\textbf{Related Subroutines}

- Inquire Interior Representation
- Set Color Model
- Set Individual ASF
- Set Interior Color Index
- Set Interior Index
- Set Interior Style Index

\textbf{SET INVISIBILITY FILTER (PHOP,WSOP,*,*)}

\textbf{Purpose}

Use Set Invisibility Filter to set the inclusion and exclusion invisibility filters for the specified workstation.
The new filters take effect when you update the workstation. The filters consist of class names which indicate which classes the graPHIGS API includes or excludes from invisibility. The same classes may exist in both the inclusion and exclusion filter. If a class is in both filters when you update the workstation, then the graPHIGS API excludes the class. For more information on classes and class names, see *The graPHIGS Programming Interface: Understanding Concepts*. Use the Inquire PHIGS Facilities subroutine to inquire the number of available names for name sets.

**Language Bindings**

**C**

```c
pset_invis_filter (ws_id, filter)
```

**Input Parameters**

```c
Pint ws_id
    Workstation identifier.

const PfILTER *filter
    Invisibility filter.
```

**FORTRAN**

```fortran
PSIVFT (wkid, isn, is, esn, es)
```

**Input Parameters**

```fortran
integer wkid
    Workstation identifier.

integer isn
    Number of names in the inclusion set (>=0).

integer is(isn)
    Inclusion set.

integer esn
    Number of names in the exclusion set (>=0).

integer es(esn)
    Exclusion set.
```

**Errors**

- **3** Function Requires State (PHOP,WSOP,*,*)
- **54** Specified Workstation Is Not Open
- **59** Specified Workstation Does Not Have Output Capability

**Related Subroutines**

- Add Names To Set
- Inquire Invisibility Filter
- Inquire PHIGS Facilities
- Remove Names From Set

## SET PATTERN REPRESENTATION (PHOP,WSOP,*,*)

**Purpose**

102 *The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference*
Use Set Pattern Representation to set a given pattern definition in the specified entry of the workstation's pattern table.

The pattern is a grid of color indexes. The color indexes point into the color table of the specified workstation.

If the workstation supports interior style PATTERN, then the pattern table in the workstation state list has predefined entries taken from the workstation description table. For every workstation of category OUTPUT or OUTIN supporting interior style PATTERN, the graPHIGS API predefines a number of pattern table entries. With this function, you can redefine the table entries.

During structure traversal, if a pattern color index specified in the pattern color index array is not available on the workstation, then the graPHIGS API uses a color index value of 1.

Some workstations require that the pattern fill is a fixed size. For these workstations, the graPHIGS API replicates the specified pattern to the fixed size. (See The graPHIGS Programming Interface: Technical Reference for interior pattern information).

**Language Bindings**

**C**

```c
pset_pat_rep (ws_id, pat_ind, pat_bundle)
```

**Input Parameters**

*Pint ws_id*
  Workstation identifier.

*Pint pat_ind*
  Pattern index (>=1).

*const Ppat_rep *pat_bundle*
  Pattern representation (The pattern color index array must be in row order).

**FORTRAN**

```fortran
PSPAR (wkid, pai, dimx, dimy, isc, isr, dx, dy, colia)
```

**Input Parameters**

*integer wkid*
  Workstation identifier.

*integer pai*
  Pattern index (>=1).

*integer dimx*
  X dimension of colia which contains the pattern color index array (>=1).

*integer dimy*
  Y dimension of colia which contains the pattern color index array (>=1).

*integer isc*
  Index to start column (>=1).

*integer isr*
  Index to start row (>=1).
integer dx
Number of columns used (>=1).

integer dy
Number of rows used (>=1).

integer colia(dimx, dimy)
Pattern color index array (A grid of dimx by dimy color indexes. The array must be in row order. The pattern within this array begins at position (isc, isr), and is of dimension dx by dy).

Errors
3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability
112 Pattern Index Value < ONE
103 Exceeded Maximum Number Of Workstation Bundle Table Entries
116 One Dimension Of Pattern Color Index Array < ONE
113 Color Index Value < ZERO

Related Subroutines
• Inquire Pattern Representation

SET POLYLINE REPRESENTATION (PHOP,WSOP,*,*)

Purpose
Use Set Polyline Representation to set the given attribute values in the specified entry of the polyline bundle table.

Language Bindings

C
pset_line_rep (ws_id, line_ind, line_bundle)

Input Parameters
Pint ws_id
Workstation identifier.

Pint line_ind
Polyline bundle index (>=1).

const Pline_bundle *line_bundle
Polyline representation.

FORTRAN
PSPLR (wkid, pli, ltype, lwidth, coli)

Input Parameters
integer wkid
Workstation identifier.
integer pli
    Polyline bundle index (>=1).

integer ltype
    Polyline line types (1=PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).

real lwidth
    Line width scale factor.

integer coli
    Polyline color index (>=0).

Errors
3      Function Requires State (PHOP,WSOP,*,*)
54     Specified Workstation Is Not Open
59     Specified Workstation Does Not Have Output Capability
100    Bundle Index Value Is Less Than One
103    Exceeded Maximum Number Of Workstation Bundle Table Entries
104    Specified Linetype Not Available On Workstation
113    Color Index Value < ZERO

Related Subroutines
- Inquire Polyline Representation
- Set Color Model
- Set Individual ASF
- Set Linetype
- Set Linewidth Scale Factor
- Set Polyline Color Index
- Set Polyline Index

SET POLYMARKER REPRESENTATION (PHOP,WSOP,*,*)

Purpose
Use Set Polymarker Representation to set the given attribute values in the specified entry of the polymarker bundle table.

Language Bindings
C

pset_marker_rep (ws_id, marker_ind, marker_bundle)

Input Parameters
Pint ws_id
    Workstation identifier.

Pint marker_ind
    Polymarker bundle index (>=1).

const Pmarker_bundle *marker_bundle
    Polymarker representation.
FORTRAN

PSPMR (wkid, pmi, mtype, mszsf, coli)

Input Parameters

integer wkid
   Workstation identifier.

integer pmi
   Polymarker bundle index (>=1).

integer mtype
   Marker type (1=POINT, 2=PPLUS, 3=PAST, 4=POMARK, 5=PXMARK).

real mszsf
   Marker size scale factor.

integer coli
   Polymarker color index (>=0).

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability
100 Bundle Index Value Is Less Than One
103 Exceeded Maximum Number Of Workstation Bundle Table Entries
105 Specified Marker Type Not Available On Workstation
113 Color Index Value < ZERO

Related Subroutines

• Inquire Polymarker Representation
• Set Color Model
• Set Individual ASF
• Set Marker Size Scale Factor
• Set Polymarker Color Index
• Set Polymarker Index

SET TEXT REPRESENTATION (PHOP,WSOP,*,*)

Purpose

Use Set Text Representation to set the given attribute values into the specified entry of the text bundle table.

If you specify a precision that the workstation does not support, then the graPHIGS API substitutes the font's highest available precision for that workstation.

Language Bindings

C

pset_text_rep (ws_id, text_ind, text_bundle)
Input Parameters

Pint ws_id
   Workstation identifier.

Pint text_ind
   Text bundle index (>=1).

const Ptext_bundle *text_bundle
   Text representation.

FORTRAN

PSTXR (wkid, txi, font, prec, chxp, chsp, coli)

Input Parameters

integer wkid
   Workstation identifier.

integer text
   Text bundle index (>=1).

integer font
   Text font.

integer prec
   Text precision (0=PSTRP, 1=PCHARP, 2=PSTRKP).

real chxp
   Character expansion factor.

real chsp
   Character spacing.

integer coli
   Text color index (>=0).

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability
100 Bundle Index Value Is Less Than One
103 Exceeded Maximum Number Of Workstation Bundle Table Entries
106 Specified Font Not Available For Requested Text Precision
113 Color Index Value < ZERO

Related Subroutines

• Inquire Text Representation
• Set Character Expansion Factor
• Set Character Spacing
• Set Color Model
• Set Individual ASF
• Set Text Color Index
• Set Text Font
• Set Text Index
• Set Text Precision

SET VIEW REPRESENTATION (PHOP, WSOP, *, *)

Purpose

Use Set View Representation to set fields in the specified entry of the workstation’s view table. The view orientation matrix and the view mapping matrix are first expanded to 4x4 matrixes. Then the graPHIGS API stores the specified values in the REQUESTED view table entry. The graPHIGS API sets the corresponding CURRENT values in the view table entry to the REQUESTED values when you update the workstation.

The clipping indicators determine to which boundaries the graPHIGS API clips the contents of the view. The graPHIGS API sets the Z portion of the requested view clipping limits for the specified view to the default values.

The workstation’s view table is 0 based, however, you cannot change view entry 0. (See The graPHIGS Programming Interface: Technical Reference for the default values for view entry 0).

Language Bindings

C

pset_view_rep (ws_id, view_ind, view_rep)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint view_ind
   View index (>=1).

const Pview_rep *view_rep
   View representation.

FORTRAN

PSVWR (wkid, viewi, vwormt, vwmpmt, vwcpml, xyclpi)

Input Parameters

integer wkid
   Workstation identifier.

integer viewi
   View index (>=1).

real vwormt(3,3)
   View orientation matrix.

real vwmpmt(3,3)
   View mapping matrix.
real vwcplm(4)
    View clipping limits in NPC (XMIN, XMAX, YMIN, YMAX).

integer xyclpi
    X-Y clipping indicator (0=PNCLIP, 1=PCLIP).

Errors
3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
57    Specified Workstation Is Of Category MI
115   View Index Value < ONE
150   Exceeded Maximum Number Of View Table Entries
153   Invalid View Clipping Limits: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
154   View Clipping Limits Are Not Within NPC Range

Related Subroutines
• Inquire View Representation
• Set View Representation 3

SET VIEW REPRESENTATION 3 (PHOP,WSOP,*,*)

Purpose

Use Set View Representation 3 to set fields in the specified entry of the workstation’s view table. The graPHIGS API stores the specified values in the REQUESTED view table entry. The graPHIGS API sets the corresponding CURRENT values in the view table entry to the REQUESTED values when you update the workstation.

The clipping indicators determine to which boundaries the graPHIGS API clips the contents of the view.

The workstation’s view table is 0 based, however, you cannot change view entry 0. (See The graPHIGS Programming Interface: Technical Reference for the default values for view entry 0).

Language Bindings

C

pset_view_rep3 (ws_id, view_ind, view_rep)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint view_ind
    View index (>=1).

const Pview_rep3 *view_rep
    View representation.

FORTRAN

PSVWR3 (wkid, viewi, vwormt, vwmpmt, vwcplm, xyclpi, bclipi, fclipi)
**Input Parameters**

integer wkid
   Workstation identifier.

integer viewi
   View index (>=1).

real vwormt(4,4)
   View orientation matrix.

real vwmpmt(4,4)
   View mapping matrix.

real vwclplm(6)
   View clipping limits in NPC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer xyclpi
   X-Y clipping indicator (0=PNCLIP, 1=PCLIP).

integer bclipi
   Back clipping indicator (0=PNCLIP, 1=PCLIP).

integer fclipi
   Front clipping indicator (0=PNCLIP, 1=PCLIP).

**Errors**

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
57  Specified Workstation Is Of Category MI
115  View Index Value < ONE
150  Exceeded Maximum Number Of View Table Entries
153  Invalid View Clipping Limits: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
154  View Clipping Limits Are Not Within NPC Range

**Related Subroutines**

- Inquire View Representation
- Set View Representation

SET VIEW TRANSFORMATION INPUT PRIORITY (PHOP,WSOP,*,*)

**Purpose**

Use Set View Transformation Input Priority to modify the input priority of the specified view in relation to another view on the specified workstation. If the specified view index is the same as the reference view index, then this function has no effect.

This subroutine only changes the input priority of a view. The display surface shows no visual changes.

**Language Bindings**

C

pset_view_tran_in_pri (ws_id, view_ind, ref_view_ind, rel_pri)
Input Parameters

\textit{Pint ws\_id}

Workstation identifier.

\textit{Pint view\_ind}

View index (\(\geq 0\)).

\textit{Pint ref\_view\_ind}

Reference view index (\(\geq 0\)).

\textit{Prel\_pri rel\_pri}

Relative priority (\(0=\text{PPRI\_HIGHER}, \ 1=\text{PPRI\_LOWER}\)).

FORTRAN

\texttt{PSVTIP (wkid, viewi, rfvwix, relpri)}

Input Parameters

\texttt{integer wkid}

Workstation identifier.

\texttt{integer viewi}

View index (\(\geq 0\)).

\texttt{integer rfvwix}

Reference view index (\(\geq 0\)).

\texttt{integer relpri}

Relative priority (\(0=\text{PHIGHR}, \ 1=\text{PLOWER}\)).

Errors

3 Function Requires State (PHOP,WSOP,*,*)

54 Specified Workstation Is Not Open

57 Specified Workstation Is Of Category MI

114 View Index Value < ZERO

101 Specified Representation Has Not Been Defined

Related Subroutines

\begin{itemize}
  \item Inquire List Of View Indices
\end{itemize}
Chapter 8. Structure Display Subroutines

The subroutines included in this section let your application program post or unpost a structure network for display on a workstation.

**POST STRUCTURE (PHOP,WSOP,*,*)**

**Purpose**

Use Post Structure to add the specified structure to the list of posted structures in the workstation state list (WSL) of the specified workstation. This subroutine adds the workstation identifier to the list of workstations to which the specified structure is posted. If the specified structure does not exist, then the graPHIGS API creates a new empty structure.

The graPHIGS API assigns the specified display priority to the structure network. The display priority indicates the relative importance of the posted structure network. If your application posts multiple structures for display to the same display space location, then the graPHIGS API displays the higher priority structure network. If two structures have the same priority, then the graPHIGS API considers the last posted structure to have the higher priority.

Upon second and subsequent posting of a structure, the graPHIGS API removes the structure from the list of posted structures, and then reposts it at the priority the application specifies.

**Language Bindings**

**C**

```c
ppost_struct (ws_id, struct_id, pri)
```

**Input Parameters**

- `Pint ws_id`: Workstation identifier.
- `Pint struct_id`: Structure identifier.
- `Pfloat pri`: Display priority (0.0<=priority<=1.0).

**FORTRAN**

```fortran
PPOST (wkid, strid, prior)
```

**Input Parameters**

- `integer wkid`: Workstation identifier.
- `integer strid`: Structure identifier.
- `real prior`: Display priority (0.0<=priority<=1.0).

**Errors**
UNPOST ALL STRUCTURES (PHOP,WSOP,*,*)

Purpose

Use Unpost All Structures to unpost all structures from the specified workstation. Unposting a structure does not delete the structure.

Language Bindings

C

punpost_all_structs (ws_id)

Input Parameters

Pint ws_id
Workstation identifier.

FORTRAN

PUPAST (wkid)

Input Parameters

integer wkid
Workstation identifier.

Errors

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability

Related Subroutines

• Inquire Posted Structures
• Inquire Set Of Workstations To Which Posted
• Post Structure
• Unpost Structure
UNPOST STRUCTURE (PHOP,WSOP,*,*)

Purpose

Use Unpost Structure to unpost the specified structure from the specified workstation. Unposting a structure does not delete the structure.

Language Bindings

C

punpost_struct (ws_id, struct_id)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint struct_id
   Structure identifier.

FORTRAN

PUPOST (wkid, strid)

Input Parameters

integer wkid
   Workstation identifier.

integer strid
   Structure identifier.

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability

Related Subroutines

- Inquire Posted Structures
- Inquire Set Of Workstations To Which Posted
- Post Structure
- Unpost Structure
Chapter 9. Structure Archiving Subroutines

The subroutines included in this section let your application program manipulate structure archive files.

Invoke these subroutines to:

- open/close archive files
- create structures and structure hierarchies in an archive file
- delete structures and structure hierarchies in an archive file
- set conflict resolution flags
- receive structure identifier and structure path information from an archive file

**ARCHIVE ALL STRUCTURES (PHOP,*,*,AROP)**

**Purpose**

Use Archive All Structures to store all structures from the structure store into the specified open archive file.

If any of the specified structures in the structure store already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

**Language Bindings**

**C**

par_all_structs(archive_id)

**Input Parameters**

\[ Pint \text{ archive_id} \]

Archive file identifier.

**FORTRAN**

PARAST(afid)

**Input Parameters**

\[ integer \text{ afid} \]

Archive file identifier.

**Errors**

- 7   Function Requires State (PHOP,*,*,AROP)
- 404 Specified Archive File Is Not Open
- 405 Name Conflict Occurred, Conflict Resolution Flag = Abandon
- 406 Warning, Archive File Is Full

**Related Subroutines**

- Archive Structure Networks
- Archive Structures
• Set Conflict Resolution

ARCHIVE STRUCTURE NETWORKS (PHOP,*,*,AROP)

Purpose

Use Archive Structure Networks to store one or more structure networks from the structure store into the specified open archive file.

If any of the specified root structures do not exist in the structure store, then the graPHIGS API issues a warning and no action is taken for the non-existing structures. If any of the specified structures in the structure network already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

If the archive file is not large enough to complete the archival process, then the graPHIGS API issues an error and does not archive any other structure networks. However, any structures that the graPHIGS API archived are archived completely.

Language Bindings

C

\texttt{par\_struct\_nets(archive\_id, struct\_ids)}

Input Parameters

\textit{Pint archive\_id}

Archive file identifier.

\textit{cont Pint\_list *struct\_ids}

List of root structure identifiers.

FORTRAN

\texttt{PARSN(afid, n, lstrid)}

Input Parameters

\textit{integer afid}

Archive file identifier.

\textit{integer n}

Number of root structure identifiers in the list.

\textit{integer lstrid(n)}

List of root structure identifiers.

Errors

7 Function Requires State (PHOP,*,*,AROP)

404 Specified Archive File Is Not Open

200 Warning, Ignoring Structures That Do Not Exist

405 Name Conflict Occurred, Conflict Resolution Flag = Abandon

406 Warning, Archive File Is Full

Related Subroutines

118 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
• Archive Structure Networks
• Archive Structures
• Set Conflict Resolution

ARCHIVE STRUCTURES (PHOP,*,*,AROP)

Purpose

Use Archive Structures to store one or more specified structures from the structure store into the specified open archive file.

If any of the specified structures do not exist in the structure store, then the graPHIGS API issues a warning and no action is taken for the non-existing structures. If any of the specified structures already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

If the archive file is not large enough to complete the archival process, then the graPHIGS API issues an error and does not archive any other structures. However, any structures that the graPHIGS API archived are archived completely.

Language Bindings

C

par_structs(archive_id, struct_ids)

Input Parameters

Pint archive_id
   Archive file identifier.

cont Pint_list *struct_ids
   List of structure identifiers.

FORTRAN

PARST(afid, n, lstrid)

Input Parameters

integer afid
   Archive file identifier.

integer n
   Number of structure identifiers in the list.

integer lstrid(n)
   List of structure identifiers.

Errors

7    Function Requires State (PHOP,*,*,AROP)
404  Specified Archive File Is Not Open
200  Warning, Ignoring Structures That Do Not Exist
405  Name Conflict Occurred, Conflict Resolution Flag = Abandon

Chapter 9. Structure Archiving Subroutines  119
Warning, Archive File Is Full

Related Subroutines

- Archive Structure Networks
- Archive Structures
- Set Conflict Resolution

CLOSE ARCHIVE FILE (PHOP,*,*,AROP)

Purpose

Use Close Archive File to close the specified open archive file. The graPHIGS API removes the archive file identifier from the set of open archive files.

If no other archive files are open, then this subroutine sets the current archive state to Archive Closed (ARCL).

Language Bindings

C

close_ar_file (archive_id)

Input Parameters

Pint archive_id

archive_id  Archive file identifier.

FORTRAN

PCLARF (afid)

Input Parameters

integer afid

afid  Archive file identifier.

Errors

7  Function Requires State (PHOP,*,*,AROP)

404  Specified Archive File Is Not Open

Related Subroutines

- Open Archive File

DELETE ALL STRUCTURES FROM ARCHIVE (PHOP,*,*,AROP)

Purpose

Use Delete All Structures from Archive to delete all structures from the specified open archive file.

This subroutine is equivalent to invoking the Delete Structures from Archive for each structure in the selected structure store.
When this subroutine call is completed, the state of the archive file is as if your application just opened it for the first time.

**Language Bindings**

**C**

```c
pdel_all_structs_ar (archive_id)
```

**Input Parameters**

*Pint archive_id*
- Archive file identifier.

**FORTRAN**

```fortran
PDASAR(afid)
```

**Input Parameters**

*integer afid*
- Archive file identifier.

**Errors**

- **7** Function Requires State (PHOP,*,*,AROP)
- **404** Specified Archive File Is Not Open

**Related Subroutines**

- Delete Structure Networks From Archive
- Delete Structures From Archive

---

**DELETE STRUCTURE NETWORKS FROM ARCHIVE (PHOP,*,*,AROP)**

**Purpose**

Use Delete Structure Networks from Archive to delete one or more structure networks from the specified open archive file.

The graPHIGS API does not check if other structures in the archive file reference the deleted structures. Therefore, the graPHIGS API does not delete the execute structure elements in other archived structures that reference the deleted structures.

If the specified structure does not exist in the archive file, then the graPHIGS API issues a warning and no action is taken for the structure.

**Language Bindings**

**C**

```c
pdel_struct_nets_ar (archive_id, struct_ids)
```

**Input Parameters**
Pint archive_id
    Archive file identifier.

cnt Pint_list *struct_ids
    List of root structure identifiers.

FORTRAN

PDSNAR (afid, n, lstrid)

Input Parameters

integer afid
    Archive file identifier.

integer n
    Number of root structure identifiers in the list.

integer lstrid(n)
    List of root structure identifiers.

Errors

7     Function Requires State (PHOP,*,*,AROP)
404   Specified Archive File Is Not Open
407   Warning, Some Specified Structures Do Not Exist On Archive File

Related Subroutines

• Delete All Structures From Archive
• Delete Structures From Archive
• Execute Structure

DELETE STRUCTURES FROM ARCHIVE (PHOP,*,*,AROP)

Purpose

Use Delete Structures from Archive to delete one or more structures from the specified open archive file.

The graPHIGS API does not check if other structures in the archive file reference the deleted structures. Therefore, the graPHIGS API does not delete the execute structure elements in other archived structures that reference the deleted structures.

If a specified structure does not exist in the archive file, then the graPHIGS API issues a warning and no action is taken for the structure.

Language Bindings

C

pdel_structs_ar (archive_id, struct_ids)

Input Parameters

Pint archive_id
    Archive file identifier.
cont Pint_list *struct_ids
        List of structure identifiers.

FORTRAN

PDSTAR (afid, n, lstrid)

Input Parameters

integer afid
        Archive file identifier.

integer n
        Number of structure identifiers in the list.

integer lstrid(n)
        List of structure identifiers.

Errors

7    Function Requires State (PHOP,*,*,AROP)
404  Specified Archive File Is Not Open
407  Warning, Some Specified Structures Do Not Exist On Archive File

Related Subroutines

• Delete All Structures From Archive
• Delete Structures From Archive
• Execute Structure

OPEN ARCHIVE FILE (PHOP,*,*,*)

Purpose

Use Open Archive File to open a graPHIGS API archive file.

This subroutine function sets the current archive state to Archive Open (AROP). The graPHIGS API adds the archive file identifier to the set of open archive files.

The graPHIGS API External Defaults File (EDF) allows the application to denote, indirectly, the actual value of the file descriptor. For more information on the External Defaults File, see The graPHIGS Programming Interface: Technical Reference under “Defaults and Nicknames”.

Language Bindings

C

popen_ar_file (archive_id, archive_file)

Input Parameters

Pint archive_id
        Archive file identifier.

const char *archive_file
        Archive file descriptor. This parameter looks like a Unix file descriptor which consists of a
[path]/filename[extension]. Path is the route of directories through the file system. Path is optional and ignored for MVS and VM. An example of a full file descriptor:

/phigs/file1.archive

where:

- path = /phigs which says go from the root directory to the directory phigs.
- filename = file1
- extension = .archive

The following rules apply to the descriptor, depending on which system the nucleus is running in:

- **AIX**
  If you did not specify the path, then the graPHIGS API uses the default directory at the time of the execution of the subroutine.

- **MVS**
  - filename - You must specify a filename. This is the DD-name of the BSAM data set of the archive file.
  - extension - Any extension is ignored.

- **VM/CMS**
  The file descriptor can have one of two forms:
  - filename [filetype [filemode]] or
  - filename[filetype[,filemode]]

  If the filetype is missing, then the graPHIGS API uses a filetype of ARCHIVE.
  If the filemode is missing, then the graPHIGS API uses a filemode of A1.

**FORTRAN**

POPARF (afid, arcfil)

Input Parameters

integer afid
  Archive file identifier.

integer arcfil
  Archive file descriptor. An integer value for this descriptor. An association between this value and a valid file descriptor is then made via an External Defaults File (EDF).

For example, the file descriptor of file1.archive is attained if the EDF contains the following line:

AFMMDFT ARCHIVE=(99,file1.archive,)

and the application is coded:

CALL POPARF(1,99)

If the EDF does not contain a match for the integer value, then a filename is created based on the integer value using the following formula:

final value = Remainder (Absolute Value (original value) / 999)

The graPHIGS API uses this final value (000<=final value<=999) to create the filename (AFMAFxxx, where xxx is the final value).

**Errors**

2  Function Requires State (PHOP,**,**)

124  The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
402 Archive File Identifier Already In Use
400 Archive File Cannot Be Opened
401 Exceeded Maximum Number Of Simultaneously Open Archive Files
403 Archive File Is Not A PHIGS Archive File

Related Subroutines
• Close Archive File

RETREIVE ALL STRUCTURES (PHOP,*,*,AROP)

Purpose

Use Retrieve All Structures to retrieve all structures from the specified open archive file and place them into the structure store.

If any of the specified structures in the archive file already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

When you retrieve a structure that is the open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:
1. Close Structure
2. Empty Structure
3. Retrieve Structure
4. Open Structure

If the structure retrieved contains an execute structure element that references a non-existing structure, then the graPHIGS API creates an empty structure.

Language Bindings

C

cret_all_structs (archive_id)

Input Parameters

Pint archive_id
    Archive file identifier.

FORTRAN

PRAST (afid)

Input Parameters

integer afid
    Archive file identifier.

Errors

7 Function Requires State (PHOP,*,*,AROP)
404 Specified Archive File Is Not Open

405 Name Conflict Occurred, Conflict Resolution Flag = Abandon

Related Subroutines
- Retrieve Structure Networks
- Retrieve Structures
- Set Conflict Resolution

**RETRIEVE PATHS TO ANCESTORS (PHOP,*,*,AROP)**

**Purpose**

Use Retrieve Paths to Ancestors to retrieve the ancestral paths of the specified structure from the specified open archive file.

A path of ancestors of a structure \( S \) is a list of ordered pairs \( ((A_1,E_1), (A_2,E_2), ..., (A_m,E_m),(S,0)) \) where each ordered pair consists of an identifier of a structure \( A_x \) that is an ancestor of the specified structure \( S \) and the position of an execute structure-type element \( E_x \) that references the next structure in the path. Ancestor structure \( A_1 \) is the top of the path (e.g., not referenced by any other structure) and \( S \) is the bottom of the path.

The path order and path depth determine the portion of each path that the graPHIGS API returns. Your application can specify the path order as **TOP_FIRST** or **BOTTOM_FIRST**. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation can result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all the returned path portions are distinct.

**Language Bindings**

**C**

```c
pret_paths_ances (ar_id, struct_id, order, depth, store, paths)
```

**Input Parameters**

* `Pint ar_id`
  Archive file identifier.

* `Pint struct_id`
  Structure identifier.

* `Path_order order`
  Path order \( (0=PORDER\_TOP\_FIRST, 1=PORDER\_BOTTOM\_FIRST) \).

* `Pint depth`
  Path depth \( (>=0) \).

* `Pstore store`
  Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See the Create Store subroutine for details on how the graPHIGS API uses this parameter.

**Output Parameters**
Structure path list. The memory referenced by *paths is managed by Store. If an error occurs, then *paths is set to NULL.

FORTRAN

PREPAN (afid, strid, pthord, pthdep, ipthsiz, n, ol, apthsiz, paths)

Input Parameters

integer afid
    Archive file identifier.

integer strid
    Structure identifier.

integer pthord
    Path order (0=PPOTOP, 1=PPOBOT).

integer pthdep
    Path depth (>=0).

integer ipthsiz
    Maximum number of path entries that the buffer can hold.

integer n
    Element of the list of paths.

Output Parameters

integer ol
    Number of paths available.

integer apthsiz
    Actual number of entries in the nth structure path.

integer paths(2,ipthsiz)
    nth structure path.

Errors

7    Function Requires State (PHOP,*,*,AROP)
201  Specified Structure Does Not Exist
207  Specified Path Depth < Zero

Related Subroutines

• Retrieve Paths To Descendants

RETRIEVE PATHS TO DESCENDANTS (PHOP,*,*,AROP)

Purpose

Use Retrieve Paths to Descendants to retrieve the descendant paths of the specified structure from the specified open archive file.

A path of descendants of a structure S is a list of ordered pairs ((S,E0), (D1,E1), (D2,E2),..., (Dn,0)) where each ordered pair consists of an identifier of a structure (Dx) that is a descendant of the specified structure
(S) and the position of an execute structure-type element (Ex) that references the next structure in the path. The specified structure S is the top of the path and descendant structure Dn is the bottom of the path (e.g., it does not reference any other structure).

The path order and path depth determine the portion of each path that the graPHIGS API returns. Your application can specify the path order as TOP_FIRST or BOTTOM_FIRST. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation can result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all the returned path portions are distinct.

Language Bindings

C

pret_paths_descs (ar_id, struct_id, order, depth, store, paths)

Input Parameters

Pint ar_id
   Archive file identifier.

Pint struct_id
   Structure identifier.

Ppath_order order
   Path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

Pint depth
   Path depth (>=0).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See the Create Store subroutine for details on how the graPHIGS API uses this parameter.

Output Parameters

Pelem_ref_list_list **paths
   Structure path list. The memory referenced by *paths is managed by Store. If an error occurs, then *paths is set to NULL.

FORTRAN

PREPDE (afid, strid, pthord, pthdep, ipthsiz, n, ol, apthsz, paths)

Input Parameters

integer afid
   Archive file identifier.

integer strid
   Structure identifier.

integer pthord
   Path order (0=PPOPTOP, 1=PPOPBOT).

integer pthdep
   Path depth (>=0).
integer ipthsz
    Maximum number of path entries that the buffer can hold.

integer n
    Element of the list of paths.

Output Parameters

integer ol
    Number of paths available

integer apthsz
    Actual number of entries in the *n*th structure path.

integer paths(2,ipthsz)
    *n*th structure path.

Errors

7  Function Requires State (PHOP,*,*,AROP)
201  Specified Structure Does Not Exist
207  Specified Path Depth < Zero

Related Subroutines

* Retrieve Paths To Ancestors

---

RETRIEVE STRUCTURE IDENTIFIERS (PHOP,*,*,AROP)

Purpose

Use Retrieve Structure Identifiers to retrieve a list of structure identifiers in the specified open archive file.

Language Bindings

C

pret_struct_id (archive_id, num elems appl_list, start ind, ids, num elems impl_list)

Input Parameters

Pint archive_id
    Archive file identifier.

Pint num elems appl_list
    Number of elements in the application list (>=0).

Pint start_ind
    Starting index (>=0).

Output Parameters

Pint_list *ids
    List of structure identifiers.

Pint *num elems impl_list
    Number of elements in the implementation list.
FORTRAN

PRSID (afid, ilsize, n, lstrid)

Input Parameters

integer afid
Archive file identifier.

integer ilsize
Size of the list (lstrid).

Output Parameters

integer n
Number of structure identifiers in list.

integer lstrid(*)
List of structure identifiers.

Errors

7 Function Requires State (PHOP,*,*,AROP)
404 Specified Archive File Is Not Open

Related Subroutines

• None

RETRIEVE STRUCTURE NETWORKS (PHOP,*,*,AROP)

Purpose

Use Retrieve Structure Networks to retrieve one or more structure networks from the specified open archive file and place them into the structure store.

If any of the specified root structures do not exist in the specified archive file and the specified structure identifier does not exist in the structure store, then the graPHIGS API issues a warning and creates an empty structure. If any of the specified root structures do not exist in the specified archive file and the specified structure identifier does exist in the structure store and the conflict resolution flag is set to UPDATE, then the graPHIGS API issues a warning and empties the structure.

If any of the specified structures in the structure network already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

When you retrieve a structure that is an open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:
1. Close Structure
2. Empty Structure
3. Retrieve Structure
4. Open Structure

If the structure retrieved contains an execute structure element that references a non-existing structure, then the graPHIGS API creates an empty structure.
Language Bindings

C

pret_struct_nets (archive_id, struct_ids)

Input Parameters

Pint archive_id
Archive file identifier.

const Pint_list *struct_ids
List of root structure identifiers.

FORTRAN

PRESN (afid, n, lstrid)

Input Parameters

integer afid
Archive file identifier.

integer n
Number of root structure identifiers in list.

integer lstrid(n)
List of root structure identifiers.

Errors

7 Function Requires State (PHOP,*,*,AROP)

404 Specified Archive File Is Not Open

405 Name Conflict Occurred, Conflict Resolution Flag = Abandon

408 Warning, Structure(s) Not In Archive, Empty One(s) To Be Created

Related Subroutines

• Retrieve All Structures
• Retrieve Structures
• Set Conflict Resolution

RETRIEVE STRUCTURES (PHOP,*,*,AROP)

Purpose

Use Retrieve Structures to retrieve one or more structures from the specified open archive file and place them into the structure store.

If any of the specified structures do not exist in the specified archive file and the specified structure identifier does not exist in the structure store, then the graPHIGS API issues a warning and creates an empty structure. If any of the specified structures do not exist in the specified archive file and the specified structure identifier does exist in the structure store and the conflict resolution flag is set to UPDATE, then the graPHIGS API issues a warning and empties the structure.
If any of the specified structures in the structure network already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution subroutine.

When you retrieve a structure that is an open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:
1. Close Structure
2. Empty Structure
3. Retrieve Structure
4. Open Structure

If the structure retrieved contains an execute structure element that references a non-existing structure, then the graPHIGS API creates an empty structure.

Language Bindings

C

pret_structs (archive_id, struct_ids)

Input Parameters

Pint archive_id
   Archive file identifier.

const Pint_list *struct_ids
   List of structure identifiers.

FORTRAN

PREST (afid, n, lstrid)

Input Parameters

integer afid
   Archive file identifier.

integer n
   Number of structure identifiers in list.

integer lstrid(n)
   List of structure identifiers.

Errors

7   Function Requires State (PHOP,*;*,AROP)
404  Specified Archive File Is Not Open
405  Name Conflict Occurred, Conflict Resolution Flag = Abandon
408  Warning, Structure(s) Not In Archive, Empty One(s) To Be Created

Related Subroutines

• Retrieve All Structures
• Retrieve Structures
• Set Conflict Resolution
SET CONFLICT RESOLUTION (PHOP,*,*,*)

Purpose

Use Set Conflict Resolution to set the conflict resolution flags for use by all archive files. The graPHIGS API uses these flags to determine how to resolve conflicts when you are archiving or retrieving one or more structures (e.g., when a structure that exists in a specified archive file has the same identifier as a structure that exists in the structure store).

There are two conflict resolution flags: the archival conflict resolution for moving structure data to an archive file from the structure store and the retrieval conflict resolution for moving structure data from an archive file to the structure store. Your application can set either of these flags to: MAINTAIN, ABANDON, or UPDATE.

- If the value is set to MAINTAIN, then the graPHIGS API does not transfer any structures when a conflict occurs.
- If the value is set to ABANDON, then the graPHIGS API does not transfer any structures when a conflict occurs.
- If the value is set to UPDATE, then the graPHIGS API transfers all the structures and replaces any conflicting structures with the new ones.

If this subroutine is not used, then the archival conflict resolution flag defaults to a value of UPDATE and the retrieval conflict resolution flag defaults to a value of ABANDON.

Language Bindings

C

\texttt{pset\_conf\_res (archive\_res, retrieval\_res)}

Input Parameters

\begin{itemize}
\item \texttt{Pconf\_res archive\_res} \\
Archival conflict resolution. (0=\texttt{PRES\_MAINTAIN}, 1=\texttt{PRES\_ABANDON}, 2=\texttt{PRES\_UPD}).
\item \texttt{Pconf\_res retrieval\_res} \\
Retrieval conflict resolution (0=\texttt{PRES\_MAINTAIN}, 1=\texttt{PRES\_ABANDON}, 2=\texttt{PRES\_UPD}).
\end{itemize}

FORTRAN

\texttt{PSCNRS (arccr, retcr)}

Input Parameters

\begin{itemize}
\item \texttt{integer arccr} \\
Archival conflict resolution (0=\texttt{PCRMNT}, 1=\texttt{PCRABA}, 2=\texttt{PCRUPD}).
\item \texttt{integer retcr} \\
Retrieval conflict resolution (0=\texttt{PCRMNT}, 1=\texttt{PCRABA}, 2=\texttt{PCRUPD}).
\end{itemize}

Errors

2 Function Requires State (PHOP,*,*,*)

Related Subroutines

- Inquire Conflict Resolution
Chapter 10. Transformation Subroutines

The transformation and clipping subroutines found in this section fall into three general categories: modeling clipping, modeling transformations and workstation transformations.

Modeling Clipping

The modeling clipping subroutines create modeling clipping structure elements, which modify the current modeling clipping values that the graPHIGS API applies to primitives during traversal.

Modeling Transformations

The modeling transformation subroutines create transformation structure elements, which modify the current transformation values that the graPHIGS API applies to primitives during traversal.

When the graPHIGS API inserts a structure element into an open structure following the element pointer, the pointer moves to the new element.

Workstation Transformations

The workstation transformation subroutines allow the application to modify the mapping of Normalized Projection Coordinates (NPC) into Device Coordinates (DC) for a specified workstation.

**RESTORE MODELING CLIPPING VOLUME (PHOP,*,STOP,*)**

Purpose

Use Restore Modeling Clipping Volume to insert a Restore Modeling Clipping Volume structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Restore Modeling Clipping Volume structure element, depending on the current edit mode.

During traversal, the Restore Modeling Clipping Volume structure element restores the current modeling clipping volume in the graPHIGS API traversal state list to the volume inherited by the structure. At structure traversal time, the graPHIGS API uses this volume to render all subsequent primitives.

Language Bindings

**C**

prestore_model_clip_vol();

**FORTRAN**

PRMCV

Errors

5 Function Requires State (PHOP,*,STOP,*).

Related Subroutines

- Inquire Modeling Clipping Facilities
- Set Modeling Clipping Indicator
SET GLOBAL TRANSFORMATION (PHOP,*,STOP,*)

Purpose

Use Set Global Transformation to insert a two-dimensional Set Global Transformation structure element into the open structure following the element pointer, or replace the element pointed at by the element pointer with a Set Global Transformation structure element, depending on the current edit mode.

When the graPHIGS API encounters this element during traversal, the graPHIGS API expands it into a 4x4 matrix as follows:

\[
\begin{pmatrix}
a & b & c \\
d & e & f \\
g & h & i \\
0 & 0 & 1
\end{pmatrix}
\]

\[
\begin{pmatrix}
a & b & 0 & c \\
d & e & 0 & f \\
0 & 0 & 1 & 0 \\
g & h & 0 & i
\end{pmatrix}
\]

and causes the expanded matrix to become the current global transformation for the current structure. The resultant matrix, in conjunction with the local modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

Language Bindings

C

*pset_global_tran (global_tran)

Input Parameters

Pmatrix global_tran


FORTRAN

PSGMT (xfrmt)

Input Parameters

real xfrmt(3,3)


Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• None

SET GLOBAL TRANSFORMATION 3 (PHOP,*,STOP,*)

Purpose

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Use Set Global Transformation 3 to insert a three-dimensional Set Global Transformation 3 structure element into the open structure following the element pointer, or replace the element pointed at by the element pointer with a Set Global Transformation 3 structure element, depending on the current edit mode.

When encountered during traversal, this element causes the specified matrix to replace the current global transformation for the current structure. The resultant matrix, in conjunction with the local modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

Language Bindings

C

pset_global_tran3 (global_tran)

Input Parameters

Pmatrix3 global_tran


FORTRAN

PSGMT3 (xfrmt)

Input Parameters

real xfrmt(4,4)


Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

• None

SET LOCAL TRANSFORMATION (PHOP,*,STOP,*)

Purpose

Use Set Local Transformation to insert a two-dimensional Set Local Transformation structure element into the open structure following the element pointer, or replace the element pointed at by the element pointer with a Set Local Transformation structure element, depending on the current edit mode.

When the graPHIGS API encounters this element during traversal, the graPHIGS API expands it into a 4 X 4 matrix as follows:

```
<table>
<thead>
<tr>
<th>a  b  c</th>
</tr>
</thead>
<tbody>
<tr>
<td>d  e  f</td>
</tr>
<tr>
<td>g  h  i</td>
</tr>
</tbody>
</table>
```

Chapter 10. Transformation Subroutines 137
Depending on the composition type, when the graPHIGS API encounters this element during traversal, the specified matrix replaces, is pre-concatenated with, or is post-concatenated with the current local modeling transformation matrix. The resultant matrix, in conjunction with the global modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

Language Bindings

C

`pset_local_tran (local_tran, compose_type)`

Input Parameters

- **Pmatrix local_tran**

- **Pcompose_type compose_type**
  - Composition type (0=PTYPE_PRECONCAT, 1=PTYPE_POSTCONCAT, 2=PTYPE_REPLACE).

FORTRAN

`PSLMT (xfrmt, ctype)`

Input Parameters

- **real xfrmt(3,3)**

- **integer ctype**
  - Composition type (0=PCPRE, 1=PCPOST, 2=PCREPL).

Errors

5  Function Requires State (PHOP,*;STOP,*)

Related Subroutines

- None

**SET LOCAL TRANSFORMATION 3 (PHOP,*;STOP,*)**

Purpose

Use Set Local Transformation 3 to insert a three-dimensional Set Local Transformation 3 structure element into the open structure following the element pointer, or replace the element pointed at by the element pointer with a Set Local Transformation 3 structure element, depending on the current edit mode.

Depending on the composition type, when the graPHIGS API encounters this element during traversal, the specified matrix replaces, is pre-concatenated with, or is post-concatenated with the current local modeling transformation matrix. The resultant matrix, in conjunction with the global modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

Language Bindings
C

\texttt{pset\_local\_tran3 (local\_tran, compose\_type)}

\textbf{Input Parameters}

\textit{Pmatrix3 local\_tran}


\textit{Pcompose\_type compose\_type}

Composition type ($0=$PTYPE\_PRECONCAT, $1=$PTYPE\_POSTCONCAT, $2=$PTYPE\_REPLACE).

\textbf{FORTRAN}

\texttt{PSLMT3 (xfrmt, ctype)}

\textbf{Input Parameters}

\textit{real xfrmt}(4,4)


\textit{integer ctype}

Composition type ($0=$PCPRE, $1=$PCPOST, $2=$PCREPL).

\textbf{Errors}

5 \quad Function Requires State (PHOP,*,STOP,*).

\textbf{Related Subroutines}

- None

\textbf{SET MODELING CLIPPING VOLUME (PHOP,*,STOP,*)}

\textbf{Purpose}

Use Set Modeling Clipping Volume to insert a two-dimensional Set Modeling Clipping Volume structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Volume structure element, depending on the current edit mode.

This structure element specifies the current modeling clipping volume. Each modeling clipping half-space contains a point and a vector defined in modeling coordinates (MC). The graPHIGS API expands each two-dimensional half-space to a three-dimensional half-space by setting the $z$ coordinate of both the point and the vector to the value $0.0$. The current modeling transformation transforms each pair of half-spaces (consisting of a point and vector) from the Modeling Coordinate (MC) system to the World Coordinate (WC) system, and defines a boundary (plane) in WC. The transformed point is on this plane and the transformed vector defines a normal to the plane which points into the acceptance half-space region. The clipping volume is obtained by intersecting all acceptance half-spaces in the list specified by this element.

During structure traversal, the volume specified by this element either replaces or intersects the current modeling clipping volume, depending on the value specified by the modeling clipping operator parameter. At structure traversal time, the graPHIGS API uses the resultant clipping volume to render subsequent primitives. Transformation elements encountered during traversal do not affect the resultant clipping volume. The resultant volume defines the acceptance region. The graPHIGS API clips portions of subsequent primitives that are outside of the acceptance region.
If the number of modeling clipping half-spaces is set to zero, then the acceptance region is all of world coordinate space (WC) and no clipping occurs.

During traversal, if the workstation does not support the specified modeling clipping operator, if the specified number of clipping half-spaces exceeds the maximum supported by the workstation, or if any half-space is found to be degenerate, then the graPHIGS API ignores this structure element.

During traversal, if the graPHIGS API encounters a Set Modeling Clipping Volume structure element and the current composite modeling transformation matrix is singular, then the graPHIGS API sets the effective clipping volume to the null volume and clips all subsequent primitives.

Language Bindings

C

pset_model_clip_vol (op, half_spaces)

Input Parameters

Pint op

Operator (1=REPLACE, 2=INTERSECT).

const Phalf_space_list *half_spaces

List of half-spaces.

FORTRAN

PSMCV (op, nhalfs, halfsp)

Input Parameters

integer op

Operator (1=PMCREP, 2=PMCINT).

integer nhalfs

Number of half-spaces in the list.

real halfsp(4,nhalfs)

List of half-spaces.

For the $i$th modeling clipping half-space:

- $HALFSP(1,i)$ is the x component of the point.
- $HALFSP(2,i)$ is the y component of the point.
- $HALFSP(3,i)$ is the Delta x component of the normal vector.
- $HALFSP(4,i)$ is the Delta y component of the normal vector.

Errors

5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines

- Inquire Modeling Clipping Facilities
- Set Modeling Clipping Indicator

SET MODELING CLIPPING VOLUME 3 (PHOP,*,STOP,*)

Purpose

140  The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Use Set Modeling Clipping Volume 3 to insert a three-dimensional Set Modeling Clipping Volume 3 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Volume 3 structure element, depending on the current edit mode.

This element specifies the current modeling clipping volume. Each modeling clipping half-space contains a point and a vector defined in modeling coordinates (MC). The current modeling transformation transforms each pair of half-spaces (consisting of a point and vector) from the Modeling Coordinate (MC) system to the World Coordinate (WC) system, and defines a boundary (plane) in WC. The transformed point is on this plane and the transformed vector defines a normal to the plane which points into the acceptance half-space region. The clipping volume is obtained by intersecting all acceptance half-spaces in the list specified by this element.

During structure traversal, the volume specified by this element either replaces or intersects the current modeling clipping volume, depending on the value specified by the modeling clipping operator parameter. At structure traversal time, the grafHIGS API uses the resultant volume to render subsequent primitives. Transformation elements encountered do not affect the resultant clipping volume. The resultant volume defines the acceptance region. The grafHIGS API clips portions of subsequent primitives that are outside of the acceptance region.

If the number of modeling clipping half-spaces is set to zero, then the acceptance region is all of world coordinate space (WC) and no clipping occurs.

During traversal, if the workstation does not support the specified modeling clipping operator, if the specified number of clipping half-spaces exceeds the maximum supported by the workstation, or if any half-space is found to be degenerate, then the grafHIGS API ignores this structure element.

During traversal, if the grafHIGS API encounters a Set Modeling Clipping Volume 3 structure element and the current composite modeling transformation matrix is singular, then the grafHIGS API sets the effective clipping volume to the null volume and clips all subsequent primitives.

Language Bindings

C

pset_model_clip_vol3 (op, half_spaces)

Input Parameters

Pint op
  Operator (1=REPLACE, 2=INTERSECT).

const Phalf_space_list3 * half_spaces
  List of half-spaces.

FORTRAN

PSMCV3 (op, nhalfs, halfsp)

Input Parameters

integer op
  Operator (1=PMCREP, 2=PMCINT).

integer nhalfs
  Number of half-spaces in the list.
real halfsp(6,nhalfs)
List of half-spaces.
For the \(i\)th modeling clipping half-space:
- \(HALFSP(1,i)\) is the x component of the point.
- \(HALFSP(2,i)\) is the y component of the point.
- \(HALFSP(3,i)\) is the z component of the point.
- \(HALFSP(4,i)\) is the Delta x component of the normal vector.
- \(HALFSP(5,i)\) is the Delta y component of the normal vector.
- \(HALFSP(6,i)\) is the Delta z component of the normal vector.

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
- Inquire Modeling Clipping Facilities
- Set Modeling Clipping Indicator

**SET MODELING CLIPPING INDICATOR (PHOP,*,STOP,*)**

**Purpose**
Use Set Modeling Clipping Indicator to insert a Set Modeling Clipping Indicator structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Indicator structure element, depending on the current edit mode.

Use this subroutine to set the value of the current modeling clipping indicator. During structure traversal, the graPHIGS API uses this value to determine whether or not to perform modeling clipping on subsequent output primitives.

The traversal default for the modeling clipping indicator is **NOCLIP**.

**Note:** Not all graPHIGS API workstations support modeling clipping. Use the graPHIGS API Inquire Workstation Description (GPQWDT) subroutine to determine if a particular workstation supports modeling clipping.

**Language Bindings**

**C**

```c
pset_model_clip_ind (clip_ind)
```

**Input Parameters**

\(Pclip\_ind\) \(clip\_ind\)
Clipping indicator (0=PIND_NO_CLIP, 1=PIND_CLIP).

**FORTRAN**

```fortran
PSMCLI (mclip)
```

**Input Parameters**
integer mclipi
Modeling clipping indicator (0=PNCLIP, 1=PCLIP)

Errors
5 Function Requires State (PHOP,*,STOP,*)

Related Subroutines
• Inquire Modeling Clipping Facilities
• Set Modeling Clipping Volume
• Set Modeling Clipping Volume 3

SET WORKSTATION VIEWPORT (PHOP,WSOP,*,*)

Purpose
Use Set Workstation Viewport to set the requested two-dimensional workstation viewport for the specified workstation.

The graPHIGS API sets the x, y components of the current workstation viewport to the requested values when you update the workstation. The graPHIGS API does not change the z coordinates of the requested workstation viewport or the current workstation viewport.

Language Bindings

C

pset_ws_vp (ws_id, ws_vp_limits)

Input Parameters
Pint ws_id
   Workstation identifier.
const Plimit *ws_vp_limits
   Workstation viewport limits in DC.

FORTRAN

PSWKV (wkid, xmin, xmax, ymin, ymax)

Input Parameters
integer wkid
   Workstation identifier.
real xmin
   Minimum workstation viewport limit on the x-axis in DC.
real xmax
   Maximum workstation viewport limit on the x-axis in DC.
real ymin
   Minimum workstation viewport limit on the y-axis in DC.
real ymax
   Maximum workstation viewport limit on the y-axis in DC.
Errors

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
57 Specified Workstation Is Of Category MI
152 Invalid Viewport: XMIN >= XMAX, YMIN >= YMAX OR ZMIN > ZMAX
157 Workstation Viewport Is Not Within Display Space

Related Subroutines

- Inquire Workstation Transformation
- Inquire Workstation Transformation 3
- Set Workstation Viewport 3
- Set Workstation Window
- Set Workstation Window 3

SET WORKSTATION VIEWPORT 3 (PHOP,WSOP,*,*)

Purpose

Use Set Workstation Viewport 3 to set the requested three-dimensional workstation viewport for the specified workstation.

The graPHIGS API sets the current workstation viewport to the requested values when you update the workstation.

Language Bindings

C

pset_ws_vp3 (ws_id, ws_vp_limits)

Input Parameters

Pint ws_id
  Workstation identifier.

const Plimit3 *ws_vp_limits
  Workstation viewport limits in DC.

FORTRAN

PSWKV3 (wkid, wkvp)

Input Parameters

integer wkid
  Workstation identifier.

real wkvp(6)
  Workstation viewport limits in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

Errors

3 Function Requires State (PHOP,WSOP,*,*)
Specified Workstation Is Not Open

Specified Workstation Is Of Category MI

Invalid Viewport: XMIN >= XMAX, YMIN >= YMAX OR ZMIN > ZMAX

Workstation Viewport Is Not Within Display Space

Related Subroutines
- Inquire Workstation Transformation
- Inquire Workstation Transformation 3
- Set Workstation Viewport
- Set Workstation Window
- Set Workstation Window 3

SET WORKSTATION WINDOW (PHOP,WSOP,*,*)

Purpose

Use Set Workstation Window to set the requested two-dimensional workstation window for the specified workstation.

The graPHIGS API sets the x, y components of the current workstation window to the requested values when you update the workstation. The graPHIGS API does not change the z coordinates of the requested workstation window or the current workstation window.

Language Bindings

C

pset_ws_win (ws_id, ws_win_limits)

Input Parameters

Pint ws_id
    Workstation identifier.

const Plimit *ws_win_limits
    Workstation window limits in NPC.

FORTRAN

PSWKW (wkid, xmin, xmax, ymin, ymax)

Input Parameters

integer wkid
    Workstation identifier.

real xmin
    Minimum workstation window limit on the x-axis in NPC.

real xmax
    Maximum workstation window limit on the x-axis in NPC.

real ymin
    Minimum workstation window limit on the y-axis in NPC.
**real ymax**

Maximum workstation window limit on the y-axis in NPC.

**Errors**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Function Requires State (PHOP,WSOP,<em>,</em>)</td>
</tr>
<tr>
<td>54</td>
<td>Specified Workstation Is Not Open</td>
</tr>
<tr>
<td>57</td>
<td>Specified Workstation Is Of Category MI</td>
</tr>
<tr>
<td>151</td>
<td>Invalid Window: Minimum Value &gt;= To Corresponding Maximum Value</td>
</tr>
<tr>
<td>156</td>
<td>Workstation Window Limits Are Not Within NPC Range</td>
</tr>
</tbody>
</table>

**Related Subroutines**

- Inquire Workstation Transformation
- Inquire Workstation Transformation 3
- Set Workstation Viewport
- Set Workstation Viewport 3
- Set Workstation Window 3

---

### SET WORKSTATION WINDOW 3 (PHOP,WSOP,*,*)

**Purpose**

Use Set Workstation Window 3 to set the requested three-dimensional workstation window for the specified workstation.

The graPHIGS API sets the current workstation window to the requested values when you update the workstation.

**Language Bindings**

**C**

```c
pset_ws_win3 (ws_id, ws_win_limits)
```

**Input Parameters**

- `Pint ws_id`
  - Workstation identifier.
- `const Plimit3 *ws_win_limits`
  - Workstation window limits in NPC.

**FORTRAN**

```fortran
PSWKW3 (wkid, wkwn)
```

**Input Parameters**

- `integer wkid`
  - Workstation identifier.
- `real wkwn(6)`
  - Workstation window limits in NPC (`XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX`).
Errors

3 Function Requires State (PHOP, WSOP, *, * )
54 Specified Workstation Is Not Open
57 Specified Workstation Is Of Category MI
151 Invalid Window: Minimum Value >= To Corresponding Maximum Value
156 Workstation Window Limits Are Not Within NPC Range

Related Subroutines

• Inquire Workstation Transformation 3
• Set Workstation Viewport
• Set Workstation Viewport 3
• Set Workstation Window
Chapter 11. Input Subroutines

Input subroutines allow users to supply input to your application. There are six logical input device classes: locator, stroke, valuator, choice, pick, and string. There are three modes of interaction with the input devices: sample, request, and event.

The subroutines discussed in this section perform the following operations:

• initialization of an input device
• setting the operating mode of an input device
• requesting input from a device
• sampling an input device’s current value
• managing the event queue
• retrieving input values from the event queue

To determine the actual input capabilities of a specific workstation, use the appropriate inquiry subroutines.

The measure and trigger of each logical input device is described in terms of the physical devices available on a workstation. See The graPHIGS Programming Interface: Technical Reference for the details of each logical device supported on a workstation.

The default size of the input queue is 16K. You can control this size with the IQSIZE default, which is explained in the The graPHIGS Programming Interface: Technical Reference.

The input queue elements all have a header of 16 bytes. Use the information supplied in Appendix B of The graPHIGS Programming Interface: Technical Reference to interpret the information after the header.

AWAIT EVENT (PHOP, WSOP,*,*)

Purpose

Use Await Event to move the next event from the input queue into the current event report. If the input queue is empty, then the graPHIGS API is placed in a wait state until at least one of the following occurs:

1. The application adds an event to the input queue.
2. The time specified in the timeout parameter has elapsed.

If the timeout parameter specifies a value of zero or less, then no wait takes place. If the timeout parameter specifies a value greater than zero, then a wait takes place for the specified time interval. The maximum time interval is 55,800 seconds (15.5 hours).

The graPHIGS API uses the operating system’s timing facility. (See The graPHIGS Programming Interface: Writing Applications).

When a timeout or error situation occurs, the graPHIGS API returns NONE for the input class parameter. Otherwise, the graPHIGS API returns the workstation identifier, input class, and the logical device number.

Input classes include: LOCATOR, STROKE, VALUATOR, CHOICE, PICK, and STRING.

The application must use the appropriate “Get” subroutine call to obtain the value(s) of the input residing in the current event report (i.e., Get Choice, Get Locator 3, Get String).

Language Bindings
C

pawait_event(timeout, ws_id, dev_class, in_num)

Input Parameters

*Pfloat timeout
  Timeout interval in seconds.

Output Parameters

*Pint *ws_id
  Workstation identifier.

*Pint *dev_class
  Input device class (0=PIN_NONE, 1=PIN_LOC, 2=PIN_STROKE, 3=PIN_VAL, 4=PIN_CHOICE, 5=PIN_PICK, 6=PIN_STRING).

*Pint *in_num
  Logical input device number.

FORTRAN

PWAIT (tout, wkid, icl, idnr)

Input Parameters

realtout
  Timeout interval in seconds.

Output Parameters

integer wkid
  Workstation identifier.

integer icl
  Input device class (0=PNCLAS, 1=PLOCAT, 2=PSTROK, 3=PVALUA, 4=PCHOIC, 5=PPICK, 6=PSTRIN).

integer idnr
  Logical input device number.

Errors

3  Function Requires State (PHOP,WSOP,*,*)

256 Warning, Input Queue Has Overflowed

  Note: The graPHIGS API performs this operation even if error 256 occurs.

Related Subroutines

- Flush Device Events
- Get Choice
- Get Locator
- Get Locator 3
- Get Pick
- Get String
- Get Stroke
• Get Stroke 3
• Get Valuator
• Inquire Input Queue Overflow
• Inquire More Simultaneous Events
• Set Choice Mode
• Set Locator Mode
• Set Pick Mode
• Set String Mode
• Set Stroke Mode
• Set Valuator Mode

FLUSH DEVICE EVENTS (PHOP,WSOP,*,*)

Purpose

Use Flush Device Events to discard all input events from the specified logical input device.

This subroutine removes all events received from the specified input device, matching the specified device class, workstation identifier and device number, from the event queue. Input classes include: LOCATOR, STROKE, VALUATOR, CHOICE, PICK, and STRING.

If the current event report includes an event matching the specified input device, then the graPHIGS API also removes the current event report.

Language Bindings

C

pflush_events (ws_id, dev_class, in_num)

Input Parameters

Pint ws_id
  Workstation identifier.
Pin_class dev_class
  Input device class (0=PIN_NONE, 1=PIN_LOC, 2=PIN_STROKE, 3=PIN_VAL, 4=PIN_CHOICE, 5=PIN_PICK, 6=PIN_STRING).
Pint in_num
  Logical input device number (>=1).

FORTRAN

PFLUSH (wkid, icl, idnr)

Input Parameters

integer wkid
  Workstation identifier.
integer icl
  Input device class (1=PLOCAT, 2=PSTROK, 3=PVALUA, 4=PCHOIC, 5=PPICK, 6=PSTRIN).
integer idnr
    Logical input device number (>=1).

Errors
3    Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
61   Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
256  Warning, Input Queue Has Overflowed

    Note: The graPHIGS API performs this operation even if error 256 occurs.

Related Subroutines
• Inquire Number Of Available Logical Input Devices

GET CHOICE (PHOP,WSOP,*,*)

Purpose
Use Get Choice to retrieve a choice input value from the current event report. The graPHIGS API identified
the device to which this value corresponds on the previous invocation of the Await Event subroutine call
Await Event The graPHIGS API does not remove the event from the current event report until the next
invocation of Await Event.

This subroutine returns a status parameter of OK or NOCHOICE.

Language Bindings

C

pget_choice (in_status, choice)

Output Parameters

Pin_status *in_status
    Choice status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Pint *choice
    Choice number (>=1).

FORTRAN

PGTCH (stat, chnr)

Output Parameters

integer stat
    Choice status (1=POK, 2=PNCHOI).

integer chnr
    Choice number (>=1).

Errors
Function Requires State (PHOP, WSOP, *, *)
Requested Device Class Not Current Input Report Class

Related Subroutines
• Await Event

GET LOCATOR (PHOP, WSOP, *, *)

Purpose
Use Get Locator to retrieve a locator input value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine call.

The view index indicates the view table entry which has a matrix that the graPHIGS API used to convert the locator point from Device Coordinates (DC) to World Coordinates (WC). This was the view with the highest input priority at the indicated screen location.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event Await Event.

This subroutine returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

Note: This subroutine returns a two-dimensional result. The graPHIGS API discards the z coordinate of the locator position. The x and y values of the locator position are identical to those returned by the Get Locator 3 subroutine for the same operator action.

Language Bindings

C

pget_loc (view_ind, loc_pos)

Output Parameters
Pint *view_ind
View index.
Ppoint *loc_pos
Locator position in WC.

FORTRAN

PGTLC (viewi, lpx, lpy)

Output Parameters
integer viewi
View index.
real lpx
x coordinate of the locator position in WC.
real lpy
y coordinate of the locator position in WC.
Errors
3  Function Requires State (PHOP, WSOP, *, *)
259  Requested Device Class Not Current Input Report Class

Related Subroutines
- Await Event
- Get Locator 3
- Set View Transformation Input Priority

**GET LOCATOR 3 (PHOP, WSOP, *, *)**

**Purpose**

Use Get Locator 3 to retrieve a locator input value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine call.

The view index indicates the view table entry which has a matrix that the graPHIGS API used to convert the locator point from Device Coordinates (DC) to World Coordinates (WC). This was the view with the highest input priority at the indicated screen location.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of the Await Event subroutine call.

This subroutine returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

**Language Bindings**

**C**

```c
pget_loc3 (view_ind, loc_pos)
```

**Output Parameters**

- `Pint *view_ind`
  View index.
- `Ppoint3 *loc_pos`
  Locator position in WC.

**FORTRAN**

```fortran
PGTLC3 (viewi, lpx, lpy, lpz)
```

**Output Parameters**

- `integer viewi`
  View index.
- `real lpx`
  x coordinate of the locator position in WC.
- `real lpy`
  y coordinate of the locator position in WC.
real lpz
  z coordinate of the locator position in WC.

Errors
3     Function Requires State (PHOP,WSOP,*,*)
259   Requested Device Class Not Current Input Report Class

Related Subroutines
• Await Event
• Get Locator
• Set View Transformation Input Priority

GET PICK (PHOP,WSOP,*,*)

Purpose
Use Get Pick to retrieve a pick input value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine.

This value consists of a pick status and pick path information describing the position of the picked primitive in the structure network. The pick status may be OK or NOPICK. The graPHIGS API returns the pick path in the order specified in the the Initialize Pick subroutine, that is, TOP FIRST or BOTTOM FIRST. If your application has not called the Initialize Pick subroutine, then the pick path order defaults to TOP FIRST. Each entry in the pickpath consists of a structure identifier, a pick identifier, and an element position.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event.

Language Bindings

C

pget_pick (depth, in_status, pick)

Input Parameters
Pint depth
  Maximum depth of the pick path to return.

Output Parameters
Pin_status *in_status
  Pick status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Ppick_path *pick
  Pick path.

FORTRAN

PGTPK (ippd, stat, ppd, pp)

Input Parameters
integer ippd
  Maximum depth of the pick path to return.
Output Parameters

`integer stat`
Pick status (1=POK, 2=PNPICK).

`integer ppd`
Depth of the actual pick path.

`integer pp(3,ippd)`
Pick path.

Errors

3  Function Requires State (PHOP,WSOP,*,*
259  Requested Device Class Not Current Input Report Class

Related Subroutines

- Await Event

GET STRING (PHOP,WSOP,*,*)

Purpose

Use Get String to retrieve a string input value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine call.

The length of the returned string is less than, or equal to, the buffer size found in the string data record at the time the device was set to event mode.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event.

Language Bindings

C

`pget_string (string)`

Output Parameters

`char *string`
String.

FORTRAN

PGTST `(lostr, str)`

Output Parameters

`integer lostr`
Number of characters returned.

`character(*) str`
String.
FORTRAN Subset

PGTST (lostr, str)

Output Parameters

integer lostr
   Number of characters returned.

character*80 str
   String.

Errors

3   Function Requires State (PHOP,WSOP,*,*)
259  Requested Device Class Not Current Input Report Class

Related Subroutines
• Await Event

GET STROKE (PHOP,WSOP,*,*)

Purpose

Use Get Stroke to retrieve a stroke input device value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine call.

The graPHIGS API limits the number of returned points to the current input buffer size found in the stroke data record at the time the device was set to event mode. The view index indicates the view table entry which has a matrix that the graPHIGS API used to convert the stroke points from Device Coordinates (DC) to World Coordinates (WC). This view was the view with the highest input priority containing all the stroke locations.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event.

View zero is the highest priority view unless modified by your application.

Note: This function returns a two-dimensional result. The graPHIGS API discards the z coordinate of the stroke points. The x and y values of the stroke points are identical to those returned by the Get Stroke 3 subroutine for the same operator action.

Language Bindings

C

pget_stroke (view_ind, stroke)

Output Parameters

Pint *view_ind
   View index.

Ppoint_list *stroke
   Stroke.
FORTRAN

PGTK (n, viewi, np, pxa, pya)

Input Parameters

integer n
   Dimension of arrays for the stroke points (>=0).

Output Parameters

integer viewi
   View index.

integer np
   Number of points.

real pxa (n)
   x coordinates of the points in the stroke in WC.

real pya (n)
   y coordinates of the points in the stroke in WC.

Errors

3   Function Requires State (PHOP,WSOP,*,*)
259 Requested Device Class Not Current Input Report Class

Related Subroutines

• Await Event
• Get Stroke 3
• Set View Transformation Input Priority

GET STROKE 3 (PHOP,WSOP,*,*)

Purpose

Use Get Stroke 3 to retrieve a stroke input device value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the the Await Event subroutine call.

The graPHIGS API limits the number of returned points to the current input buffer size found in the stroke data record at the time the device was set to event mode. The view index indicates the view table entry which has a matrix that the graPHIGS API used to convert the stroke points from Device Coordinates (DC) to World Coordinates (WC). This view was the view with the highest input priority containing all the stroke locations.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event.

View zero is the highest priority view unless modified by your application.

Language Bindings

C

pget_stroke3 (view_ind, stroke)
Output Parameters

\textit{Pint} *\textit{view\_ind}

View index.

\textit{Ppoint\_list3} *\textit{stroke}

Stroke.

\textbf{FORTRAN}

\texttt{PGTSK3 (n, viewi, np, pxa, pya, pza)}

\textbf{Input Parameters}

\textit{integer} \textit{n}

Dimension of arrays for the stroke points (\(\geq 0\)).

\textbf{Output Parameters}

\textit{integer} \textit{viewi}

View index.

\textit{integer} \textit{np}

Number of points.

\textit{real} \textit{pxa} (\textit{n})

\(x\) coordinates of the points in the stroke in WC.

\textit{real} \textit{pya} (\textit{n})

\(y\) coordinates of the points in the stroke in WC.

\textit{real} \textit{pza} (\textit{n})

\(z\) coordinates of the points in the stroke in WC.

\textbf{Errors}

\textit{3} Function Requires State (PHOP,WSOP,\*,\*)

\textit{259} Requested Device Class Not Current Input Report Class

\textbf{Related Subroutines}

- Await Event
- Get Stroke
- Set View Transformation Input Priority

\textbf{GET VALUATOR (PHOP,WSOP,\*,\*)}

\textbf{Purpose}

Use Get Valuator to retrieve a valuator input value from the current event report. The graPHIGS API does not remove the event from the current event report until the next invocation of the Await Event subroutine call.

The graPHIGS API returns a value within the range found in the valuator data record at the time the device was set to event mode.

The graPHIGS API identified the device to which this value corresponds on the previous invocation of Await Event.
Language Bindings

C
pget_val (value)

Output Parameters

float *value
Valuator value.

FORTRAN
PGTVL (val)

Output Parameters

real val
Valuator value.

Errors
3 Function Requires State (PHOP,WSOP,*,*)
259 Requested Device Class Not Current Input Report Class

Related Subroutines

• Await Event

INITIALIZE CHOICE (PHOP,WSOP,*,*)

Purpose

Use Initialize Choice to specify initial values for the specified choice device.

The Initialize Choice subroutine stores the initial choice number, prompt/echo type, echo area, and data record in the workstation state list for the specified device. The z coordinates of the echo volume remain unchanged. For a keyboard choice device, an initial choice number less than 256 is interpreted using the workstation’s input device character set. For details on the specific devices available on different workstation types, see The graPHIGS Programming Interface: Technical Reference, or use the appropriate inquiry subroutines.

This function supports the following prompt/echo types:

• Type One designates the current choice number using a workstation-dependent technique.
• Type Two lets you indicate choice numbers by invoking the prompting capability. The physical input devices that are most commonly used normally have a built-in prompting capability, such as lighted program function keys (LPFKs). If the value of the ith element of the “prompt array” in the choice data record is OFF, then the graPHIGS API turns off prompting of the ith alternative of the specified choice input device. If the value of the ith element of the “prompt array” in the choice data record is ON, then the graPHIGS API turns on prompting of the ith alternative of the specified choice input device. The first entry in the choice data record is the list of choice prompts.

Note: The choice device must be in Request mode.

Language Bindings
C

pinit_choice (ws_id, choice_num, init_status, init_choice, pet, echo_area, choice_data)

Input Parameters

Pint ws_id
Workstation identifier.

Pint choice_num
Choice device number (>=1).

Pin_status init_status
Initial choice status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Pint init_choice
Initial choice device number (>=1).

Pint pet
Prompt and echo type.

const Plimit *echo_area
Echo area.

const Pchoice_data *choice_data
Data record.

FORTRAN

PINCH (wkid, chdnr, istat, ichnr, pet, xmin, xmax, ymin, ymax, ldr, datrec)

Input Parameters

integer wkid
Workstation identifier.

integer chdnr
Choice device number (>=1).

integer istat
Initial choice status (1=POK, 2=PNCHOI).

integer ichnr
Initial choice device number (>=1).

integer pet
Prompt and echo type.

real xmin
Minimum x coordinate determining echo area in DC.

real xmax
Maximum x coordinate determining echo area in DC.

real ymin
Minimum y coordinate determining echo area in DC.

real ymax
Maximum y coordinate determining echo area in DC.

integer ldr
Dimension of the data record array.
character*80 datrec(ldr)
   Data record.

The PPREC parameters to build a choice input data record for PET=2 are as follows:
(IL=number of choice alternatives, IA=list of prompts(0=POFF, 1=PON), RL=0, RA=(), SL=0, LSTR=(), STR=()).

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input/Echo Type Not Available On Specified Workstation
261 Initial Value Is Invalid

Related Subroutines

- Inquire Choice Device State
- Inquire Default Choice Device Data
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Choice Mode

**INITIALIZE CHOICE 3 (PHOP,WSOP,*,*)**

**Purpose**

Use Initialize Choice 3 to specify initial values for a specified choice device.

The Initialize Choice 3 subroutine stores the initial choice number, prompt/echo type, echo volume, and data record in the workstation state list for the specified device. For a keyboard choice device, an initial choice number less than 256 is interpreted using the workstation’s input device character set. For details on the specific devices available on different workstation types, see *The graPHIGS Programming Interface: Technical Reference*, or use the appropriate inquiry subroutines.

This function supports the following prompt/echo types:

- Type One designates the current choice number using a workstation-dependent technique.
- Type Two lets you indicate choice numbers by invoking the prompting capability. The physical input devices that are most commonly used normally have a built-in prompting capability, such as lighted program function keys (LPFKs). If the value of the \(i^{th}\) element of the “prompt array” in the choice data record is *OFF*, then the graPHIGS API turns off prompting of the \(i^{th}\) alternative of the specified choice input device. If the value of the \(i^{th}\) element of the “prompt array” in the choice data record is *ON*, then the graPHIGS API turns on prompting of the \(i^{th}\) alternative of the specified choice input device. The first entry in the choice data record is the list of choice prompts.
Note: The choice device must be in Request mode.

Language Bindings

C

pinit_choice3 (ws_id, choice_num, init_status, init_choice, pet, echo_vol, choice_data)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint choice_num
   Choice device number (>=1).

Pin_status init_status
   Initial choice status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Pint init_choice
   Initial choice device number (>=1).

Pint pet
   Prompt and echo type.

const Plimit3 *echo_vol
   Echo volume in DC.

const Pchoice_data3 *choice_data
   Data record.

FORTRAN

PINCH3 (wkid, chdnr, istat, ichnr, pet, evol, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.

integer chdnr
   Choice device number (>=1).

integer istat
   Initial choice status (1=POK, 2=PNCHOI).

integer ichnr
   Initial choice device number (>=1).

integer pet
   Prompt and echo type.

real evol(6)
   Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Dimension of the data record array.

character*80 datrec(ldr)
   Data record.
The **PPREC** parameters to build a **choice input data record** for **PET=2** are as follows:

(\(IL=n\)umber of choice alternatives, \(IA=li\)st of prompts (0=POFF, 1=PON), RL=0, RA=(), SL=0, LSTR=(), STR=()).

**Errors**

3 Function Requires State (PHOP,WSOP,*,*)

54 Specified Workstation Is Not Open

61 Specified Workstation Is Not Of Category Input Or Outin

250 Specified Device Not Available On Workstation

251 Function Requires Input Device To Be In Request Mode

254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX

255 Echo Area/Volume Boundary Point(s) Outside Device Range

253 Prompt/Echo Type Not Available On Specified Workstation

260 Input Device Data Record Field Is In Error

261 Initial Value Is Invalid

**Related Subroutines**

- Inquire Choice Device State 3
- Inquire Default Choice Device Data 3
- Inquire Display Space Size 3
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Choice Mode

**INITIALIZE LOCATOR (PHOP,WSOP,*,*)**

**Purpose**

Use Initialize Locator to initialize the specified locator device.

The Initialize Locator subroutine stores the initial locator position, initial view index, prompt/echo type, echo area, and locator data record in the workstation state list for the specified device. The \(z\) coordinates of both the echo volume and initial locator position remain unchanged.

Two positions are required for some locator prompt/echo types: the initial locator position, which remains fixed during input operation, and the current locator position, which varies dynamically as you use the locator.

This function supports the following prompt/echo types:

- Type One designates the current position of the locator using a workstation-dependent technique.
- Type Two, the cross hair, designates the current position of the locator by spanning the display surface or device echo area with both a vertical and a horizontal line. The lines intersect at the current locator position. Whether the cross hair spans the entire display surface or only the echo area depends on the capabilities of the workstation.
- Type Three designates the current position of the locator using a tracking cross.
- Type Four designates the current position of the locator using a rubber band line connecting the initial locator position given by this subroutine and the current locator position.
Note: The locator device must be in Request mode.

Language Bindings

C

pinit_loc (ws_id, loc_num, init_view_ind, init_loc_pos, pet, echo_area, loc_data)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint loc_num
    Locator device number (>=1).

Pint init_view_ind
    Initial view index (>=0).

const Ppoint *init_loc_pos
    Initial locator position in WC.

Pint pet
    Prompt and echo type.

cconst Plimit *echo_area
    Echo area in DC.

cconst Ploc_data *loc_data
    Data record.

FORTRAN

PINLC (wkid, lcdnr, iviewi, ipx, ipy, pet, xmin, xmax, ymin, ymax, ldr, datrec)

Input Parameters

integer wkid
    Workstation identifier.

integer lcdnr
    Locator device number (>=1).

integer iviewi
    Initial view index (>=0).

real ipx
    x coordinate of the initial locator position in WC.

real ipy
    y coordinate of the initial locator position in WC.

integer pet
    Prompt and echo type.

real xmin
    Minimum x coordinate determining echo area in DC.

real xmax
    Maximum x coordinate determining echo area in DC.

real ymin
    Minimum y coordinate determining echo area in DC.
real ymax
Maximum y coordinate determining echo area in DC.

integer ldr
Dimension of the data record array.

c character*80 datrec(ldr)
Data record.

The PPREC parameters to build a **locator input data record** for PET=4 are as follows: (IL=7, IA=unused, linetype ASF, line width scale factor ASF, polyline color index ASF, polyline index, line type, polyline color index, RL=1, RA=line width scale factor, SL=0, LSTR=(), STR=() )

**Errors**

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid
114 View Index Value < ZERO

**Related Subroutines**
- Inquire Locator Device State
- Inquire Default Locator Device Data
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Locator Mode

**INITIALIZE LOCATOR 3 (PHOP,WSOP,*,*)**

**Purpose**

Use Initialize Locator 3 to initialize the specified locator device.

The Initialize Locator 3 subroutine stores the initial locator position, initial view index, prompt/echo type, echo volume, and locator data record in the workstation state list for the specified device.

Two positions are required for some locator prompt/echo types: the initial locator position, which remains fixed during input operation, and the current locator position, which varies dynamically as you use the locator.

This function supports the following prompt/echo types:
- Type One designates the current position of the locator using a workstation-dependent technique.
• Type Two, the cross hair, designates the current position of the locator by spanning the display surface or device echo area with both a vertical and a horizontal line. The lines intersect at the current locator position. Whether the cross hair spans the entire display surface or only the echo area depends on the capabilities of the workstation.

• Type Three designates the current position of the locator using a tracking cross.

• Type Four designates the current position of the locator using a rubber band line connecting the initial locator position given by this subroutine and the current locator position.

**Note:** The locator device must be in Request mode.

**Language Bindings**

**C**

```c
pinit_loc3 (ws_id, loc_num, init_view_ind, init_loc_pos, pet, echo_vol, loc_data)
```

**Input Parameters**

- `ws_id`  
  Workstation identifier.

- `loc_num`  
  Locator device number (>=1).

- `init_view_ind`  
  Initial view index (>=0).

- `init_loc_pos`  
  Initial locator position in WC.

- `pet`  
  Prompt and echo type.

- `echo_vol`  
  Echo volume in DC.

- `loc_data`  
  Data record.

**FORTRAN**

```fortran
PINLC3 (wkid, lcdnr, iviewi, ipx, ipy, ipz, pet, evol, ldr, datrec)
```

**Input Parameters**

- `wkid`  
  Workstation identifier.

- `lcdnr`  
  Locator device number (>=1).

- `iviewi`  
  Initial view index (>=0).

- `ipx`  
  x coordinate of the initial locator position in WC.

- `ipy`  
  y coordinate of the initial locator position in WC.
real ipz
  z coordinate of the initial locator position in WC.

integer pet
  Prompt and echo type.

real evol(6)
  Echo volume (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
  Dimension of the data record array.

character*80 datrec(ldr)
  Data record.

The PPREC parameters to build a locator input data record for PET=4 are as follows: (IL=7, IA=unused, linetype ASF, line width scale factor ASF, polyline color index ASF, polyline index, line type, polyline color index, RL=1, RA=line width scale factor, SL=0, LSTR=(), STR=() ).

Errors
3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid
114 View Index Value < ZERO

Related Subroutines
• Inquire Locator Device State 3
• Inquire Default Locator Device Data 3
• Inquire Display Space Size
• Inquire Number Of Available Logical Input Devices
• Pack Data Record
• Set Locator Mode

INITIALIZE PICK (PHOP,WSOP,*,*)

Purpose
Use Initialize Pick to initialize the specified pick device.

The Initialize Pick subroutine stores the prompt/echo type, echo area, initial pick path depth, initial pick path, pick data record and pick path order in the workstation state list for the specified device. The z coordinates of the echo volume remain unchanged.
The pick status may be initialized to OK or NOPICK. The pick path order is the order in which the graPHIGS API returns the elements of the pick path. If you specify the pick path order as TOP FIRST, then the structure specified in any pick path element is a parent of the structure specified in the subsequent pick path element. If you specify the pick path order as BOTTOM FIRST, then the structure specified in any pick path element is a child of the structure specified in the subsequent pick path element. Each pick path element consists of a structure identifier, a pick identifier, and an element position.

This function supports the following prompt/echo types:

- Type One uses a workstation-dependent technique that highlights the picked primitive. The graPHIGS API does not require a data record.

  **Note:** The pick device must be in Request Mode.

**Language Bindings**

**C**

```c
pinit_pick (ws_id, pick_num, init_status, init_pick, pet, echo_area, pick_data, order)
```

**Input Parameters**

- `Pint ws_id`  
  Workstation identifier.
- `Pint pick_num`  
  Pick device number (>=1).
- `Pint init_status`  
  Initial pick status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).
- `const Ppick_path *init_pick`  
  Initial pick path.
- `Pint pet`  
  Prompt and echo type.
- `const Plimit *echo_area`  
  Echo area in DC.
- `const Ppick_data *pick_data`  
  Data record.
- `Ppath_order order`  
  Pick path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

**FORTRAN**

```fortran
PINPK (wkid, pkdnr, istat, ippd, pp, pet, xmin, xmax, ymin, ymax, ldr, datrec, ppordh)
```

**Input Parameters**

- `integer wkid`  
  Workstation identifier.
- `integer pkdnr`  
  Pick device number (>=1).
- `integer istat`  
  Initial pick status (1=POK, 2=PNPICK).
integer ippd  
    Depth of initial pick path.

integer pp(3,ippd)  
    Initial pick path.

integer pet  
    Prompt and echo type.

real xmin  
    Minimum x coordinate determining echo area in DC.

real xmax  
    Maximum x coordinate determining echo area in DC.

real ymin  
    Minimum y coordinate determining echo area in DC.

real ymax  
    Maximum y coordinate determining echo area in DC.

integer ldr  
    Dimension of the data record array.

character*80 datrec(ldr)  
    Data record.

integer ppordr  
    Pick path order (0=PPOTOP, 1=PPOBOT).

Errors

3  Function Requires State (PHOP,WSOP,*,*)

54  Specified Workstation Is Not Open

60  Specified Workstation Is Not Of Category Input Or Outin

250  Specified Device Not Available On Workstation

251  Function Requires Input Device To Be In Request Mode

254  Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX

255  Echo Area/Volume Boundary Point(s) Outside Device Range

253  Prompt/Echo Type Not Available On Specified Workstation

260  Input Device Data Record Field Is In Error

261  Initial Value Is Invalid

Related Subroutines

• Add Names To Set
• Inquire Pick Device State
• Inquire Default Pick Device Data
• Inquire Display Space Size
• Inquire Number Of Available Logical Input Devices
• Pack Data Record
• Remove Names From Set
• Set Pick Filter
• Set Pick Identifier
• Set Pick Mode
INITIALIZE PICK 3 (PHOP,WSOP,*,*)

Purpose

Use Initialize Pick 3 to initialize the specified pick device.

The Initialize Pick 3 subroutine stores the prompt/echo type, echo volume, initial pick path depth, initial pick path, pick data record and pick path order in the workstation state list for the specified device.

The pick status may be initialized to OK or NOPICK. The pick path order is the order in which the graPHIGS API returns elements of the pick path. If you specify the pick path order as TOP FIRST, then the structure specified in any pick path element is a parent of the structure specified in the subsequent pick path element. If you specify the pick path order as BOTTOM FIRST, then the structure specified in any pick path element is a child of the structure specified in the subsequent pick path element. Each pickpath element consists of a structure identifier, a pick identifier, and an element position.

This function supports the following prompt/echo types:

- Type One uses a workstation-dependent technique that highlights the picked primitive. The graPHIGS API does not require a data record.

  Note: The pick device must be in Request Mode.

Language Bindings

C

pinit_pick3 (ws_id, pick_num, init_status, init_pick, pet, echo_vol, pick_data, order)

Input Parameters

Pint ws_id
  Workstation identifier.

Pint pick_num
  Pick device number (>=1).

Pint init_status
  Initial pick status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

const Ppick_path *init_pick
  Initial pick path.

Pint pet
  Prompt and echo type.

const Plimit3 *echo_vol
  Echo volume in DC.

const Ppick_data3 *pick_data
  Data record.

Ppath_order order
  Pick path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

FORTRAN

PINPK3 (wkid, pkdnr, istat, ippd, pp, pet, evol, ldr, datrec, ppordr)
Input Parameters

integer wkid
   Workstation identifier.

integer pkdnr
   Pick device number (>=1).

integer istat
   Initial pick status (1=POK, 2=PNPICK).

integer ippd
   Depth of initial pick path.

integer pp(3,ippd)
   Initial pick path.

integer pet
   Prompt and echo type.

real evol(6)
   Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Dimension of the data record array.

c character*80 datrec(ldr)
   Data record.

integer ppordr
   Pick path order (0=PPOTOP, 1=PPOBOT).

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
60  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid

Related Subroutines

• Add Names To Set
• Inquire Pick Device State 3
• Inquire Default Pick Device Data 3
• Inquire Display Space Size
• Inquire Number Of Available Logical Input Devices
• Pack Data Record
• Remove Names From Set
• Set Pick Filter
• Set Pick Identifier
INITIALIZE STRING (PHOP, WSOP, *, *)

Purpose

Use Initialize String to initialize the specified string input device.

The Initialize String subroutine stores the initial string, prompt/echo type, echo area, and string data record in the workstation state list for the specified device. The z coordinates of the echo volume remain unchanged. The graPHIGS API uses the string device’s input character set to interpret the initial string.

For all prompt and echo types, the first entry of the string data record is the input buffer size, which is an integer in the range (1..n). The graPHIGS API compares this against the available input buffer size for the specified string device. If the requested buffer size is greater, then the graPHIGS API records the available size in the workstation state list instead of the specified input buffer size. If the initial string is longer than the buffer size, then the graPHIGS API issues an error.

When the graPHIGS API receives string input, it obtains a buffer of the size defined by the input buffer size. The graPHIGS API copies the initial string into the buffer, and places the cursor at the initial editing position within the buffer. Replacement of characters begins at this initial position.

This function supports the following prompt/echo types:

- Type One displays the current string value within the echo area using a workstation-dependent technique.

  **Note:** The string device must be in Request mode.

Language Bindings

C

```c
pinit_string (ws_id, string_num, init_string, pet, echo_area, string_data)
```

Input Parameters

- **Pint ws_id**
  Workstation identifier.

- **Pint string_num**
  String device number (>=1).

- **const char *init_string**
  Initial string.

- **Pint pet**
  Prompt and echo type.

- **const Plimit *echo_area**
  Echo area in DC.

- **const Pstring_data *string_data**
  Data record.

FORTRAN

```fortran
PINST (wkid, stdnr, lstr, istr, pet, xmin, xmax, ymin, ymax, ldr, datrec)
```
**Input Parameters**

*integer wkid*
   
   Workstation identifier.

*integer stdnr*
   
   String device number (>=1).

*integer lstr*
   
   Length of the initial string (>=0). The number of characters actually used is the minimum of `lstr` and the length of `istr`.

*character*(*) `istr`
   
   Initial string.

*integer pet*
   
   Prompt and echo type.

*real xmin*
   
   Minimum x coordinate determining echo area in DC.

*real xmax*
   
   Maximum x coordinate determining echo area in DC.

*real ymin*
   
   Minimum y coordinate determining echo area in DC.

*real ymax*
   
   Maximum y coordinate determining echo area in DC.

*integer ldr*
   
   Dimension of the data record array.

*character*80 `datrec(ldr)`

   Data record.

   The `pprec` parameters used to build the string input data record are as follows: (IL=2, IA=\textit{input buffer size}, \textit{initial editing position}, RL=0, RA=(), SL=0, LSTR=(), STR=()) .

**FORTRAN Subset**

`PINST (wkid, stdnr, lstr, istr, pet, xmin, xmax, ymin, ymax, ldr, datrec)`

**Input Parameters**

*integer wkid*
   
   Workstation identifier.

*integer stdnr*
   
   String device number (>=1).

*integer lstr*
   
   Length of the initial string.

*character*80 `istr`
   
   Initial string.

*integer pet*
   
   Prompt and echo type.

*real xmin*
   
   Minimum x coordinate determining echo area in DC.

*real xmax*
   
   Maximum x coordinate determining echo area in DC.
real ymin
Minimum y coordinate determining echo area in DC.

real ymax
Maximum y coordinate determining echo area in DC.

integer ldr
Dimension of the data record array.

character*80 datrec(ldr)
Data record.
The pprec parameters used to build the string input data record are as follows: ( IL=2, IA=input buffer size, initial editing position, RL=0, RA=(), SL=0, LSTR=(), STR=() )

Errors
3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>XMAX, YMIN>YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid
263 Length Of Initial String > Buffer Size

Related Subroutines
• Inquire String Device State
• Inquire Default String Device Data
• Inquire Display Space Size
• Inquire Number Of Available Logical Input Devices
• Pack Data Record
• Set String Mode

INITIALIZE STRING 3 (PHOP,WSOP,*,*)

Purpose
Use Initialize String 3 to initialize the specified string input device.

The Initialize String 3 subroutine stores the initial string, prompt/echo type, echo volume, and string data record in the workstation state list for the specified device. The graPHIGS API uses the string device’s input character set to interpret the initial string and prompt strings.

For all prompt and echo types, the first entry of the string data record is the input buffer size, which is an integer in the range (1..n). The graPHIGS API compares this against the available input buffer size for the specified string device. If the requested buffer size is greater, then the graPHIGS API records the available size in the workstation state list instead of the specified input buffer size. If the initial string is longer than the buffer size, then the graPHIGS API issues an error.
When the graPHIGS API receives string input, it obtains a buffer of the size defined by the input buffer size. The graPHIGS API copies the initial string into the buffer, and places the cursor at the initial editing position within the buffer. Replacement of characters begins at this initial position.

This function supports the following prompt/echo types:

- Type One displays the current string value within the echo area using a workstation-dependent technique.

Note: The string device must be in Request mode.

**Language Bindings**

**C**

```c
pinit_string3 (ws_id, string_num, init_string, pet, echo_vol, string_data)
```

**Input Parameters**

- `Pint ws_id` 
  Workstation identifier.
- `Pint string_num` 
  String device number (>=1).
- `const char *init_string` 
  Initial string.
- `Pint pet` 
  Prompt and echo type.
- `const Plimit3 *echo_vol` 
  Echo volume in DC.
- `const Pstring_data3 *string_data` 
  Data record.

**FORTRAN**

```fortran
PINST3 (wkid, stdnr, lstr, istr, pet, evol, ldr, datrec)
```

**Input Parameters**

- `integer wkid` 
  Workstation identifier.
- `integer stdnr` 
  String device number (>=1).
- `integer lstr` 
  Length of the initial string (>=0). The number of characters actually used is the minimum of `lstr` and the length of `istr`.
- `character(*) istr` 
  Initial string.
- `integer pet` 
  Prompt and echo type.
- `real evol(6)` 
  Echo volume in DC (`XMIN`, `XMAX`, `YMIN`, `YMAX`, `ZMIN`, `ZMAX`).
integer ldr
   Dimension of the data record array.

character*80 datrec(ldr)
   Data record.

   The *prec parameters used to build the **string input data record** are as follows: (IL=2, IA=**input buffer size, initial editing position, RL=0, RA=(), SL=0, LSTR=(), STR=() ).

FORTRAN Subset

PINST3 (wkid, stdnr, lstr, istr, pet, evol, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.

integer stdnr
   String device number (>=1).

integer lstr
   Length of the initial string.

character*80 istr
   Initial string.

integer pet
   Prompt and echo type.

real evol(6)
   Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Dimension of the data record array.

character*80 datrec(ldr)
   Data record.

   The *prec parameters used to build the **string input data record** are as follows: (IL=2, IA=**input buffer size, initial editing position, RL=0, RA=(), SL=0, LSTR=(), STR=() ).

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid
263 Length Of Initial String > Buffer Size
Related Subroutines
- Inquire String Device State 3
- Inquire Default String Device Data 3
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set String Mode

**INITIALIZE STROKE (PHOP,WSOP,*,*)**

**Purpose**

Use Initialize Stroke to initialize the specified stroke device.

The Initialize Stroke subroutine stores the initial stroke, initial view index, prompt/echo type, echo area and stroke data record in the workstation state list for the specified device. The z coordinates of the echo volume remain unchanged.

For all prompt and echo types, the first entry of the stroke data record is the input buffer size, which is an integer in the range (1..n). The graPHIGS API compares this against the available input buffer size for the specified stroke device. If the requested buffer size is greater, then the graPHIGS API records the available size in the workstation state list instead of the specified input buffer size. If the initial stroke is longer than the buffer size, then the graPHIGS API issues an error.

When a stroke measure process begins, it acquires a buffer of the current input buffer size. The graPHIGS API copies the initial stroke pointlist into this buffer, and places the editing position at the initial buffer editing position. The replacement of points begins at this initial position. The x, y, and time intervals (where possible) of the data record control the frequency and density of stroke points.

This function supports the following prompt/echo types:
- Type One displays the current stroke using a workstation-dependent technique.
- Type Three displays a marker at each point of the current stroke. The marker representation is selected by a marker index, which is stored in the stroke data record.
- Type Four displays a line joining successive points in the current stroke. A polyline index in the stroke data record selects the line representation used.

**Note:** The stroke device must be in Request mode.

**Language Bindings**

**C**

```c
pinit_stroke (ws_id, stroke_num, init_view_ind, init_stroke, pet, echo_area, stroke_data)
```

**Input Parameters**

**Pint ws_id**

Workstation identifier.

**Pint stroke_num**

Stroke device number (>=1).

**Pint init_view_ind**

Initial view index (>=0).
const Ppoint_list *init_stroke
  Initial stroke in WC.

Pint pet
  Prompt and echo type.

const Plimit *echo_area
  Echo area in DC.

const Pstroke_data *stroke_data
  Data record.

FORTRAN

PINSK (wkid, skdnr, iviewi, n, ipx, ipy, pet, xmin, xmax, ymin, ymax, ldr, datrec)

Input Parameters

integer wkid
  Workstation identifier.

integer skdnr
  Stroke device number (>=1).

integer iviewi
  Initial view index (>=0).

integer n
  Number of coordinates of initial stroke.

real ipx(*)
  x coordinates of initial stroke in WC. The actual arguments are dimensioned by at least max(1,n).

real ipy(*)
  y coordinates of initial stroke in WC. The actual arguments are dimensioned by at least max(1,n).

integer pet
  Prompt and echo type.

real xmin
  Minimum x coordinate determining echo area in DC.

real xmax
  Maximum x coordinate determining echo area in DC.

real ymin
  Minimum y coordinate determining echo area in DC.

real ymax
  Maximum y coordinate determining echo area in DC.

integer ldr
  Dimension of the data record array.

character*80 datrec(ldr)
  Data record.

The pprec parameters used to build the stroke input data record for pet= 1 are as follows: (IL=2, IA=input buffer size, editing position, RL=3, RA=x interval, y interval, time interval in seconds, SL=0, LSTR=(), STR=() ).

The pprec parameters used to build the stroke input data record for pet=3 are as follows: (IL=9, IA=input buffer size, editing position, unused, marker type ASF, marker size scale factor ASF,
polymarker color index ASF, polymarker index, marker type, polymarker color index, RL=4, RA=x interval, y interval, time interval in seconds, marker size scale factor, SL=0, LSTR=(), STR=() ).

The pprec parameters used to build the stroke input data record for pet=4 are as follows: (IL=9, IA=input buffer size, editing position, unused, line type ASF, line width scale factor ASF, polyline color index ASF, polyline index, line type, polyline color index, RL=4, RA=x interval, y interval, time interval in seconds, line width scale factor, SL=0, LSTR=(), STR=() ).

Errors

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
251  Function Requires Input Device To Be In Request Mode
254  Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255  Echo Area/Volume Boundary Point(s) Outside Device Range
253  Prompt/Echo Type Not Available On Specified Workstation
260  Input Device Data Record Field Is In Error
261  Initial Value Is Invalid
262  Number Of Points In Initial Stroke > Buffer Size
114  View Index Value < ZERO

Related Subroutines

- Inquire Stroke Device State 3
- Inquire Default Stroke Device Data 3
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Stroke Mode

**INITIALIZE STROKE 3 (PHOP,WSOP,*,*)**

Purpose

Use Initialize Stroke 3 to initialize the specified stroke device.

The Initialize Stroke 3 subroutine stores the initial stroke, initial view index, prompt/echo type, echo volume and stroke data record in the workstation state list for the specified device.

For all prompt and echo types, the first entry of the stroke data record is the input buffer size, which is an integer in the range(1..n). The graPHIGS API compares this against the available input buffer size for the specified stroke device. If the requested buffer size is greater, then the graPHIGS API records the available size in the workstation state list instead of the specified input buffer size. If the initial stroke is longer than the buffer size, then the graPHIGS API issues an error.

When a stroke measure process begins, it acquires a buffer of the current input buffer size. The graPHIGS API copies the initial stroke pointlist into this buffer, and places the editing position at the initial buffer editing position. The replacement of points begins at this initial position. The x, y, z, and time intervals (where possible) of the data record control the frequency and density of stroke points.
This function supports the following prompt/echo types:

- Type One displays the current stroke using a workstation-dependent technique.
- Type Three displays a marker at each point of the current stroke. The marker representation is selected by a marker index, which is stored in the stroke data record.
- Type Four displays a line joining successive points in the current stroke. A polyline index in the stroke data record selects the line representation used.

**Note:** The stroke device must be in Request mode.

**Language Bindings**

**C**

```c
pinit_stroke3 (ws_id, stroke_num, init_view_ind, init_stroke, pet, echo_vol, stroke_data)
```

**Input Parameters**

- `Pint ws_id`  
  Workstation identifier.
- `Pint stroke_num`  
  Stroke device number (>=1).
- `Pint init_view_ind`  
  Initial view index (>=0).
- `const Ppoint_list3 *init_stroke`  
  Initial stroke in WC.
- `Pint pet`  
  Prompt and echo type.
- `const Plimit3 *echo_vol`  
  Echo volume in DC.
- `const Pstroke_data3 *stroke_data`  
  Data record.

**FORTRAN**

```fortran
PINSK3 (wkid, skdnr, iviewi, n, ipx, ipy, ipz, pet, evol, ldr, datrec)
```

**Input Parameters**

- `integer wkid`  
  Workstation identifier.
- `integer stdnr`  
  Stroke device number (>=1).
- `integer iviewi`  
  Initial view index (>=0).
- `integer n`  
  Number of coordinates of initial stroke.
- `real ipx (*)`  
  x coordinates of initial stroke in WC. The actual arguments are dimensioned by at least `max(1,n)`.
- `real ipy (*)`  
  y coordinates of initial stroke in WC. The actual arguments are dimensioned by at least `max(1,n)`. 
real ipz (*)
z coordinates of initial stroke in WC. The actual arguments are dimensioned by at least max(1,n).

integer pet
Prompt and echo type.

real evol(6)
Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
Dimension of the data record array.

character*80 datrec(ldr)
Data record.

The pprec parameters used to build the stroke input data record for pet= 1 are as follows: (IL=2, IA=input buffer size, editing position, RL=3, RA=x interval, y interval, time interval in seconds, SL=0, LSTR=(), STR=() ).

The pprec parameters used to build the stroke input data record for pet= 3 are as follows: ( IL=9, IA=input buffer size, editing position, unused, marker type ASF, marker size scale factor ASF, polymarker color index ASF, polymarker index, marker type, polymarker color index, RL=4, RA=x interval, y interval, time interval in seconds, marker size scale factor, SL=0, LSTR=(), STR=() ).

The pprec parameters used to build the stroke input data record for pet= 4 are as follows: (IL=9, IA=input buffer size, editing position, unused, line type ASF, line width scale factor ASF, polyline color index ASF, polyline index, line type, polyline color index, RL=5, RA=x interval, y interval, time interval in seconds, line width scale factor, SL=0, LSTR=(), STR=() ).

Errors
3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid
262 Number Of Points In Initial Stroke > Buffer Size
114 View Index Value < ZERO

Related Subroutines
• Inquire Stroke Device State 3
• Inquire Default Stroke Device Data 3
• Inquire Display Space Size
• Inquire Number Of Available Logical Input Devices
• Pack Data Record
• Set Stroke Mode
INITIALIZE VALUATOR (PHOP,WSOP,*,*)

Purpose

Use Initialize Valuator to initialize the specified valuator device.

The Initialize Valuator subroutine stores the initial value, prompt/echo type, echo area, and valuator data record in the workstation state list for the specified workstation. The z coordinates of the echo volume remain unchanged.

For all valuator prompt/echo types a low value and a high value specify the range for input from that valuator. The graPHIGS API scales the values from the physical device linearly to the specified range.

This function supports the following prompt/echo types:
• Type One designates the current valuator value using a workstation-dependent technique.
• Type Three displays a numerical representation of the current valuator value within the echo area.

Note: The valuator device must be in Request mode.

Language Bindings

C

pinit_val (ws_id, val_num, init_value, pet, echo_area, val_data)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint val_num
   Valuator device number (>=1).

Pfloat init_value
   Initial value.

Pint pet
   Prompt and echo type.

const Plimit *echo_area
   Echo area in DC.

const Pval_data *val_data
   Data record.

FORTRAN

PINVL (wkid, vldnr, ival, pet, xmin, xmax, ymin, ymax, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.

integer vldnr
   Valuator device number (>=1).

real ival
   Initial value.
integer pet
   Prompt and echo type.

real xmin
   Minimum x coordinate determining echo area in DC.

real xmax
   Maximum x coordinate determining echo area in DC.

real ymin
   Minimum y coordinate determining echo area in DC.

real ymax
   Maximum y coordinate determining echo area in DC.

integer ldr
   Dimension of the data record array.

character*80 datrec(ldr)
   Data record.

   The pprec parameters used to build the valuator input data record are as follows: (IL=0, IA=(), RL=2, RA=low value of valuator range, high value of valuator range, SL=0, LSTR=(), STR=() ).

Errors

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode
254 Invalid Echo Area/Volume: XMIN>=XMAX, YMIN>=YMAX OR ZMIN>ZMAX
255 Echo Area/Volume Boundary Point(s) Outside Device Range
253 Prompt/Echo Type Not Available On Specified Workstation
260 Input Device Data Record Field Is In Error
261 Initial Value Is Invalid

Related Subroutines
- Inquire Valuator Device State
- Inquire Default Valuator Device Data
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Valuator Mode

INITIALIZE VALUATOR 3 (PHOP,WSOP,*,*)

Purpose

Use Initialize Valuator 3 to initialize the specified valuator device.

The Initialize Valuator subroutine stores the initial value, prompt/echo type, echo volume, and valuator data record in the workstation state list for the specified workstation.
For all valuator prompt/echo types a low value and a high value specify the range for input from that valuator. The graPHIGS API scales the values from the physical device linearly to the specified range. This function supports the following prompt/echo types:

- Type One designates the current valuator value using a workstation-dependent technique.
- Type Three displays a numerical representation of the current valuator value within the echo volume.

**Note:** The valuator device must be in Request mode.

**Language Bindings**

**C**

```c
pinit_val3 (ws_id, val_num, init_value, pet, echo_vol, val_data)
```

**Input Parameters**

- `Pint ws_id`
  Workstation identifier.
- `Pint val_num`
  Valuator device number (>=1).
- `Pfloat init_value`
  Initial value.
- `Pint pet`
  Prompt and echo type.
- `const Plimit3 *echo_vol`
  Echo volume in DC.
- `const Pval_data3 *val_data`
  Data record.

**FORTRAN**

```fortran
PINVL3 (wkid, vdlnr, ival, pet, evol, ldr, datrec)
```

**Input Parameters**

- `integer wkid`
  Workstation identifier.
- `integer vdlnr`
  Valuator device number (>=1).
- `real ival`
  Initial value.
- `integer pet`
  Prompt and echo type.
- `real evol(6)`
  Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).
- `integer ldr`
  Dimension of the data record array.
- `character*80 datrec(ldr)`
  Data record.
The \textit{pprec} parameters used to build the \textbf{valuator input data record} are as follows: \((IL=0, IA=(), RL=2, RA=\text{low value of valuator range}, \text{high value of valuator range}, SL=0, LSTR=(), STR=() )\).

\textbf{Errors}

\begin{itemize}
  \item 3 \hspace{1em} Function Requires State \((PHOP,WSOP,*,*\))
  \item 54 \ Specified Workstation Is Not Open
  \item 61 \ Specified Workstation Is Not Of Category Input Or Outin
  \item 250 \ Specified Device Not Available On Workstation
  \item 251 \ Function Requires Input Device To Be In Request Mode
  \item 254 \ Invalid Echo Area/Volume: \(XMIN>=XMAX, YMIN>=YMAX \ OR \ ZMIN>ZMAX\)
  \item 255 \ Echo Area/Volume Boundary Point(s) Outside Device Range
  \item 253 \ Prompt/Echo Type Not Available On Specified Workstation
  \item 260 \ Input Device Data Record Field Is In Error
  \item 263 \ Length Of Initial String > Buffer Size
  \item 261 \ Initial Value Is Invalid
\end{itemize}

\textbf{Related Subroutines}

- Inquire Valuator Device State 3
- Inquire Default Valuator Device Data
- Inquire Display Space Size
- Inquire Number Of Available Logical Input Devices
- Pack Data Record
- Set Valuator Mode

\textbf{REQUEST CHOICE (PHOP,WSOP,***)}

\textbf{Purpose}

Use Request Choice to have the graPHIGS API execute a request to the specified choice device. The graPHIGS API returns the choice input value, which is the current measure of the choice device.

A status of \textit{NONE} means that a break action occurred. If the measure of the choice device indicates no choice, then the graPHIGS API returns a status of \textit{NOCHOICE}. Otherwise, the graPHIGS API returns a status of \textit{OK} together with a choice number which the graPHIGS API sets according to the current measure of the choice device.

\textbf{Language Bindings}

\textbf{C}

\texttt{preq\_choice (ws\_id, choice\_num, in\_status, choice)}

\textbf{Input Parameters}

\begin{itemize}
  \item \texttt{Pint ws\_id}
    \hspace{1em}Workstation identifier.
  \item \texttt{Pint choice\_num}
    \hspace{1em}Choice device number (\(\geq 1\)).
\end{itemize}
Output Parameters

\textit{Pin\_status *in\_status}

Choice status \((0=\text{PIN\_STATUS\_NONE}, 1=\text{PIN\_STATUS\_OK}, 2=\text{PIN\_STATUS\_NO\_IN})\).

\textit{Pint *choice}

Requested choice.

\textbf{FORTRAN}

\texttt{PRQCH (wkid, chdnr, stat, chnr)}

\textbf{Input Parameters}

\textit{integer wkid}

Workstation identifier.

\textit{integer chdnr}

Choice device number \((>=1)\).

\textbf{Output Parameters}

\textit{integer stat}

Choice status \((0=\text{PNONE}, 1=\text{POK}, 2=\text{PNCHOI})\).

\textit{integer chnr}

Choice number \((>=1)\).

\textbf{Errors}

\begin{itemize}
  \item \texttt{3} Function Requires State \((\text{PHOP,WSOP,*,*})\)
  \item \texttt{54} Specified Workstation Is Not Open
  \item \texttt{61} Specified Workstation Is Not Of Category Input Or Outin
  \item \texttt{250} Specified Device Not Available On Workstation
  \item \texttt{251} Function Requires Input Device To Be In Request Mode
\end{itemize}

\textbf{Related Subroutines}

\begin{itemize}
  \item Initialize Choice
  \item Inquire Number Of Available Logical Input Devices
  \item Set Choice Mode
\end{itemize}

\textbf{REQUEST LOCATOR (PHOP,WSOP,*,*)}

\textbf{Purpose}

Use Request Locator to have the graPHIGS API execute a request to the specified locator device.

The graPHIGS API returns the locator position and the index of the view which the graPHIGS API used to convert the location from Device Coordinates (DC) to World Coordinates (WC).

The graPHIGS API returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

A status of \textit{NONE} means that a break action occurred. Otherwise, the graPHIGS API returns a status of \textit{OK} together with the logical input value which is the current measure of the locator device.
**Note:** This function returns a two-dimensional result. The graPHIGS API discards the $z$ coordinate of the locator position. The $x$ and $y$ values of the locator position are identical to those returned by the Request Locator 3 subroutine for the same operator action.

**Language Bindings**

**C**

```c
preq_loc (ws_id, loc_num, in_status, view_ind, loc_pos)
```

**Input Parameters**

- `Pint ws_id`
  - Workstation identifier.
- `Pint loc_num`
  - Locator device number ($\geq 1$).

**Output Parameters**

- `Pin_status *in_status`
  - Input status ($0 = \text{PIN\_STATUS\_NONE}$, $1 = \text{PIN\_STATUS\_OK}$).
- `Pint *view_ind`
  - View index ($\geq 0$).
- `Ppoint *loc_pos`
  - Locator position in WC.

**FORTRAN**

```fortran
PRQLC (wkid, lcdnr, stat, viewi, px, py)
```

**Input Parameters**

- `integer wkid`
  - Workstation identifier.
- `integer lcdnr`
  - Locator device number ($\geq 1$).

**Output Parameters**

- `integer stat`
  - Input status ($0 = \text{PNONE}$, $1 = \text{POK}$).
- `integer viewi`
  - View index ($\geq 0$).
- `real px`
  - $x$ coordinate of the locator position in WC.
- `real py`
  - $y$ coordinate of the locator position in WC.

**Errors**

- **3** Function Requires State (PHOP,WSOP,*,*)
- **54** Specified Workstation Is Not Open
- **61** Specified Workstation Is Not Of Category Input Or Outin

**188** The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Specified Device Not Available On Workstation

Function Requires Input Device To Be In Request Mode

Related Subroutines

- Initialize Locator
- Inquire Number Of Available Logical Input Devices
- Set Locator Mode

REQUEST LOCATOR 3 (PHOP,WSOP,*,*)

Purpose

Use Request Locator 3 to have the graPHIGS API execute a request to the specified locator device.

The graPHIGS API returns the locator position and the index of the view which the graPHIGS API used to convert the location from Device Coordinates (DC) to World Coordinates (WC).

The graPHIGS API returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

A status of NONE means that a break action occurred. Otherwise, the graPHIGS API returns a status of OK together with the logical input value which is the current measure of the locator device.

Language Bindings

C

`preq_loc3 (ws_id, loc_num, in_status, view_ind, loc_pos)`

Input Parameters

- `Pint ws_id`: Workstation identifier.
- `Pint loc_num`: Locator device number (>=1).

Output Parameters

- `Pin_status *in_status`: Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK).
- `Pint *view_ind`: View index (>=0).
- `Ppoint3 *loc_pos`: Locator position in WC.

FORTRAN

`PRQLC3 (wkid, lcdnr, stat, viewi, px, py, pz)`

Input Parameters

- `integer wkid`: Workstation identifier.
integer lcdnr
    Locator device number (>=1).

Output Parameters
integer stat
    Input status (0=PNONE, 1=POK).
integer viewi
    View index (>=0).
real px
    x coordinate of the locator position in WC.
real py
    y coordinate of the locator position in WC.
real pz
    z coordinate of the locator position in WC.

Errors
3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation
251   Function Requires Input Device To Be In Request Mode

Related Subroutines
• Initialize Locator 3
• Inquire Number Of Available Logical Input Devices
• Set Locator Mode

REQUEST PICK (PHOP,WSOP,*,*)

Purpose
Use Request Pick to have the graPHIGS API execute a request to the specified pick device.

The graPHIGS API returns the pick path information in the order specified in the Initialize Pick subroutine, that is, TOP FIRST or BOTTOM FIRST. If your application has not called Initialize Pick, then the pick path defaults to TOP FIRST.

A status of NONE means that a break action occurred. If the measure of the pick device indicates no pick, then the graPHIGS API returns a status of NOPICK. Otherwise, the graPHIGS API returns a status of OK together with a pick path which the graPHIGS API sets according to the current measure of the pick device. Each entry in the pick path consists of a structure identifier, a pick identifier, and an element position.

Language Bindings

C

preq_pick (ws_id, pick_num, depth, in_status, pick)

Input Parameters
190  The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Pint ws_id
    Workstation identifier.

Pint pick_num
    Pick device number (>=1).

Pint depth
    Maximum depth of the returned pick path.

Output Parameters
Pin_status *in_status
    Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Ppick_path *pick
    Requested pick path.

FORTRAN
PRQPK (wkid, pkdnr, ippd, stat, ppd, pp)

Input Parameters
integer wkid
    Workstation identifier.

integer pkdnr
    Pick device number (>=1).

integer ippd
    Maximum depth of the returned pick path.

Output Parameters
integer stat
    Input status (0=PNONE, 1=POK, 2=PNPICK).

integer ppd
    Depth of the actual pick path.

integer pp (3,ippd)
    Requested pick path.

Errors
3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation
251   Function Requires Input Device To Be In Request Mode

Related Subroutines
• Add Names To Set
• Initialize Pick
• Inquire Number Of Available Logical Input Devices
• Remove Names From Set
• Set Pick Filter
REQUEST STRING (PHOP, WSOP, *, *)

Purpose

Use Request String to have the graPHIGS API execute a request to the specified string device.

A status of NONE means that a break action occurred. Otherwise, the graPHIGS API returns a status of OK together with a character string which is set according to the current measure of the string device.

Language Bindings

C

prev_string (ws_id, string_num, in_status, string)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint string_num
   String device number (>=1).

Output Parameters

Pin_status *in_status
   Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK).

c <string
   Requested string. The application must allocate the memory for the character string returned. The Initialize String subroutine or the Initialize String 3 subroutine specifies the maximum size of the returned character string. The Inquire Default String Device Data subroutine or the Inquire Default String Device Data 3 subroutine returns the maximum size of a character string supported by the workstation.

FORTRAN

PRQST (wkid, stdnr, stat, lostr, str)

Input Parameters

integer wkid
   Workstation identifier.

integer stdnr
   String device number (>=1).

Output Parameters

integer stat
   Input status (0=PNONE, 1=POK).

integer lostr
   Number of characters returned.
character(*) str
   Character string.

FORTRAN Subset
PRQST (wkid, stdnr, stat, lostr, str)

Input Parameters
integer wkid
   Workstation identifier.
integer stdnr
   String device number (>=1).

Output Parameters
integer stat
   Input status (0=PNONE, 1=POK).
integer lostr
   Number of characters returned.
character*80 str
   Character string.

Errors
3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
251 Function Requires Input Device To Be In Request Mode

Related Subroutines
- Initialize String
- Inquire Number Of Available Logical Input Devices
- Set String Mode

REQUEST STROKE (PHOP,WSOP,*,*)

Purpose
Use Request Stroke to have the graPHIGS API execute a request to the specified stroke device.

A status of NONE means that a break action occurred. Otherwise, the graPHIGS API returns a status of OK together with a sequence of World Coordinate (WC) points and the view table index which has a matrix that the graPHIGS API used to convert the stroke locations from Device Coordinates (DC) to World Coordinates. The graPHIGS API sets these values according to the current measure of the stroke device.

This subroutine returns the stroke input from the view with the highest input priority that contains all the points. View zero is the highest priority view unless modified by your application.

Note: This function returns a two-dimensional result. The graPHIGS API discards the z coordinates of the stroke points. The x and y values of the stroke points are identical to those returned by the Request Stroke 3 subroutine for the same operator action.
Language Bindings

C

`preq_stroke (ws_id, stroke_num, in_status, view_ind, stroke)`

Input Parameters

`Pint ws_id`

Workstation identifier.

`Pint stroke_num`

Stroke device number (>=1).

Output Parameters

`Pin_status *in_status`

Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK).

`Pint *view_ind`

View index.

`Ppoint_list *stroke`

Requested stroke point list. The application must allocate the memory for the point list returned. The Initialize Stroke subroutine specifies the maximum size of the returned stroke point list. The Inquire Default Stroke Device Data subroutine returns the maximum size of a stroke point list supported by the workstation.

FORTRAN

`PRQSK (wkid, skdnr, n, stat, viewi, np, pxa, pya)`

Input Parameters

`integer wkid`

Workstation identifier.

`integer skdnr`

Stroke device number (>=1).

`integer n`

Dimension of arrays for stroke points.

Output Parameters

`integer stat`

Input status (0=PNONE, 1=POK).

`integer viewi`

View index (>=0).

`integer np`

Number of points.

`real pxa (n)`

x coordinates of points in the stroke in WC.

`real pya (n)`

y coordinates of points in the stroke in WC.

Errors
Function Requires State (PHOP, WSOP, *, *)

Specified Workstation Is Not Open

Specified Workstation Is Not Of Category Input Or Outin

Specified Device Not Available On Workstation

Function Requires Input Device To Be In Request Mode

Related Subroutines
- Initialize Stroke
- Inquire Number Of Available Logical Input Devices
- Set Stroke Mode

REQUEST STROKE 3 (PHOP, WSOP, *, *)

Purpose
Use Request Stroke 3 to have the graPHIGS API execute a request to the specified stroke device.

A status of NONE means that a break action occurred. Otherwise, the graPHIGS API returns a status of OK together with a sequence of World Coordinate (WC) points and the view table index which has a matrix that the graPHIGS API used to convert the stroke locations from Device Coordinates (DC) to World Coordinates. The graPHIGS API sets these values according to the current measure of the stroke device.

This subroutine returns the stroke input from the view with the highest input priority that contains all the points. View zero is the highest priority view unless modified by your application.

Language Bindings

C

preq_stroke3 (ws_id, stroke_num, in_status, view_ind, stroke)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint stroke_num
   Stroke device number (>=1).

Output Parameters

Pin_status *in_status
   Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK).

Pint *view_ind
   View index.

Ppoint_list3 *stroke
   Requested stroke point list. The application must allocate the memory for the point list returned. The Initialize Stroke 3 subroutine specifies the maximum size of the returned stroke point list. The Inquire Default Stroke Device Data subroutine returns the maximum size of a stroke point list supported by the workstation.
FORTRAN

PRQSK3 (wkid, skdnr, n, stat, viewi, np, pxa, pya, pza)

Input Parameters

integer wkid
  Workstation identifier.

integer skdnr
  Stroke device number (>=1).

integer n
  Dimension of arrays for stroke points.

Output Parameters

integer stat
  Input status (0=PNONE, 1=POK).

integer viewi
  View index (>=0).

integer np
  Number of points.

real pxa (n)
  x coordinates of points in the stroke in WC.

real pya (n)
  y coordinates of points in the stroke in WC.

real pza (n)
  z coordinates of points in the stroke in WC.

Errors

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
251  Function Requires Input Device To Be In Request Mode

Related Subroutines

  • Initialize Stroke 3
  • Inquire Number Of Available Logical Input Devices
  • Set Stroke Mode

REQUEST VALUATOR (PHOP,WSOP,*,*)

Purpose

Use Request Valuator to have the graPHIGS API execute a request to the specified valuator device.

A status of NONE means that a break action occurred. Otherwise, the graPHIGS API returns a status of OK together with the logical input value which is the current measure of the valuator device. The value returned is in the range specified by your application through the Initialize Valuator subroutine.
Language Bindings

C

`preq_val (ws_id, val_num, in_status, value)`

Input Parameters

*`Pint ws_id`*
  Workstation identifier.

*`Pint val_num`*
  Valuator device number (>=1).

Output Parameters

*`Pin_status *in_status`*
  Input status (0=PIN_STATUS_NONE, 1=PIN_STATUS_OK).

*`Pfloat *value`*
  Requested valuator value.

FORTRAN

`PRQVL (wkid, vldnr, stat, val)`

Input Parameters

*`integer wkid`*
  Workstation identifier.

*`integer vldnr`*
  Valuator device number (>=1).

Output Parameters

*`integer stat`*
  Input status (0=PNONE, 1=POK).

*`real val`*
  Valuator value.

Errors

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
251  Function Requires Input Device To Be In Request Mode

Related Subroutines

- Initialize Valuator
- Inquire Number Of Available Logical Input Devices
- Set Valuator Mode
SAMPLE CHOICE (PHOP,WSOP,*,*)

Purpose

Use Sample Choice to immediately retrieve the current measure of the specified choice device.

If the measure of the choice device indicates no choice, then the graPHIGS API returns a status of NOCHOICE. Otherwise, the graPHIGS API returns a status of OK together with a choice number which is set according to the current measure of the choice device.

Language Bindings

C

psample_choice (ws_id, choice_num, choice_in_status, choice)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint choice_num
   Choice device number (>=1).

Output Parameters

Pin_status *choice_in_status
   Choice input status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Pint *choice
   Choice number.

FORTRAN

PSMCH (wkid, chdnr, stat, chnr)

Input Parameters

integer wkid
   Workstation identifier.

integer chdnr
   Choice device number (>=1).

Output Parameters

integer stat
   Choice input status (1=POK, 2=PNCHOI).

integer chnr
   Choice number.

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
198 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Function Requires Input Device To Be In Request Mode

Related Subroutines

- Inquire Number Of Available Logical Input Devices
- Set Choice Mode

SAMPLE LOCATOR (PHOP,WSOP,*,*)

Purpose

Use Sample Locator to immediately retrieve the current measure of the specified locator device.

The measure consists of a locator position in World Coordinates (WC) and the index of the view table entry which has a matrix that the graPHIGS API used to convert the location from Device Coordinates (DC) to World Coordinates.

This subroutine returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

Note: This function returns a two-dimensional result. The graPHIGS API discards the z coordinate of the locator position. The x and y values of the locator position are identical to those returned by the Sample Locator 3 subroutine for the same operator action.

Language Bindings

C

psample_loc (ws_id, loc_num, view_ind, loc_pos)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint loc_num
   Locator device number (>=1).

Output Parameters

Pint *view_ind
   View index (>=0).

Ppoint *loc_pos
   Locator position in WC.

FORTRAN

PSMLC (wkid, lcdnr, viewi, lpx, lpy)

Input Parameters

integer wkid
   Workstation identifier.

integer lcdnr
   Locator device number (>=1).
Output Parameters

\begin{itemize}
\item \texttt{integer viewi}
  View index (\texttt{>=0}).
\item \texttt{real lpx}
  \( x \) coordinate of locator position in WC.
\item \texttt{real lpy}
  \( y \) coordinate of locator position in WC.
\end{itemize}

Errors

\begin{itemize}
\item 3  Function Requires State (PHOP,WSOP,*,*)
\item 54 Specified Workstation Is Not Open
\item 61 Specified Workstation Is Not Of Category Input Or Outin
\item 250 Specified Device Not Available On Workstation
\item 252 Function Requires Input Device To Be In Sample Mode
\end{itemize}

Related Subroutines

\begin{itemize}
\item Inquire Number Of Available Logical Input Devices
\item Set Locator Mode
\end{itemize}

\section*{SAMPLE LOCATOR 3 (PHOP,WSOP,*,*)}

\textbf{Purpose}

Use Sample Locator 3 to immediately retrieve the current measure of the specified locator device.

The measure consists of a locator position in World Coordinates (WC) and the index of the view table entry which has a matrix that the graPHIGS API used to convert the location from Device Coordinates (DC) to World Coordinates (WC).

This subroutine returns the locator input from the view with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

\textbf{Language Bindings}

\section*{C}

\texttt{psample_loc3 (ws\_id, loc\_num, view\_ind, loc\_pos)}

\textbf{Input Parameters}

\begin{itemize}
\item \texttt{Pint ws\_id}
  Workstation identifier.
\item \texttt{Pint loc\_num}
  Locator device number (\texttt{>=1}).
\end{itemize}

\textbf{Output Parameters}

\begin{itemize}
\item \texttt{Pint *view\_ind}
  View index (\texttt{>=0}).
\item \texttt{Ppoint3 *loc\_pos}
  Locator position in WC.
\end{itemize}
FORTRAN

PSMLC3 (wkid, lcdnr, viewi, lpx, lpy, lpz)

Input Parameters

integer wkid
   Workstation identifier.

integer lcdnr
   Locator device number (>=1).

Output Parameters

integer viewi
   View index (>=0).

real lpx
   x coordinate of locator position in WC.

real lpy
   y coordinate of locator position in WC.

real lpz
   z coordinate of locator position in WC.

Errors

3   Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
61   Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
252  Function Requires Input Device To Be In Sample Mode

Related Subroutines

• Inquire Number Of Available Logical Input Devices
• Set Locator Mode

SAMPLE PICK (PHOP,WSOP,*,*)

Purpose

Use Sample Pick to immediately retrieve the current measure of the specified pick device.

The graPHIGS API returns the pick path information in the order specified in the Initialize Pick subroutine, that is, TOP FIRST or BOTTOM FIRST. If your application has not called Initialize Pick, then the pick path order defaults to TOP FIRST.

If the measure of the pick device indicates no pick, then the graPHIGS API returns a status of NOPICK Otherwise, the graPHIGS API returns a status of OK together with a pick path which is set according to the current measure of the pick device. Each entry in the pick path consists of a structure identifier, a pick identifier, and an element position.

Language Bindings
C

psample_pick (ws_id, pick_num, depth, pick_in_status, pick)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint pick_num
    Pick device number (>=1).

Pint depth
    Maximum depth of pick path to return.

Output Parameters

Pin_status *pick_in_status
    Pick input status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Ppick_path *pick
    Pick path.

FORTRAN

PSMPK (wkid, pkdnr, ippd, stat, ppd, pp)

Input Parameters

integer wkid
    Workstation identifier.

integer pkdnr
    Pick device number (>=1).

integer ippd
    Maximum depth of pick path to return.

Output Parameters

integer stat
    Pick input status (1=POK, 2=PNPICK).

integer ppd
    Depth of the actual pick path.

integer pp (3,ippd)
    Pick path.

Errors

3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation
252   Function Requires Input Device To Be In Sample Mode

Related Subroutines

• Inquire Number Of Available Logical Input Devices

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
• Set Pick Mode

**SAMPLE STRING (PHOP,WSOP,*,*)**

**Purpose**

Use Sample String to retrieve the current measure of the specified string device.

**Language Bindings**

C

```c
psample_string (ws_id, string_num, string)
```

**Input Parameters**

*Pint ws_id*

Workstation identifier.

*Pint string_num*

String device number (>=1).

**Output Parameters**

*char *string*

Character string. The application must allocate the memory for the character string returned. The Initialize String subroutine or the Initialize String 3 subroutine specifies the maximum size of the returned character size. The Inquire Default String Device Data subroutine or the Inquire Default String Device Data 3 subroutine returns the maximum size of a character string supported by the workstation.

**FORTRAN**

```fortran
PSMST (wkid, stdnr, lostr, str)
```

**Input Parameters**

*integer wkid*

Workstation identifier.

*integer stdnr*

String device number (>=1).

**Output Parameters**

*integer lostr*

Number of characters returned.

*character(*) str*

Character string.

**FORTRAN Subset**

```fortran
PSMST (wkid, stdnr, lostr, str)
```

**Input Parameters**
integer wkid
   Workstation identifier.

integer stdnr
   String device number (>=1).

Output Parameters

integer losstr
   Number of characters returned.

c,character*80 str
   Character string.

Errors

3   Function Requires State (PHOP,WSOP;*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
252 Function Requires Input Device To Be In Sample Mode

Related Subroutines

- Inquire Number Of Available Logical Input Devices
- Set String Mode

SAMPLE STROKE (PHOP,WSOP;*,*)

Purpose

Use Sample Stroke to immediately retrieve the current measure of the specified stroke device.

This measure consists of a sequence of stroke positions (not exceeding the current input buffer size) in World Coordinates (WC), and the index of the view table entry which has a matrix that the graPHIGS API used to convert the stroke locations from Device Coordinates (DC) to World Coordinates.

This subroutine returns the stroke input from the view with the highest input priority which contains all the points. View zero is the highest priority view unless modified by your application.

   Note: This function returns a two-dimensional result. The graPHIGS API discards the z coordinates of the stroke points. The x and y values of the stroke points are identical to those returned by the the Sample Stroke 3 subroutine for the same operator action.

Language Bindings

C

psample_stroke (ws_id, stroke_num, view_ind, stroke)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint stroke_num
   Stroke device number (>=1).
Output Parameters

*view_ind
View index.

*stroke
Stroke point list in WC. The application must allocate the memory for the point list returned. The Initialize Stroke subroutine specifies the maximum size of the returned stroke point list. The Inquire Default Stroke Device Data subroutine returns the maximum size of a stroke point list supported by the workstation.

FORTRAN

PSMSK (wkid, skdnr, n, viewi, np, pxa, pya)

Input Parameters

integer wkid
Workstation identifier.

integer skdnr
Stroke device number (>=1).

integer n
Dimension of arrays for stroke points.

Output Parameters

integer viewi
View index (>=0).

integer np
Number of points.

real pxa (n)
x coordinates of points in the stroke in WC.

real pya (n)
y coordinates of points in the stroke in WC.

Errors

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
252 Function Requires Input Device To Be In Sample Mode

Related Subroutines

• Inquire Number Of Available Logical Input Devices
• Set Stroke Mode

SAMPLE STROKE 3 (PHOP,WSOP,*,*)

Purpose

Use Sample Stroke 3 to immediately retrieve the current measure of the specified stroke device.
This measure consists of a sequence of stroke positions (not exceeding the current input buffer size) in World Coordinates (WC), and the index of the view table entry which has a matrix that the graPHIGS API used to convert the stroke locations from Device Coordinates (DC) to World Coordinates.

This subroutine returns the stroke input from the view with the highest input priority which contains all the points. View zero is the highest priority view unless modified by your application.

**Language Bindings**

**C**

```c
cpsample_stroke3 (ws_id, stroke_num, view_ind, stroke)
```

**Input Parameters**

Pint `ws_id`  
Workstation identifier.

Pint `stroke_num`  
Stroke device number (>=1).

**Output Parameters**

Pint `*view_ind`  
View index.

Ppoint_list3 `*stroke`  
Stroke point list in WC. The application must allocate the memory for the point list returned. The Initialize Stroke 3 subroutine specifies the maximum size of the returned stroke point list. The Inquire Default Stroke Device Data 3 subroutine returns the maximum size of a stroke point list supported by the workstation.

**FORTRAN**

```fortran
PSMSK3 (wkid, skdnr, n, viewi, np, pxa, pya, pza)
```

**Input Parameters**

```
integer wkid  
Workstation identifier.

integer skdnr  
Stroke device number (>=1).

integer n  
Dimension of arrays for stroke points.
```

**Output Parameters**

```
integer viewi  
View index (>=0).

integer np  
Number of points.

real pxa (n)  
$x$ coordinates of points in the stroke in WC.

real pya (n)  
$y$ coordinates of points in the stroke in WC.
```
real pza (n)
z coordinates of points in the stroke in WC.

Errors
3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
252 Function Requires Input Device To Be In Sample Mode

Related Subroutines
• Inquire Number Of Available Logical Input Devices
• Set Stroke Mode

SAMPLE VALUATOR (PHOP,WSOP,*,*)

Purpose
Use Sample Valuator to retrieve the current measure of the specified valuator device.
The returned value is in the range specified for this device through the Initialize Valuator subroutine.

Language Bindings

C

psample_val (ws_id, val_num, value)

Input Parameters

Pint ws_id
  Workstation identifier.

Pint val_num
  Valuator device number (>=1).

Output Parameters

Pfloat *value
  Valuator value.

FORTRAN

PSMVL (wkid, vldnr, val)

Input Parameters

integer wkid
  Workstation identifier.

integer vldnr
  Valuator device number (>=1).

Output Parameters
real val
   Valuator value.

Errors
3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation
252 Function Requires Input Device To Be In Sample Mode

Related Subroutines
• Inquire Number Of Available Logical Input Devices
• Set Valuator Mode

SET CHOICE MODE (PHOP,WSOP,*,*)

Purpose
Use Set Choice Mode to set the operating mode of the specified choice input device.

After the choice mode is set, the graPHIGS API sets the echoing state to _ECHO_ or _NOECHO_. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to _SAMPLE_ or _EVENT_.

Language Bindings

C

_pset_choice_mode_ (ws_id, choice_num, op_mode, echo_switch)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint choice_num
   Choice device number (>=1).

Pop_mode op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

FORTRAN

_PSCHM_ (wkid, chdnr, mode, esw)

Input Parameters
integer wkid
Workstation identifier.

integer chdnr
Choice device number (>=1).

integer mode
Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
Echo switch (0=PNECHO, 1=PECHO).

Errors
3 Function Requires State (PHOP,WSOP,* , *)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Related Subroutines
• Await Event
• Initialize Choice
• Initialize Choice 3
• Inquire Number Of Available Logical Input Devices
• Request Choice
• Sample Choice

SET LOCATOR MODE (PHOP,WSOP,* , *)

Purpose
Use Set Locator Mode to set the operating mode of the specified locator device.

After the Locator Mode is set, the graPHIGS API sets the echoing state to ECHO or NOECHO. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to SAMPLE or EVENT.

Language Bindings

C

pset_loc_mode (ws_id, loc_num, op_mode, echo_switch)

Input Parameters
Pint ws_id
Workstation identifier.

Pint loc_num
Locator device number (>=1).

Pop_mode op_mode
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).
Pecho_switch echo_switch
    Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

FORTRAN

PSLCM (wkid, lcdnr, mode, esw)

Input Parameters

integer wkid
    Workstation identifier.

integer lcdnr
    Locator device number (>=1).

integer mode
    Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
    Echo switch (0=PNECHO, 1=PECHO).

Errors

3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation

Related Subroutines

• Await Event
• Initialize Locator
• Initialize Locator 3
• Inquire Number Of Available Logical Input Devices
• Request Locator
• Request Locator 3
• Sample Locator
• Sample Locator 3

SET PICK FILTER (PHOP,WSOP,*,*)

Purpose

Use Set Pick Filter to set the pick inclusion and exclusion filters for the specified pick device.

The filters consist of class names which indicate which class names to include and which to exclude from pickability (detectability).

Language Bindings

C

pset_pick_filter (ws_id, pick_num, filter)

Input Parameters

210 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Pint ws_id
   Workstation identifier.

Pint pick_num
   Pick device number (>=1).

const Pfilter *filter
   Pick filter.

FORTRAN

PSPKFT (wkid, pkdnr, isn, is, esn, es)

Input Parameters

integer wkid
   Workstation identifier.

integer pkdnr
   Pick device number (>=1).

integer isn
   Number of names in the inclusion set (>=0).

integer is (isn)
   Inclusion set.

integer esn
   Number of names in the exclusion set (>=0).

integer es (esn)
   Exclusion set.

Errors

3     Function Requires State (PHOP,WSOP,* , * )
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation

Related Subroutines

• Add Names To Set
• Inquire Number Of Available Logical Input Devices
• Remove Names From Set

SET PICK MODE (PHOP,WSOP,* , * )

Purpose

Use Set Pick Mode to set the operating mode of the specified pick device.

After the pick mode is set, the graPHIGS API sets the echoing state to ECHO or NOECHO. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to SAMPLE or EVENT.
Language Bindings

C

cpset_pick_mode(ws_id, pick_num, op_mode, echo_switch)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint pick_num
    Pick device number (>=1).

Pop_mode op_mode
    Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch echo_switch
    Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

FORTRAN

PSPKM(wkid, pkdnr, mode, esw)

Input Parameters

integer wkid
    Workstation identifier.

integer pkdnr
    Pick device number (>=1).

integer mode
    Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
    Echo switch (0=PNECHO, 1=PECHO).

Errors

3    Function Requires State (PHOP,WSOP,*,*).
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250    Specified Device Not Available On Workstation

Related Subroutines

- Await Event
- Initialize Pick
- Initialize Pick 3
- Inquire Number Of Available Logical Input Devices
- Request Pick
- Sample Pick
SET STRING MODE (PHOP, WSOP, *, *)

Purpose

Use Set String Mode to set the operating mode of the specified string device.

After the string mode is set, the graPHIGS API sets the echoing state to ECHO or NOECHO. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to SAMPLE or EVENT.

Language Bindings

C

pset_string_mode (ws_id, string_num, op_mode, echo_switch)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint string_num
   String device number (>=1).

Pop_mode op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

FORTRAN

PSSTM (wkid, stdnr, mode, esw)

Input Parameters

integer wkid
   Workstation identifier.

integer stdnr
   String device number (>=1).

integer mode
   Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

Errors

3    Function Requires State (PHOP, WSOP, *, *)
54   Specified Workstation Is Not Open
61   Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation
Related Subroutines
- Await Event
- Initialize String
- Initialize String 3
- Inquire Number Of Available Logical Input Devices
- Request String
- Sample String

SET STROKE MODE (PHOP, WSOP, *, *)

Purpose

Use Set Stroke Mode to set the operating mode of the specified stroke device.

After the stroke mode is set, the graPHIGS API sets the echoing state to ECHO or NOECHO. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may either begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to SAMPLE or EVENT.

Language Bindings

C

pset_stroke_mode (ws_id, stroke_num, op_mode, echo_switch)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint stroke_num
   Stroke device number (>=1).

Pop_mode op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch echo_switch
   Echo switch (0=P SWITCH_NO_ECHO, 1=P SWITCH_ECHO).

FORTRAN

PSSKM (wkid, skdnr, mode, esw)

Input Parameters

integer wkid
   Workstation identifier.

integer skdnr
   Stroke device number (>=1).

integer mode
   Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).
integer esw
    Echo switch (0=PNECHO, 1=PECHO).

Errors
3       Function Requires State (PHOP,WSOP,*,*)
54      Specified Workstation Is Not Open
61      Specified Workstation Is Not Of Category Input Or Outin
250     Specified Device Not Available On Workstation

Related Subroutines
•   Await Event
•   Initialize Stroke
•   Initialize Stroke 3
•   Inquire Number Of Available Logical Input Devices
•   Request Stroke
•   Request Stroke 3
•   Sample Stroke
•   Sample Stroke 3

SET VALUATOR MODE (PHOP,WSOP,*,*)

Purpose

Use Set Valuator Mode to set the operating mode of the specified valuator device.

After the valuator mode is set, the graPHIGS API sets the echoing state to ECHO or NOECHO. Depending on the specified operating mode—Request, Sample, or Event—an interaction with the given device may either begin or end.

Note: The graPHIGS API resets the input device with the initialization values when your application calls the Description subroutine with the operating mode parameter set to SAMPLE or EVENT.

Language Bindings

C

pset_valuator_mode (ws_id, val_num, op_mode, echo_switch)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint val_num
    Valuator device number (>=1).

Pop_mode op_mode
    Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch echo_switch
    Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).
FORTRAN

PSVLM (wkid, vldnr, mode, esw)

Input Parameters

integer wkid
   Workstation identifier.

integer vldnr
   Valuator device number (>=1).

integer mode
   Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

Errors

3    Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
61   Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation

Related Subroutines

- Await Event
- Initialize Valuator
- Initialize Valuator 3
- Inquire Number Of Available Logical Input Devices
- Request Valuator
- Sample Valuator
Chapter 12. Utility Subroutines

The subroutines in this category provide convenient mechanisms for modifying data or performing calculations.

Most subroutines perform transformations on matrixes. In addition, the Pack Data Record and Unpack Data Record utilities provide a convenient mechanism for the handling of data records used by input device initialization subroutines. These two utilities are defined only for the FORTRAN binding.

The Create Store and Delete Store utilities are defined only for the C binding. The graPHIGS API uses an object of type Store to facilitate the task of using a C binding subroutine which returns complex data.

BUILD TRANSFORMATION MATRIX (PHOP,*,*,*)

Purpose

Use Build Transformation Matrix to calculate a specified two-dimensional homogenous transformation matrix. The order of transformation is: scale, rotate (both relative to the specified fixed point), and shift.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pbuild_trn_matrix (point, shift_vec, angle, scale_vec, err_ind, result_tran)

Input Parameters

const Ppoint *point
Fixed point.

const Pvec *shift_vec
Shift vector.

Pfloat angle
Rotation angle in radians (positive if counterclockwise).

const Pvec *scale_vec
Scale vector.

Output Parameters

Pint *err_ind
Error indicator.

Pmatrix result_tran
Transformation matrix. (See Chapter 15. ”ISO PHIGS Transformations” for a description of the storage of the transformation matrix).
FORTRAN

PBLTM (x0, y0, dx, dy, phi, fx, fy, errind, xfrmt)

Input Parameters

real x0
    x coordinate of the fixed point.
real y0
    y coordinate of the fixed point.
real dx
    x offset of the shift vector.
real dy
    y offset of the shift vector.
real phi
    Rotation angle in radians (positive if counterclockwise).
real fx
    x-axis scale factor.
real fy
    y-axis scale factor.

Output Parameters

integer errind
    Error indicator.
real xfrmt(3,3)

Errors

None

Related Subroutines

• None

BUILD TRANSFORMATION MATRIX 3 (PHOP,*,*,*)

Purpose

Use Build Transformation Matrix 3 to calculate a specified three-dimensional homogenous transformation matrix. The order of transformation which is all relative to the specified fixed point is: scale, rotate x, rotate y, rotate z, and shift.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings
C

`pbuild_tran_matrix3 (point, shift_vec, x_angle, y_angle, z_angle, scale_vec, err_ind, result_tran)`

**Input Parameters**

- `const Ppoint3 *point`
  - Fixed point.

- `const Pvec3 *shift_vec`
  - Shift vector.

- `Pfloat x_angle`
  - Rotation angle x in radians (positive if counterclockwise).

- `Pfloat y_angle`
  - Rotation angle y in radians (positive if counterclockwise).

- `Pfloat z_angle`
  - Rotation angle z in radians (positive if counterclockwise).

- `const Pvec3 *scale_vec`
  - Scale vector.

**Output Parameters**

- `Pint *err_ind`
  - Error indicator.

- `Pmatrix3 result_tran`

**FORTRAN**

`PBLTM3 (x0, y0, z0, dx, dy, dz, phix, phiy, phiz, fx, fy, fz, errind, xfrmt)`

**Input Parameters**

- `real x0`
  - x coordinate of the fixed point.

- `real y0`
  - y coordinate of the fixed point.

- `real z0`
  - z coordinate of the fixed point.

- `real dx`
  - x offset of the shift vector.

- `real dy`
  - y offset of the shift vector.

- `real dz`
  - z offset of the shift vector.

- `real phix`
  - Rotation angle x in radians (positive if counterclockwise).

- `real phiy`
  - Rotation angle y in radians (positive if counterclockwise).
real phiz
    Rotation angle z in radians (positive if counterclockwise).

real fx  x-axis scale factor.
real fy  y-axis scale factor.
real fz  z-axis scale factor.

Output Parameters

integer errind
    Error indicator.

real xfrmt(4,4)

Errors

None

Related Subroutines

• None

---

COMPOSE MATRIX (PHOP,*,*,*)

Purpose

Use Compose Matrix to perform a 3x3 matrix multiplication and return the result.

The graPHIGS API computes: Transformation Matrix A x Transformation Matrix B and returns the result as the composed transformation matrix.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the composed transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pcompose_matrix (tran_a, tran_b, err_ind, result_tran)

Input Parameters

Pmatrix tran_a

Pmatrix tran_b

Output Parameters
FORTRAN

**PCOM (xfrmta, xfrmtb, errind, xfrmtc)**

**Input Parameters**

- **real xfrmta(3,3)**

- **real xfrmtb(3,3)**

**Output Parameters**

- **integer errind**
  - Error indicator.

- **real xfrmtc(3,3)**

**Errors**

None

**Related Subroutines**

- None

---

**COMPOSE MATRIX 3 (PHOP,*,*,*)**

**Purpose**

Use Compose Matrix 3 to perform a 4x4 matrix multiplication and return the results.

The graPHIGS API computes: Transformation Matrix A x Transformation Matrix B and returns the result as the composed transformation matrix.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the composed transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

**Language Bindings**

**C**

pcompose_matrix3 (tran_a, tran_b, err_ind, result_tran)
Input Parameters

$P_{\text{matrix3}}\ tran\_a$

$P_{\text{matrix3}}\ tran\_b$

Output Parameters

$P_{\text{int}}\ *\ err\_ind$
Error indicator.

$P_{\text{matrix3}}\ result\_tran$

FORTRAN

PCOM3 ($xfrmta, xfrmtb, errind, xfrmto$)

Input Parameters

real $xfrmta(4,4)$

real $xfrmtb(4,4)$

Output Parameters

integer $errind$
Error indicator.

real $xfrmto(4,4)$

Errors

None

Related Subroutines

• None

COMPOSE TRANSFORMATION MATRIX (PHOP,*,*,*)

Purpose

Use Compose Transformation Matrix to compute a two-dimensional transformation matrix which is the composition of the specified matrix with the matrix defined by the fixed point, shift, rotate and scale parameters. The order of transformation is: scale, rotate (both relative to the specified fixed point), and shift.
If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the composed transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the composed transformation matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pcompose_tran_matrix (tran, point, shift_vec, angle, scale_vec, err_ind, result_tran)

Input Parameters

Pmatrix tran

const Ppoint *point
  Fixed point.

const Pvec *shift_vec
  Shift vector.

Pfloat angle
  Rotation angle in radians (positive if counterclockwise).

const Pvec *scale_vec
  Scale vector.

Output Parameters

Pint *err_ind
  Error indicator.

Pmatrix result_tran

FORTRAN

PCOTM (xfrmti, x0, y0, dx, dy, phi, fx, fy, errind, xfrmt0)

Input Parameters

real xfrmti(3,3)

real x0
  x coordinate of the fixed point.

real y0
  y coordinate of the fixed point.

real phi
  Rotation angle in radians (positive if counterclockwise).

real fx
  x-axis scale factor.

real fy
  y-axis scale factor.
Output Parameters

integer errind
    Error indicator.

real xfrmto(3,3)

Errors

None

Related Subroutines

• None

**COMPOSE TRANSFORMATION MATRIX 3 (PHOP,*,*,*)**

Purpose

Use Compose Transformation Matrix 3 to compute a three-dimensional transformation matrix which is the composition of the specified matrix with the matrix defined by the fixed point, shift, rotate and scale parameters. The order of transformation (all relative to the specified fixed point) is: scale, rotate x, rotate y, rotate z, and shift.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero, and returns the composed transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the composed transformation matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2   Function Requires State (PHOP,*,*,*)

Language Bindings

C

pcompose_tran_matrix3 (tran, point, shift_vec, x_angle, y_angle, z_angle, scale_vec, err_ind, result_tran)

Input Parameters

Pmatrix3 tran

const Ppoint3 *point
    Fixed point.

const Pvec3 *shift_vec
    Shift vector.

Pfloat x_angle
    Rotation angle x in radians (positive if counterclockwise).

Pfloat y_angle
    Rotation angle y in radians (positive if counterclockwise).

Pfloat z_angle
    Rotation angle z in radians (positive if counterclockwise).
const Pvec3 *scale_vec
  Scale vector.

**Output Parameters**

*Pint* *err_ind
  Error indicator.

*Pmatrix3 result_tran

**FORTRAN**

PCOTM3 (xfrmti, x0, y0, z0, dx, dy, dz, phix, phiy, phiz, fx, fy, fz, errind, xfrmt0)

**Input Parameters**

*real xfrmti(4,4)

*real x0
  x coordinate of the fixed point.

*real y0
  y coordinate of the fixed point.

*real z0
  z coordinate of the fixed point.

*real phix
  Rotation angle x in radians (positive if counterclockwise).

*real phiy
  Rotation angle y in radians (positive if counterclockwise).

*real phiz
  Rotation angle z in radians (positive if counterclockwise).

*real fx
  x-axis scale factor.

*real fy
  y-axis scale factor.

*real fz
  z-axis scale factor.

**Output Parameters**

*integer errind
  Error indicator.

*real xfrmt0(4,4)

**Errors**

None

**Related Subroutines**

- None
CREATE STORE (PHOP,*,*,*)

Purpose

Use Create Store to create a new Store resource. The graPHIGS API uses the Store resource to manage the memory needed by the subroutines that return complex data. Use of the Store resource provides two levels of memory management: low level and high level. The graPHIGS API manages the memory at a low level because it uses, re-uses, allocates, and deallocates memory from the system in order to return data to the application. However, the application manages the memory at a high level because it creates and deletes the Stores. An application may create multiple Stores.

The application can pass the newly created Store resource as a parameter to a subroutine returning complex data. Another parameter to a subroutine returning complex data is a pointer to a pointer to a structure which defines the additional memory referenced by fields within the structure. The application accesses the returned data through its pointer to the structure. It does not use the Store resource to access the data.

A Store continues to hold the information from the function until the Delete Store subroutine deletes the Store or until the application uses the Store as a parameter to a subsequent subroutine which returns complex data. Then the graPHIGS API replaces the old information with the newly requested data. A Store resource only contains the results of the last subroutine.

If the graPHIGS API can create the Store resource, then the graPHIGS API sets the error indicator to zero and returns the Store handle. If the graPHIGS API cannot create the Store resource, then the value of the Store handle is unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2     Function Requires State (PHOP,*,*,*)
2203  Error While Allocating Store

Language Binding

C

pcreate_store (err_ind, store)

Output Parameters

Pint *err_ind
    Error indicator.

Pstore *store
    New Store.

Errors

None

Related Subroutines

• Delete Store

DELETE STORE (PHOP,*,*,*)

Purpose

Use Delete Store to delete a Store and all internal resources associated with it.
If the graPHIGS API can delete the Store resource, then the graPHIGS API sets the error indicator to zero and sets the Store parameter to NULL. If the graPHIGS API cannot delete the Store resource, then the Store parameter is unaffected and the graPHIGS API sets the error indicator to the following error:

2  Function Requires State (PHOP,*,*,*)

**Language Binding**

C

**pdel_store** (*err_ind, store*)

**Output Parameters**

Pint *err_ind
  Error indicator.
Pstore *store
  Store to be deleted.

**Errors**

None

**Related Subroutines**

- Create Store

---

**EVALUATE VIEW MAPPING MATRIX (PHOP,*,*,*)**

**Purpose**

Use Evaluate View Mapping Matrix to create a two-dimensional view mapping matrix. Your application can use the matrix as input to the Set View Representation subroutine.

When calculating the view mapping matrix, the graPHIGS API:

- sets the z extents for the viewport to the z extents of the Normalized Projection Coordinates (NPC) range.
- sets the projection type to *PARALLEL*.
- places the projection reference point on a line perpendicular to the center of the specified window.
- sets the z value of the projection reference point to one-half of the maximum of the Umax-Umin and Vmax-Vmin.
- sets the view plane distance to zero.
- sets the far clipping plane to the negative of one-half of the maximum of the Umax-Umin and Vmax-Vmin.
- sets the near clipping plane to one-half of the maximum of the Umax-Umin and Vmax-Vmin.

If the graPHIGS API can compute the view mapping matrix, then the graPHIGS API sets the error indicator to zero and returns the view mapping matrix. If the graPHIGS API cannot compute the view mapping matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)

151  Invalid Window: Minimum Value >= To Corresponding Maximum Value

152  Invalid Viewport: XMIN >= XMAX, YMIN >= YMAX OR ZMIN > ZMAX
Projection Viewport Limits Are Not Within NPC Range

Language Bindings

C

`peval_view_map_matrix (mapping, err_ind, result_tran)`

**Input Parameters**

```c
const Pview_map *mapping
```

View mapping (window limits in VC and projection viewport limits in NPC).

**Output Parameters**

```c
Pint *err_ind
```

Error indicator.

```c
Pmatrix result_tran
```


**FORTRAN**

`PEVMM (vwwnlm, pjvplm, errind, vwmpmt)`

**Input Parameters**

``` Fortran
real vwwnlm(4)
```

Window limits in VC (`UMIN, UMAX, VMIN, VMAX`).

``` Fortran
real pjvplm(4)
```

Projection viewport limits in NPC (`XMIN, XMAX, YMIN, YMAX`).

**Output Parameters**

``` Fortran
integer errind
```

Error indicator.

``` Fortran
real vwmpmt(3,3)
```


**Errors**

None

**Related Subroutines**

- None

---

**EVALUATE VIEW MAPPING MATRIX 3 (PHOP, ** , ** )**

**Purpose**

Use Evaluate View Mapping Matrix 3 to create a view mapping matrix. Your application can use the matrix as input to the Set View Representation 3 subroutine.
If the graPHIGS API can compute the view mapping matrix, then the graPHIGS API sets the error indicator to zero and returns the view mapping matrix. If the graPHIGS API cannot compute the view mapping matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)

151 Invalid Window: Minimum Value >= To Corresponding Maximum Value

152 Invalid Viewport: XMIN >= XMAX, YMIN >= YMAX OR ZMIN > ZMAX

158 Front Plane Distance = Back Plane Distance When Z-Extent Non-Zero

162 Projection Reference Point Between Front And Back Planes

163 Projection Reference Point Cannot Be Positioned On View Plane

164 Back Plane Is In Front Of The Front Plane

155 Projection Viewport Limits Are Not Within NPC Range

Language Bindings

C

peval_view_map_matrix3 (mapping, err_ind, result_tran)

Input Parameters

const Pview_map3 *mapping
  View mapping (window limits in VC, projection viewport limits in NPC, projection type and projection reference point in VC, and view, front and back plane distances in VC).

Output Parameters

Pint *err_ind
  Error indicator.

Pmatrix3 result_tran

FORTRAN

PEVMM3 (vwwnlm, pjvplm, pjtype, pjrx, pjry, pjrz, vpld, bpld, fpld, errind, vwmpmt)

Input Parameters

real vwwnlm(4)
  Window limits in VC (UMIN, UMAX, VMIN, VMAX).

real pjvplm(6)
  Projection viewport limits in NPC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer pjtype
  Projection type (PPARL, PPERS).

real pjrx
  x-axis projection reference point in VC.

real pjry
  y-axis projection reference point in VC.
real pjrz
    z-axis projection reference point in VC.

real vpld
    View plane distance in VC.

real bpld
    Back plane distance in VC.

real fpld
    Front plane distance in VC.

Output Parameters

integer errind
    Error indicator.

real vwmpmt(4,4)

Errors

None

Related Subroutines

• None

EVALUATE VIEW ORIENTATION MATRIX (PHOP,*,*,*)

Purpose

Use Evaluate View Orientation Matrix to calculate a two-dimensional viewing matrix based on the specified orientation.

The matrix returned performs a change from the World Coordinate (WC) system to a Viewing Coordinate (VC) system in which the origin is the view reference point with a z coordinate of zero, the n-axis is the view plane normal assumed to be [0,0,1], and the v-axis lies in the half plane designated by the view up vector with a z coordinate of zero.

If the graPHIGS API can compute the view orientation matrix, then the graPHIGS API sets the error indicator to zero and returns the view orientation matrix. If the graPHIGS API cannot compute the view orientation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2      Function Requires State (PHOP,*,*,*)
160    View UP Vector Has Length Zero

Language Bindings

C

peval_view_ori_matrix (view_ref_point, view_up_vec, err_ind, result_tran)

Input Parameters

cnst Ppoint *view_ref_point
    View reference point in WC.
const Pvec *view_up_vec
  View up vector in WC.

Output Parameters
Pint *err_ind
  Error indicator.
Pmatrix result_tran

FORTRAN
PEVOM (vwrx, vwry, vupx, vupy, errind, vwormt)

Input Parameters
real vwrx
  x coordinate of the view reference point in WC.
real vwry
  y coordinate of the view reference point in WC.
real vupx
  x-axis directional component of the view up vector in WC.
real vupy
  y-axis directional component of the view up vector in WC.

Output Parameters
integer errind
  Error indicator.
real vwormt(3,3)

Errors
None

Related Subroutines
• None

EVALUATE VIEW ORIENTATION MATRIX 3 (PHOP,*,*,*)

Purpose
Use Evaluate View Orientation Matrix 3 to calculate a three-dimensional viewing matrix based on the specified orientation.

The matrix returned performs a change from the World Coordinate (WC) system to a Viewing Coordinate (VC) system in which the origin is the view reference point, the n-axis is the view plane normal, and the v-axis lies in the half plane designated by the view up vector.
If the graPHIGS API can compute the view orientation matrix, then the graPHIGS API sets the error indicator to zero and returns the view orientation matrix. If the graPHIGS API cannot compute the view orientation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
159 View Plane Normal Vector Has Length Zero
160 View UP Vector Has Length Zero
161 View UP AND View Plane Normal Vectors Are Parallel

Language Bindings

C

peval_view_ori_matrix3 (view_ref_point, view_norm_vec, view_up_vec, err_ind, result_tran)

Input Parameters

const Ppoint3 *view_ref_point
    View reference point in WC.
const Pvec3 *view_norm_vec
    View plane normal vector in WC.
const Pvec3 *view_up_vec
    View up vector in WC.

Output Parameters

Pint *err_ind
    Error indicator.
Pmatrix3 result_tran
    View orientation matrix. (See [Chapter 15. “ISO PHIGS Transformations”](#) for a description of the storage of the transformation matrix).

FORTRAN

PEVOM3 (vwx, vwy, vwz, vpx, vpy, vz, vupx, vupy, vupz, errind, vwrm)

Input Parameters

real vx
    x coordinate of the view reference point in WC.
real vy
    y coordinate of the view reference point in WC.
real vz
    z coordinate of the view reference point in WC.
real vpx
    x-axis directional component of the view plane normal in WC.
real vpy
    y-axis directional component of the view plane normal in WC.
real vpz
    z-axis directional component of the view plane normal in WC.
real vupx
   x-axis directional component of the view up vector in WC.

real vupy
   y-axis directional component of the view up vector in WC.

real vupz
   z-axis directional component of the view up vector in WC.

Output Parameters

integer errind
   Error indicator.

real vwormt(4,4)

Errors

None

Related Subroutines

• None

PACK DATA RECORD (PHOP,*,*,*

Purpose

Use Pack Data Record to construct a data record for passing to input device initialization routines. The data record constructed by Pack Data Record consists of a header identifying the number of integers, reals, and character strings in the data record, followed by the actual data. Your application can pass the ldr and datrec output parameters to the desired input device initialization subroutine.

Pack Data Record accepts as input a list of integers, a list of reals, and a list of character strings. The length of each character string is specified in a separate array of lengths (lstr).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2    Function Requires State (PHOP,*,*,*
2001  Output Parameter Size Insufficient
2003  Invalid Data Record

Note:  This utility is defined only for the FORTRAN binding.

Language Binding

FORTRAN

PPREC (il, ia, rl, ra, sl, lstr, str, mldr, errind, ldr, datrec)

Input Parameters

integer il
   Number of integer entries (>=0).
**Input Parameters**

- `integer il`  
  Number of integer entries (>=0).

- `integer ia (il)`  
  Array containing integer entries.

- `integer rl`  
  Number of real entries (>=0).

- `real ra (rl)`  
  Array containing real entries.

- `integer sl`  
  Number of character string entries (>=0).

- `integer lstr(sl)`  
  Lengths of each character string entry (>=0).

- `character*80 str(*)`  
  Character string entries.

- `integer mldr`  
  Dimension of the data record array.

**Output Parameters**

- `integer errind`  
  Error indicator.

- `integer ldr`  
  Number of array elements used in `datrec`.

- `character*80 datrec(mldr)`  
  Data record.

**FORTRAN Subset**

PPREC (il, ia, rl, ra, sl, lstr, str, mldr, errind, ldr, datrec)
integer errind
   Error indicator.

integer ldr
   Number of array elements used in datrec.

character*80 datrec(ldr)
   Data record.

Errors
None

Related Subroutines
- Initialize Choice
- Initialize Choice 3
- Initialize Locator
- Initialize Locator 3
- Initialize Pick
- Initialize Pick 3
- Initialize String
- Initialize String 3
- Initialize Stroke
- Initialize Stroke 3
- Initialize Valuator
- Initialize Valuator 3
- Unpack Data Record

**ROTATE (PHOP,*,*,*)**

Purpose

Use Rotate to calculate a two-dimensional transformation matrix to perform the specified 2D-axis rotation.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2   Function Requires State (PHOP,*,*,*)

Language Bindings

C

protate (angle, err_ind, result_tran)

Input Parameters

Pfloat angle
   Rotational angle in radians (positive if counterclockwise).

Output Parameters
Pint *err_ind
    Error indicator.

Pmatrix result_tran

FORTRAN

PRO (rotang, errind, xfrmt)

Input Parameters
real rotang
    Rotational angle in radians (positive if counterclockwise).

Output Parameters
integer errind
    Error indicator.

real xfrmt(3,3)

Errors
None

Related Subroutines
- None

ROTATE X (PHOP,*,*,*)

Purpose
Use Rotate X to calculate a three-dimensional transformation matrix to rotate around the x-axis using a given angle of rotation.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:
2          Function Requires State (PHOP,*,*,*)

Language Bindings

C

rotate_x (angle, err_ind, result_tran)

Input Parameters
Pfloat angle
    Rotational angle in radians (positive if counterclockwise).
Output Parameters

Output

Parameters

\[ P\text{int }*\text{err\_ind} \]
Error indicator.

\[ P\text{matrix3 result\_tran} \]

FORTRAN

PROX (\text{rotang}, \text{errind}, \text{xfmt})

Input Parameters

Input Parameters

\[ \text{real rotang} \]
Rotational angle in radians (positive if counterclockwise).

Output Parameters

Output Parameters

\[ \text{integer errind} \]
Error indicator.

\[ \text{real xfmt(4,4)} \]

Errors

None

Related Subroutines

• None

ROTATE Y (PHOP,*,*,*)

Purpose

Use Rotate Y to calculate a three-dimensional transformation matrix to rotate around the y-axis using a given angle of rotation.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2  Function Requires State (PHOP,*,*,*)

Language Bindings

C

\[ \text{protate\_y (angle, err\_ind, result\_tran)} \]

Input Parameters

Input Parameters

\[ \text{float angle} \]
Rotational angle in radians (positive if counterclockwise).
Output Parameters

\[
\begin{align*}
\text{Pint} & \quad \text{*err\_ind} \\
\text{Error indicator.} \\
\text{Pmatrix3} & \quad \text{result\_tran} \\
\text{Transformation matrix. (See Chapter 15. “ISO PHIGS Transformations“ for a description of the storage of the transformation matrix).}
\end{align*}
\]

FORTRAN

\[\text{PROY (rotang, errind, xfmr)}\]

Input Parameters

\[
\begin{align*}
\text{real rotang} \\
\text{Rotational angle in radians (positive if counterclockwise).}
\end{align*}
\]

Output Parameters

\[
\begin{align*}
\text{integer errind} \\
\text{Error indicator.} \\
\text{real xfmr\(4,4\)} \\
\text{Transformation matrix. (See Chapter 15. “ISO PHIGS Transformations“ for a description of the storage of the transformation matrix).}
\end{align*}
\]

Errors

None

Related Subroutines

- None

\[\text{ROTATE Z (PHOP,*,*,*)}\]

Purpose

Use Rotate Z to calculate a three-dimensional transformation matrix to rotate around the z-axis using a given angle of rotation.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

\[\text{2 Function Requires State (PHOP,*,*,*)}\]

Language Bindings

C

\[\text{protate\_z (angle, err\_ind, result\_tran)}\]

Input Parameters

\[
\begin{align*}
\text{Pfloat angle} \\
\text{Rotational angle in radians (positive if counterclockwise).}
\end{align*}
\]
Output Parameters

Pint *err_ind
  Error indicator.

Pmatrix3 result_tran

FORTRAN

PROZ (rotang, errind, xfmt)

Input Parameters

real rotang
  Rotational angle in radians (positive if counterclockwise).

Output Parameters

integer errind
  Error indicator.

real xfmt(4,4)

Errors

None

Related Subroutines

• None

SCALE (PHOP, *, *, *)

Purpose

Use Scale to calculate a two-dimensional transformation matrix to perform the specified 2D-axis scaling.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP, *, *, *)

Language Bindings

C

pscale (scale_vec, err_ind, result_tran)

Input Parameters

const Pvec *scale_vec
  Scale factor vector.
Output Parameters

*err_ind
   Error indicator.

Pmatrix result_trn

FORTRAN

PSC (fx, fy, errind, xfrmt)

Input Parameters

real fx
   x-axis scale factor.
real fy
   y-axis scale factor.

Output Parameters

integer errind
   Error indicator.
real xfrmt(3,3)

Errors

None

Related Subroutines

- None

SCALE 3 (PHOP,*,*,*)

Purpose

Use Scale 3 to calculate a three-dimensional transformation matrix to perform the specified 3D-axis scaling.

If the graPHIGS API can compute the transformation matrix, then the graPHIGS API sets the error indicator to zero and returns the transformation matrix. If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2    Function Requires State (PHOP,*,*,*)

Language Bindings

C

cscale3 (scale_vec, err_ind, result_trn)

Input Parameters
const Pvec3 *scale_vec
    Scale factor vector.

**Output Parameters**

Pint *err_ind
    Error indicator.

Pmatrix3 result_tran

**FORTRAN**

PSC3 (fx, fy, fz, errind, xfrmt)

**Input Parameters**

real fx
    x-axis scale factor.

real fy
    y-axis scale factor.

real fz
    z-axis scale factor.

**Output Parameters**

integer errind
    Error indicator.

real xfrmt(4,4)

**Errors**

None

**Related Subroutines**

• None

---

**TRANSFORM POINT (PHOP,*,*,*)**

**Purpose**

Use Transform Point to transform a point using a specified transformation matrix.

The graPHIGS API returns the result of multiplying the given point by the transformation. If the graPHIGS API cannot return the transformed point, then the values of the point are unpredictable and the graPHIGS API sets the error indicator to the following error:

2    Function Requires State (PHOP,*,*,*)

**Language Bindings**
C

ptran_point (point, tran, err_ind, result)

Input Parameters

const Ppoint *point
Point.

Pmatrix tran

Output Parameters

Pint *err_ind
Error indicator.

Ppoint *result
Transformed point.

FORTRAN

PTP (xi, yi, xfrmt, errind, xo, yo)

Input Parameters

real xi
x coordinate of the point.

real yi
y coordinate of the point.

real xfrmt(3,3)

Output Parameters

integer errind
Error indicator.

real xo
x coordinate of the transformed point.

real yo
y coordinate of the transformed point.

Errors

None

Related Subroutines

• None
TRANSFORM POINT 3 (PHOP,*,*,*)

Purpose

Use Transform Point 3 to transform a point using a specified transformation matrix.

The graPHIGS API returns the result of multiplying the given point by the transformation. If the graPHIGS API cannot return the transformed point, then the values of the point are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

ptran_point3 (point, tran, err_ind, result)

Input Parameters

const Ppoint3 *point
   Point.

Pmatrix3 tran

Output Parameters

Pint *err_ind
   Error indicator.

Ppoint3 *result
   Transformed point.

FORTRAN

PTP3 (xi, yi, zi, xfrmt, errind, xo, yo, zo)

Input Parameters

real xi
   x coordinate of the point.

real yi
   y coordinate of the point.

real zi
   z coordinate of the point.

real xfrmt(4,4)

Output Parameters

integer errind
   Error indicator.
**real xo**

$x$ coordinate of the transformed point.

**real yo**

$y$ coordinate of the transformed point.

**real zo**

$z$ coordinate of the transformed point.

**Errors**

None

**Related Subroutines**

- None

---

**TRANSLATE (PHOP,*,*,*)**

**Purpose**

Use Translate to calculate a two-dimensional transformation matrix to perform the specified 2D-axis translation.

If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

**Language Bindings**

**C**

```c
ptranslate (trans_vec, err_ind, result_tran)
```

**Input Parameters**

*const Pvec* `trans_vec`

Translation vector.

**Output Parameters**

*Pint* `err_ind`

Error indicator.

*Pmatrix* `result_tran`


**FORTRAN**

```fortran
PTR (dx, dy, errind, xfrmt)
```

**Input Parameters**

*real dx*

$x$-axis translation vector.
real dy
    y-axis translation vector.

Output Parameters

integer errind
    Error indicator.

real xfrmt(3,3)

Errors

None

Related Subroutines

• None

TRANSLATE 3 (PHOP,*,*,*)

Purpose

Use Translate 3 to calculate a three-dimensional transformation matrix to perform the specified 3D-axis translation.

If the graPHIGS API cannot compute the transformation matrix, then the values of the matrix are unpredictable and the graPHIGS API sets the error indicator to the following error:

2   Function Requires State (PHOP,*,*,*)

Language Bindings

C

ctranslate3 (trans_vec, err_ind, result_tran)

Input Parameters

c const Pvec3 *trans_vec
    Translation vector.

Output Parameters

Pint *err_ind
    Error indicator.

Pmatrix3 result_tran

FORTRAN

PTR3 (dx, dy, dz, errind, xfrmt)

Input Parameters

For C and FORTRAN input parameters, please refer to the input parameters section in the referenced chapter for detailed descriptions.
real dx
    x-axis translation vector.
real dy
    y-axis translation vector.
real dz
    z-axis translation vector.

Output Parameters

integer errind
    Error indicator.

real xfrmt(4,4)

Errors

None

Related Subroutines

• None

UNPACK DATA RECORD (PHOP,*,*,*)

Purpose

Use Unpack Data Record to unpack a data record returned by FORTRAN binding input device inquiries. The data record unpacked by Unpack Data Record consists of a header identifying the number of integers, reals, and character strings in the data record, followed by the actual data. The graPHIGS API can interpret the output parameters by using the data record information supplied with the FORTRAN input device initialization subroutines.

For input, Unpack Data Record accepts a data record, array addresses for integer, real and character arrays, and size specifications for those arrays.

Unpack Data Record and Pack Data Record are inverse functions.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
2001 Output Parameter Size Insufficient
2003 Invalid Data Record
2004 Input Parameter Size Out Of Range

Note: This utility is defined only for the FORTRAN binding.

Language Bindings

FORTRAN

PUREC (ldr, datrec, iil, irl, isl, errind, il, ia, rl, ra, sl, lstr, str)
Input Parameters

`integer ldr`
Number of array elements used in `datrec`.

`character*80 datrec(ldr)`
Data record.

`integer ill`
Dimension of integer array.

`integer irl`
Dimension of real array.

`integer isl`
Dimension of character array.

Output Parameters

`integer errind`
Error indicator.

`integer il`
Number of integer entries.

`integer irl`
Array containing integer entries.

`integer rl`
Number of real entries.

`real ra (irl)`
Array containing real entries.

`integer sl`
Number of character string entries.

`integer lstr(isl)`
Length of each character string entry.

`character*(*) str(isl)`
Character string entries.

FORTRAN Subset

PUREC (`ldr, datrec, ill, irl, isl, errind, il, ra, rl, sl, lstr, str`)
Output Parameters

integer errind
   Error indicator.

integer il
   Number of integer entries.

integer ia (iil)
   Array containing integer entries.

integer rl
   Number of real entries.

real ra (irl)
   Array containing real entries.

integer sl
   Number of character string entries.

integer lstr(isl)
   Length of each character string entry.

character*80 str(isl)
   Character string entries.

Errors

None

Related Subroutines

- Initialize Choice
- Initialize Choice 3
- Initialize Locator
- Initialize Locator 3
- Initialize Pick
- Initialize Pick 3
- Initialize String
- Initialize String 3
- Initialize Stroke
- Initialize Stroke 3
- Initialize Valuator
- Initialize Valuator 3
- Inquire Choice Device State
- Inquire Choice Device State 3
- Inquire Default Choice Device Data
- Inquire Default Choice Device Data 3
- Inquire Default Locator Device Data
- Inquire Default Locator Device Data 3
- Inquire Default Pick Device Data
- Inquire Default Pick Device Data 3
- Inquire Default String Device Data
- Inquire Default String Device Data 3
- Inquire Default Stroke Device Data
• Inquire Default Stroke Device Data 3
• Inquire Default Valuator Device Data
• Inquire Default Valuator Device Data 3
• Inquire Locator Device State
• Inquire Locator Device State 3
• Inquire Pick Device State
• Inquire Pick Device State 3
• Inquire String Device State
• Inquire String Device State 3
• Inquire Stroke Device State
• Inquire Stroke Device State 3
• Inquire Valuator Device State
• Inquire Valuator Device State 3
• Pack Data Record
Chapter 13. Error Control Subroutines

The subroutines in this category allow your application to modify the error handling characteristics of the graPHIGS API system. By default, the graPHIGS API gives the error handling function control when the graPHIGS API detects an error. The graPHIGS API provides the default error handling function. However, your application can provide an alternative error handling function which supersedes the default and receives control when the graPHIGS API detects an error.

The default error handling function provided by the graPHIGS API calls the Error Logging subroutine function. An application program supplied Error Handling function may invoke inquiry subroutines, the Error Logging function, and/or the Emergency Close PHIGS subroutine.

If desired, you can use the Set Error Handling Mode subroutine to set error handling OFF. When your application sets error handling OFF, processing continues until the graPHIGS API reaches a terminating condition. Generally, error handling should remain ON in a program development environment. By default, the graPHIGS API sets error handling to ON.

EMERGENCY CLOSE PHIGS (PHCL, WSCL, STCL, ARCL)

Purpose

Use Emergency Close PHIGS to terminate all PHIGS processing for this application process. The graPHIGS API closes any open structure and updates and closes all open workstations. The graPHIGS API detaches all resources attached to or created by your application. The graPHIGS API closes all PHIGS files and releases all system resources, such as storage or locks. The graPHIGS API sets the PHIGS system state to PHIGS Closed. Your application can reopen PHIGS by invoking the Open PHIGS subroutine ( ).

Language Bindings

C

pemergency_close_phigs ()

FORTRAN

PECLPH

Errors

None

Related Subroutines

• Close PHIGS
• Open PHIGS

ERROR HANDLING (PHCL, WSCL, STCL, ARCL)

Purpose

Either the graPHIGS API supplies the Error Handling function by default, or the application supplies the function. The graPHIGS API calls the Error Handling subroutine function whenever the graPHIGS API detects an error. The default or standard Error Handling function calls the Error Logging function with the
same parameters. An application supplied error handling function, which accepts the same arguments as the default error handler, may also call the Error Logging function.

See the Open PHIGS subroutine for error file specification for each supported binding. See [Chapter 21: "Implementation Errors and graPHIGS API Messages for ISO PHIGS-Defined Errors"] for details on implementation errors and ISO PHIGS defined errors.

Users of the FORTRAN binding may replace the default error handling subroutine with a user supplied subroutine of the same name, PERHND. C binding users may use the Set Error Handling subroutine to install a user supplied error handling function during execution time.

**Language Bindings**

**C**

```c
perr_hand (error_num, func_num, error_file)
```

**Input Parameters**

- `Pint error_num`  
  Error number.

- `Pint func_num`  
  Identifier of function that detected the error (see [Chapter 17. ISO PHIGS C Type and Macro Definitions reference #1](#)).

- `const char *error_file`  
  Name of the error file.

**FORTRAN**

```fortran
PERHND (errnr, fctid, errfil)
```

**Input Parameters**

- `integer errnr`  
  Error number.

- `integer fctid`  
  Function identification (see [Chapter 18. ISO PHIGS FORTRAN Enumeration Types](#)).

- `integer errfil`  
  Error file.

**Errors**

None

**Related Subroutines**

- Emergency Close PHIGS
- Error Logging
- Open PHIGS
- Set Error Handling
- Set Error Handling Mode
ERROR LOGGING (PHCL,WSCL,STCL,ARCL)

Purpose

Use Error Logging to print an error message to the specified error file. When your application invokes Error Logging, the graPHIGS API writes the specified error message along with the name of the function which caused the error to the specified file. If the graPHIGS API cannot open the specified file or the file is blank, then the graPHIGS API logs the error to the console from which the application was started.

This subroutine is available only to an Error Handling function.

Language Bindings

C

perr_log (error_num, func_num, error_file)

Input Parameters

Pint error_num
    Error number.

Pint func_num
    Identifier of function that detected the error (see phigs.h include file).

const char *error_file
    Name of the error file.

FORTRAN

PERLOG (errnr, fctid, errfil)

Input Parameters

integer errnr
    Error number.

integer fctid
    Function identification (see PHIGS include file).

integer errfil
    Error file.

Errors

None

Related Subroutines

• Error Handling
• Set Error Handling
• Set Error Handling Mode

SET ERROR HANDLING (PHCL,WSCL,STCL,ARCL)

Purpose
Use Set Error Handling to set the PHIGS error handling function to the specified new error handling function. The graPHIGS API returns the previous error handling function on this call.

The graPHIGS API gives control to this application specified routine when the graPHIGS API detects an error. The application defined error handler must accept the same arguments as the standard error handler, **perr_hand**. Refer to **perr_hand** for a description of those arguments. If your application has not defined an error handler, then the graPHIGS API uses the default, **perr_hand**, which logs an error in the file specified by the first parameter of the Open PHIGS subroutine (**Open PHIGS**). An application defined error handler may invoke Inquiry subroutines (**Inquiry** subroutines) and the Error Logging subroutine.

**Note:** This subroutines is defined only for the C binding.

**Language Binding**

**C**

**pset_err_hand** (**new_err_hand**, **old_err_hand**)

**Input Parameters**

```c
void (**new_err_hand) ()
```

The address of the routine receiving control when the graPHIGS API detects an error.

**Output Parameters**

```c
void (**old_err_hand) ()
```

The address of the previous error handling routine.

**Errors**

None

**Related Subroutines**

- Error Handling
- Error Logging
- Inquire Error Handling Mode
- Open PHIGS
- Set Error Handling Mode

---

**SET ERROR HANDLING MODE (PHOP,*,*,*)**

**Purpose**

Use Set Error Handling Mode to enable or disable graPHIGS API error handling.

The graPHIGS API sets the error handling mode in the Error State List to the value specified. If your application sets the error handling mode to **OFF**, then the graPHIGS API ignores any errors detected. By default, the graPHIGS API sets error handling mode to **ON**.

**Language Bindings**

**C**

**pset_err_hand_mode** (**error_mode**)
Input Parameters

Perr_mode error_mode
Error handling mode (0=PERR_OFF, 1=PERR_ON).

FORTRAN

PSERHM (erhm)

Input Parameters

integer erhm
Error handling mode (0=POFF, 1=PON).

Errors

2 Function Requires State (PHOP,*,*,*)

Related Subroutines

• Inquire Error Handling Mode
• Set Error Handling
Chapter 14. Special Interface Subroutines

This section contains the definition of an escape mechanism for allowing access to hardware specific features. The use of this mechanism reduces portability of your application program, but it does it in an easily identifiable manner.

**ESCAPE (PHOP,WSCL,STCL,ARCL)**

**Purpose**

Use Escape to perform an escape function. The specified escape subroutine is identified by way of the subroutine identifier parameter. In general, an escape subroutine accepts both an input data record and an output data record to place any output generated by the escape subroutine.

If the graPHIGS API does not support the specified escape identifier, then the graPHIGS API ignores this subroutine. Currently, the graPHIGS API does not support any escape identifiers. However, your application can access escapes through the Escape (GPES) subroutine. See *The graPHIGS Programming Interface: Subroutine Reference* for details on those escape functions.

**Language Bindings**

**C**

```c
pescape (func_id, in_data, store, out_data);
```

**Input Parameters**

**Pint** `func_id`

Escape function identifier.

**const Pescape_in_data *in_data**

Input data for the function.

**Pstore store**

Handle to Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See the Create Store (**CREATE STORE (PHOP,*,*,*)**) subroutine for more information.

**Output Parameters**

**Pescape_out_data **out_data**

Output data of the function. The memory referenced by *out_data is managed by Store.

**FORTRAN**

```fortran
PESC (fctid, lidr, idr, mlodr, lodr, odr)
```

**Input Parameters**

**integer fctid**

Function identification.

**integer lidr**

Dimension of the input data record array.

**character*80 idr(lidr)**

Input data record.
integer mlodr
   Maximum length of the output data record.

Output Parameters

integer lodr
   Number of array elements occupied in odr.

character*80 odr(mlodr)
   Output data record.

Errors

2   Function Requires State (PHOP,*,*,*)
350  Warning, Specified Escape Unavailable On One Or More Workstations
351  One Of The Fields Within The Escape Data Record Is In Error

Related Subroutines

• None
Chapter 15. Inquire Subroutines

Inquiry programming subroutines allow application programs to obtain information such as the following:

- Default system characteristics
- Current state of the system
- Default workstation characteristics
- Current state of a workstation
- Configuration of a workstation
- Structure existence and relationships
- Structure content
- List of conflicting structures
- Ancestor and descendant path data
- Error state

Note: The graPHIGS API returns SET values for both REALIZED and SET output parameters.

General information for C binding inquiry subroutines

The graPHIGS API often requires a store parameter of type Pstore as input. See the Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

start_ind is an input parameter for inquiries that request simple lists. An index of zero specifies the first element in the graPHIGS API list, therefore, zero is a valid value for parameter start_ind. start_ind indicates the element in the graPHIGS API list that you want copied into index zero of the application’s list.

General information for FORTRAN binding inquiry subroutines

If the \( n^\text{th} \) list element that you requested is unavailable, then the graPHIGS API returns the error 2002 in the error indicator parameter and also returns the implementation’s list length.

If you request the \( 0^\text{th} \) list element, then the list element is undefined, but the graPHIGS API generates no error and returns the implementation’s list length.

**ELEMENT SEARCH (PHOP,*,*,*)**

**Purpose**

Use Element Search to search through a specified structure for an element that matches a given criteria. The search starts at a specified element position and searches in a designated direction (BACKWARD or FORWARD) until the graPHIGS API either finds an element that matches the criteria or reaches the limits of the structure.

**Search criteria:** The graPHIGS API selects an element if the type of the element is in the element inclusion list and not in the element exclusion list.

**Element exclusion:** The graPHIGS API excludes an element if the element type is either not in the inclusion list or it is in the exclusion list.

**Starting search position:** The search starts at element position of zero if the specified starting position is less than zero. The search starts with the last element in the structure, if the specified starting position is larger than the number of elements in the structure.
If the search is successful, then the application sets the status indicator to SUCCESS, and the graPHIGS API returns the element position in the position parameter. Otherwise, the application sets the status indicator to FAILURE and the value returned in the position parameter is unpredictable.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
201 Specified Structure Does Not Exist

Language Bindings

C

pelem_search (struct_id, start_elem, dir, incl, excl, err_ind, status, found_elem_ptr);

Input Parameters

Pint struct_id
   Structure identifier.

Pint start_elem
   Starting element position (>=0).

Psearch_dir dir
   Search direction (0=PDIR_BACKWARD, 1=PDIR_FORWARD).

cost Pelem_type_list *incl
   Element inclusion list (0=PELEM_ALL, 1=PELEM_NIL, 2=PELEM_POLYLINE3,
   3=PELEM_POLYLINE, 4=PELEM_POLYMARKER3, 5=PELEM_POLYMARKER,
   6=PELEM_TEXT3, 7=PELEM_TEXT, 8=PELEM_ANNO_TEXT_REL3,
   9=PELEM_ANNO_TEXT_REL, 10=PELEM_FILL_AREA3, 11=PELEM_FILL_AREA,
   12=PELEM_FILL_AREA_SET3, 13=PELEM_FILL_AREA_SET, 14=PELEM_CELL_ARRAY3,
   15=PELEM_CELL_ARRAY, 16=PELEM_GDP3, 17=PELEM_GDP, 18=PELEM_LINE_IND,
   19=PELEM_MARKER_IND, 20=PELEM_TEXT_IND, 21=PELEM_INT_IND,
   22=PELEM_EDGE_IND, 23=PELEM_LINETYPE, 24=PELEM_LINENTH,
   25=PELEM_LINE_COLR_IND, 26=PELEM_MARKER_TYPE, 27=PELEM_MARKER_SIZE,
   28=PELEM_MARKER_COI, 29=PELEM_TEXT_FONT, 30=PELEM_TEXT_PREC,
   31=PELEM_CHAR_EXPAN, 32=PELEM_CHAR_SPACE, 33=PELEM_TEXT_COLR_IND,
   34=PELEM_CHAR_HT, 35=PELEM_CHAR_UP_VEC, 36=PELEM_TEXT_PATH,
   37=PELEM_TEXT_ALIGN, 38=PELEM_ANNO_CHAR_HT, 39=PELEM_ANNO_CHAR_UP_VEC,
   40=PELEM_ANNO_PATH, 41=PELEM_ANNO_ALIGN, 42=PELEM_ANNO_STYLE,
   43=PELEM_INT_STYLE, 44=PELEM_INT_STYLE_IND, 45=PELEM_INT_COLR_IND,
   46=PELEM_EDGE_FLAG, 47=PELEM_EDGE_TYPE, 48=PELEM_EDGETYPE,
   49=PELEM_EDGE_COLR_IND, 50=PELEM_PAT_SIZE, 51=PELEM_PAT_REF_POINT_VECS,
   52=PELEM_PAT_REF_POINT, 53=PELEM_ADD_NAMES_SET,
   54=PELEM_REMOVE_NAMES_SET, 55=PELEM_INDIV_ASF, 56=PELEM_HLHSR_ID,
   57=PELEM_LOCAL_MODEL_TRAN3, 58=PELEM_LOCAL_MODEL_TRAN,
   59=PELEM_GLOBAL_MODEL_TRAN3, 60=PELEM_GLOBAL_MODEL_TRAN,
   61=PELEM_MODEL_CLIP_VOL3, 62=PELEM_MODEL_CLIP_VOL,
   63=PELEM_MODEL_CLIP_IND, 64=PELEM_RESTORE_MODEL_CLIP_VOL,
   65=PELEM_VIEW_IND, 66=PELEM_EXEC_STRUCT, 67=PELEM_LABEL,
   68=PELEM_APP_DATA, 69=PELEM_GSE, 70=PELEM_PICK_ID).

cost Pelem_type_list *excl
   Element exclusion list (Enumerated types are the same as that of the element inclusion list).
Output Parameters

Pint *err_ind
   Error indicator.

Psearch_status *status
   Search status indicator (0=PSEARCH_STATUS_FAILURE, 1=PSEARCH_STATUS_SUCCESS).

Pint *found_elem_ptr
   Found element position.

FORTRAN

PELS (strid, strtep, srcdir, eisn, eis, eesen, ees, errind, status, fndep)

Input Parameters

integer strid
   Structure identifier.

integer strtep
   Start element position(>=0).

integer srcdir
   Search direction (0=PBWD, 1=PFWD).

integer eis
   Number of elements in the element inclusion set.

integer eis(eisn)
   Element inclusion set (0=PEALL, 1=PENIL, 2=PEPL3, 3=PEPL, 4=PEPM3, 5=PEPM, 6=PETX3, 7=PETX, 8=PEATR3, 9=PEATR, 10=PEFA3, 11=PEFA, 12=PEFAS3, 13=PEFAS, 14=PECA3, 15=PECA, 16=PEGDP3, 17=PEGDP, 18=PEPLI, 19=PEPMI, 20=PETXI, 21=PEII, 22=PEEDI, 23=PELDN, 24=PELDN3, 25=PELDN1, 26=PELDN, 27=PELDN2, 28=PELDN, 29=PELXFN, 30=PELXPR, 31=PECHXP, 32=PECHXP3, 33=PECHXP1, 34=PECHXP, 35=PECHXP, 36=PECHXP, 37=PECHXP, 38=PECHXP, 39=PECHXP, 40=PECHXP, 41=PECHXP, 42=PECHXP, 43=PECHXP, 44=PECHXP, 45=PECHXP, 46=PECHXP, 47=PECHXP, 48=PECHXP, 49=PECHXP, 50=PECHXP, 51=PECHXP, 52=PECHXP, 53=PECHXP, 54=PECHXP, 55=PECHXP, 56=PECHXP, 57=PECHXP, 58=PECHXP, 59=PECHXP, 60=PECHXP, 61=PECHXP, 62=PECHXP, 63=PECHXP, 64=PECHXP, 65=PECHXP, 66=PECHXP, 67=PECHXP, 68=PECHXP, 69=PECHXP, 70=PECHXP).

integer eesen
   Number of elements in the element exclusion set.

integer ees(eesen)
   Element exclusion set (Enumerated types are the same as that of the element inclusion set).

Output Parameters

integer errind
   Error indicator.

integer status
   Status indicator (0=PFAIL, 1=PSUCC).

integer fndep
   Found element position.

Errors

None
Related Subroutines

- None

**INQUIRE ALL CONFLICTING STRUCTURES (PHOP,*,*,AROP)**

**Purpose**

Use Inquire All Conflicting Structures to inquire a list of structure identifiers that exist in both the structure store and the specified open archive file.

If the inquire information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 7  Function Requires State (PHOP,*,*,AROP)
- 404  Specified Archive File Is Not Open

**Language Bindings**

**C**

```c
pinq_all_conf_structs(ar_id, num_elems_appl_list, start_ind, err_ind, ids, num_elems_impl_list);
```

**Input Parameters**

- `Pint ar_id`
  - Archive file identifier.
- `Pint num_elems_appl_list`
  - Number of elements in the application list (>=0).
- `Pint start_ind`
  - Starting index (>=0).

**Output Parameters**

- `Pint *err_ind`
  - Error indicator.
- `Pint_list *ids`
  - List of conflicting structure identifiers.
- `Pint *num_elems_impl_list`
  - Number of elements in the implementation list.

**FORTRAN**

```fortran
PQCST (afid, n, errind, ol, ostrid)
```

**Input Parameters**

- `integer afid`
  - Archive file identifier.
- `integer n`
  - Set member requested (>=0).
Output Parameters

integer errind
Error indicator.

integer ol
Number of structure identifiers in list.

integer ostrid
nth structure identifier in list.

Errors

None

Related Subroutines

• Inquire Conflicting Structures in Network

INQUIRE ANNOTATION FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire Annotation Facilities to inquire the list of annotation styles and the number and range of annotation text character heights supported by a workstation. If the number of annotation text character heights is zero, then the graPHIGS API supports a continuous range from the minimum height to the maximum height. Possible annotation styles include: UNCONNECTED and LEAD LINE.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_anno_facs (ws_type, num elems appl list, start ind, err ind, styles, num elems_impl list, num anno char hts, min anno char ht, max anno char ht);

Input Parameters

Pint ws_type
Workstation type.

Pint num elems appl list
Number of elements in the application list (>=0).

Pint start ind
Starting index (>=0).

Output Parameters


\textit{Pint} \*err\_ind

Error indicator.

\textit{Pint\_list} \*styles

List of annotation styles (1=\texttt{PANNO\_STYLE\_UNCONNECTED}, 2=\texttt{PANNO\_STYLE\_LEAD\_LINE}).

\textit{Pint} \*num\_elems\_impl\_list

Number of elements in the implementation list.

\textit{Pint} \*num\_anno\_char\_hts

Number of available annotation text character heights.

\textit{Pfloat} \*min\_anno\_char\_ht

Minimum annotation text character height in DC.

\textit{Pfloat} \*max\_anno\_char\_ht

Maximum annotation text character height in DC.

\textbf{FORTRAN}

\texttt{PQANF} (\texttt{wtype}, \texttt{n}, \texttt{errind}, \texttt{nas}, \texttt{as}, \texttt{nchh}, \texttt{minchh}, \texttt{maxchh})

\textbf{Input Parameters}

\textit{integer} \texttt{wtype}

Workstation type.

\textit{integer} \texttt{n}

List element of the annotation styles requested (\texttt{>=0}).

\textbf{Output Parameters}

\textit{integer} \texttt{errind}

Error indicator.

\textit{integer} \texttt{nas}

Number of available annotation styles.

\textit{integer} \texttt{as}

\texttt{r}\textsuperscript{th} element of the list of available annotation styles (1=\texttt{PUNCON}, 2=\texttt{PLDLN}).

\textit{integer} \texttt{nchh}

Number of available annotation text character heights.

\textit{real} \texttt{minchh}

Minimum annotation text character height in DC.

\textit{real} \texttt{maxchh}

Maximum annotation text character height in DC.

\textbf{Errors}

None

\textbf{Related Subroutines}

- None

\textbf{INQUIRE ARCHIVE FILES (PHOP,*,*,*)}

\textbf{Purpose}

Use Inquire Archive Files to inquire the list of archive file identifiers and archive file names.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

`pinq_ar_files (store, err_ind, ar_files);`

Input Parameters

Pstore store
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See the Create Store (PHOP,*,*,*) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
Error indicator.

Par_file_list **ar_files
List of archive file names and identifiers. The memory referenced by *ar_files is managed by the parameter store.

FORTRAN

PQARF (n, errind, numberafid, arcfil)

Input Parameters

integer n
List element requested (>=0).

Output Parameters

integer errind
Error indicator.

integer number
Number of archive files open.

integer afid
nth open archive file identifier.

integer arcfil
nth open archive file name.

Errors

None

Related Subroutines

• None
INQUIRE ARCHIVE STATE VALUE (PHCL,WSCL,STCL,ARCL)

Purpose

Use Inquire Archive State Value to inquire the current archive state value of the graPHIGS API. The archive state is either Archive Open (AROP) or Archive Closed (ARCL). If the state is AROP, then at least one archive file is open. If the state is ARCL, then no archive files are open.

Language Bindings

C

pinq_ar_st (ar_st)

Output Parameters

Par_st *ar_st
  Archive state value (0=PST_ARCL, 1=PST_AROP).

FORTRAN

PQARS (arsta)

Output Parameters

integer arsta
  Archive state value (0=PARCL, 1=PAROP).

Errors

None

Related Subroutines

• None

INQUIRE CHOICE DEVICE STATE (PHOP,WSOP,*,*)

Purpose

Use Inquire Choice Device State to inquire the current state of the specified choice device on the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings
C

\texttt{pinq\_choice\_st} (\texttt{ws\_id, choice\_num, store, err\_ind, op\_mode, echo\_switch, init\_status, init\_choice, prompt\_echo, echo\_area, choice\_data})

\textbf{Input Parameters}

\textit{Pint ws\_id}

Workstation identifier.

\textit{Pint choice\_num}

Choice device number (\texttt{>=1}).

\textit{Pstore store}

Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See the \texttt{Create Store} subroutine for details on how the graPHIGS API uses this parameter on inquiries.

\textbf{Output Parameters}

\textit{Pint *err\_ind}

Error indicator.

\textit{Pop\_mode *op\_mode}

Operating mode (\texttt{0=POP\_REQ, 1=POP\_SAMPLE, 2=POP\_EVENT}).

\textit{Pecho\_switch *echo\_switch}

Echo switch (\texttt{0=PSWITCH\_NO\_ECHO, 1=PSWITCH\_ECHO}).

\textit{Pin\_status *init\_status}

Initial choice status (\texttt{1=PIN\_STATUS\_OK, 2=PIN\_STATUS\_NO\_IN}).

\textit{Pint *init\_choice}

Initial choice number.

\textit{Pint *prompt\_echo}

Prompt and echo type.

\textit{Plimit *echo\_area}

Echo area.

\textit{Pchoice\_data3 **choice\_data}

Data record. The memory referenced by \texttt{*choice\_data} is managed by the parameter \texttt{store}.

\textbf{FORTRAN}

\texttt{PQCHS (wkid, chdnr, mldr, errind, mode, esw, istat, ichnr, pet, earea, ldr, datrec)}

\textbf{Input Parameters}

\textit{integer wkid}

Workstation identifier.

\textit{integer chdnr}

Choice device number (\texttt{>=1}).

\textit{integer mldr}

Dimension of data record array.

\textbf{Output Parameters}
integer errind
    Error indicator.

integer mode
    Operating mode (0=PREQ, 1=PAMPL, 2=PEVENT).

integer esw
    Echo switch (0=PNECHO, 1=PECHO).

integer istat
    Initial choice status (1=POK, 2=PNCHOI).

integer ichnr
    Initial choice number.

integer pet
    Prompt and echo type.

real earea(4)
    Echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
    Number of the array elements used in the data record.

character*80 datrec(mldr)
    Data record.

Errors
None

Related Subroutines
• Initialize Choice
• Initialize Choice 3
• Inquire Choice Device State 3

INQUIRE CHOICE DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose
Use Inquire Choice Device State 3 to inquire the current state of the specified choice device on the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3   Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings
C

ping_choice_st3 (ws_id, choice_num, store, err_ind, op_mode, echo_switch, init_status, init_choice, prompt_echo, echo_vol, choice_data)
Input Parameters

**Pint ws_id**
Workstation identifier.

**Pint choice_num**
Choice device number (>=1).

**Pstore store**
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store CREATE STORE (PHOP; ; ;;) subroutine for details on how the graPHIGS API uses this parameter on inquires.

Output Parameters

**Pint *err_ind**
Error indicator.

**Pop_mode *op_mode**
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

**Pecho_switch *echo_switch**
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

**Pin_status *init_status**
Initial choice status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

**Pint *init_choice**
Initial choice number.

**Pint *prompt_echo**
Prompt and echo type.

**Plimit3 *echo_vol**
Prompt and echo volume.

**Pchoice_data3 **choice_data**
Data record. The memory referenced by *choice_data is managed by the parameter store.

FORTRAN

PQCHS3 (wkid, chdnr, mldr, errind, mode, esw, istat, ichnr, pet, evol, ldr, datrec)

Input Parameters

integer wkid
Workstation identifier.

integer chdnr
Choice device number (>=1).

integer mldr
Dimension of data record array.

Output Parameters

integer errind
Error indicator.

integer mode
Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
Echo switch (0=PNECHO, 1=PECHO).
integer istat
Initial choice status (1=POK, 2=PNCHOI).

integer ichnr
Initial choice number.

integer pet
Prompt and echo type.

real evol(6)
Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
Number of the array elements used in the data record.

character*80 datrec(mldr)
Data record.

Errors
None

Related Subroutines
• Initialize Choice
• Initialize Choice 3
• Inquire Choice Device State 3

INQUIRE COLOR FACILITIES (PHOP,**,**)

Purpose
Use Inquire Color Facilities to inquire the color facilities of a specified workstation type.

The graPHIGS API returns the total number of available colors, the color status, (monochrome or color), the quantity of predefined color table entries in the workstation’s default color table, and the primary colors for the workstation display. The graPHIGS API returns the primary colors as the CIELUV chromaticity coefficients $u'$, $v'$ and luminance value $y$.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,**,**)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_colr_facs (ws_type, err_ind, fac)

Input Parameters
Pint ws_type
  Workstation type.

Output Parameters
Pint *err_ind
  Error indicator.
Pcolr_facs *fac
  Color facilities.

FORTRAN
PQCF (wtype, errind, ncoli, cola,npci, cc)

Input Parameters
integer wtype
  Workstation type.

Output Parameters
integer errind
  Error indicator.
integer ncoli
  Number of color indexes.
integer cola
  Color available (0=PMONOC, 1=PCOLOR).
integer npci
  Number of predefined color indexes.
real cc(9)
  Primary colors. Chromaticity coefficients and luminance value for the primaries for the display device ($R_u$, $R_v$, $R_y$, $G_u$, $G_v$, $G_y$, $B_u$, $B_v$, $B_y$).

Errors
None

Related Subroutines
• None

INQUIRE COLOR MODEL (PHOP,WSOP,*,*)

Purpose
Use Inquire Color Model to inquire the current color model for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability
Language Bindings

C

pinq_colr_model (ws_id, err_ind, model)

Input Parameters

Pint ws_id
Workstation identifier.

Output Parameters

Pint *err_ind
Error indicator.

Pint *model
Current color model (1=PMODEL_RGB, 2=PMODEL_CIELUV, 3=PMODEL_HSV).

FORTRAN

PQCMD (wkid, errind, cmodel)

Input Parameters

integer wkid
Workstation identifier.

Output Parameters

integer errind
Error indicator.

integer cmodel
Current color model (1=PRGB, 2=PCIE, 3=PHSV).

Errors

None

Related Subroutines

• Inquire Color Model Facilities
• Set Color Model

INQUIRE COLOR MODEL FACILITIES (PHOP,**,*)

Purpose

Use Inquire Color Model Facilities to inquire the list of available color models and the default color model on the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,**,*)
52 Workstation Type Not Recognized By Implementation
Language Bindings

C

pinq_colr_model_facs (ws_type, num elems appl_list, start_ind, err ind, models, num elems impl_list, def)

Input Parameters

Pint ws_type
   Workstation type.

Pint num elems appl_list
   Number of elements in the application list (>=0).

Pint start ind
   Starting index (>=0).

Output Parameters

Pint *err ind
   Error indicator.

Pint_list *models
   List of color models (1=PMODEL_RGB, 2=PMODEL_CIELUV, 3=PMODEL_HSV).

Pint *num elems impl_list
   Number of elements in the implementation list.

Pint *def
   Default color model (1=PMODEL_RGB, 2=PMODEL_CIELUV, 3=PMODEL_HSV).

FORTRAN

PQCMD (wtype, n, errind, ol, cmod, dfcmod)

Input Parameters

integer wtype
   Workstation type.

integer n
   Element of the list of available color models (>=0).

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of available color models.

integer cmod
   nth available color model (1=PRGB, 2=PCIE, 3=PHSV).

integer dfcmod
   Default color model (1=PRGB, 2=PCIE, 3=PHSV).
Errors
None

Related Subroutines
- Inquire Color Model
- Set Color Model

INQUIRE COLOR REPRESENTATION (PHOP, WSOP, *, *)

Purpose
Use Inquire Color Representation to inquire the current color values in the specified workstation’s color table. The color specification parameters are the coordinates of the color in the current color model at the workstation.

If the type of returned values is REALIZED and your application has neither predefined nor set the color associated with the color index, or the color index is greater than the range of the color table at the workstation, then the graPHIGS API sets the output color parameters to the color associated with color index 1.

If the type of returned values is SET and your application has predefined or set the color associated with the color index, then the graPHIGS API sets the output color parameters as closely as possible to the color associated with the color index as it was predefined or set. This may be the same as the case when the type of returned value is REALIZED.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- Function Requires State (PHOP, WSOP, *, *)
- Specified Workstation Is Not Open
- Specified Workstation Does Not Have Output Capability
- Color Index Value < ZERO
- Specified Representation Has Not Been Defined

Use Inquire Workstation State Table Lengths subroutine to determine the actual size of the workstation’s color table.

Language Bindings

C

cinq_colr_rep (ws_id, colr_ind, type, err_ind, colr_rep)

Input Parameters

- Pint ws_id
  Workstation identifier.
- Pint colr_ind
  Color index (>=0).
- Pinq_type type
  Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).
Output Parameters

*err_ind
  Error indicator.

colr_rep
  Color representation.

FORTRAN

PQCR (wkid, coli, ccsbsz, type, errind, ol, cspec)

Input Parameters

integer wkid
  Workstation identifier.

integer coli
  Color index (>=0).

integer ccsbsz
  Color component specification buffer size.

integer type
  Type of returned values (0=PSET, 1=PREAL).

Output Parameters

integer errind
  Error indicator.

integer ol
  Number of color components in the color specification.

real cspec(ccsbsz)
  Color specification.

Errors

None

Related Subroutines

- Set Color Representation

INQUIRE CONFLICT RESOLUTION (PHOP,*,*,*)

Purpose

Use Inquire Conflict Resolution to inquire the archival conflict resolution flag and the retrieval conflict resolution flag. Each flag may have the value MAINTAIN, ABANDON, or UPDATE.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2  Function Requires State (PHOP,*,*,*)

Language Bindings
C

pinq_conf_res (err_ind, archive_res, retrieve_res)

Output Parameters

Pint *err_ind
   Error indicator.

Pconf_res *archive_res
   Archival conflict resolution (0=PRES_MAINTAIN, 1=PRES_ABANDON, 2=PRES_UPD).

Pconf_res *retrieve_res
   Retrieval conflict resolution (0=PRES_MAINTAIN, 1=PRES_ABANDON, 2=PRES_UPD).

FORTRAN

PQCNRS (errind, arccr, retrcr)

Output Parameters

integer errind
   Error indicator.

integer arccr
   Archival conflict resolution (0=PCRMNT, 1=PCRABA, 2=PCRUPD).

integer retrcr
   Retrieval conflict resolution (0=PCRMNT, 1=PCRABA, 2=PCRUPD).

Errors

None

Related Subroutines

• None

INQUIRE CONFLICTING STRUCTURES IN NETWORK (PHOP,*,*,AROP)

Purpose

Use Inquire Conflicting Structures in Network to inquire a list of structure identifiers from a specified structure network that exists in both the structure store and the specified open archive file.

The value for the source determines whether the structure network originates from the structure store or from the archive file.

If the inquire information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

7     Function Requires State (PHOP,*,*,AROP)
201   Specified Structure Does Not Exist
404   Specified Archive File Is Not Open

Language Bindings

276   The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
C

pinq_conf_structs_net(ar_id, struct_id, source, num_elems_appl_list, start_ind, err_ind, ids, num_elems_impl_list)

Input Parameters

Pint ar_id
   Archive file identifier.

Pint struct_id
   Structure identifier.

Pstruct_net_source source
   Structure network source (0=PNET_CSS, 1=PNET_AR).

Pint num_elems_appl_list
   Number of elements in the application list (>=0).

Pint start_ind
   Starting index (>=0).

Output Parameters

Pint *err_ind
   Error indicator.

Pint_list *ids
   List of conflicting structure identifiers.

Pint *num_elems_impl_list
   Number of elements in the implementation list.

FORTRAN

PQCSTN (afid, strid, snsrc, n, errind, ol, ostrid)

Input Parameters

integer afid
   Archive file identifier.

integer strid
   Structure identifier.

integer snsrc
   Structure network source (0=PCSS, 1=PARCHV).

integer n
   Set member requested (>=0).

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of structure identifiers in list.

integer ostrid
   nth structure identifier in list.

Errors
Related Subroutines

• Inquire All Conflicting Structures

INQUIRE CURRENT ELEMENT CONTENT (PHOP,*,STOP,*)

Purpose

Use Inquire Current Element Content to retrieve the current structure element content that is indicated by the element pointer.

This subroutine returns the data contained in the current element. The graPHIGS API returns the data in a binding specific format. See Chapter 17, “ISO PHIGS C Type and Macro Definitions” for the C binding formats and see Chapter 16, “FORTRAN Structure Content Data Records” for the FORTRAN binding formats. Use the Inquire Current Element Type And Size subroutine to determine the element type and size.

To execute this subroutine, a structure must be open.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

5 Function Requires State (PHOP,*,STOP,*)

Language Bindings

C

pinq_cur_elem_content (store, err_ind, elem_data)

Input Parameters

Pstore store
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store CREATE STORE (PHOP,*,*,*) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
Error indicator.

Pelem_data **elem_data
Element data. The memory referenced by *elem_data is managed by the parameter store.

FORTRAN

PQCECO (iil, irl, isl, errind, il, ia, rl, ra, sl, lstr, str)

Input Parameters

integer iil
Dimension of integer array (>=0).
integer irl
    Dimension of real array (>=0).

integer isl
    Dimension of character array (>=0).

Output Parameters
integer errind
    Error indicator.

integer il
    Number of integer entries.

integer ia(iil)
    Array containing integer entries.

integer rl
    Number of real entries.

integer ra(irl)
    Array containing real entries.

integer sl
    Number of character string entries.

integer lstr isl)
    Length of each character string entry.

character(*) str isl)
    Character string entries.

FORTRAN Subset
PQCECO (iil, irl, isl, errind, il, ia, rl, ra, sl, lstr, str)

Input Parameters
integer iil
    Dimension of integer array (>=0).

integer irl
    Dimension of real array (>=0).

integer isl
    Dimension of character array (>=0).

Output Parameters
integer errind
    Error indicator.

integer il
    Number of integer entries.

integer ia(iil)
    Array containing integer entries.

integer rl
    Number of real entries.

integer ra(irl)
    Array containing real entries.
integer $sl$

   Number of character string entries.

integer lstr($isl$)

   Length of each character string entry.

character*80 str($isl$)

   Character string entries.

Errors

None

Related Subroutines

• [Inquire Current Element Type And Size](#)

**INQUIRE CURRENT ELEMENT TYPE AND SIZE (PHOP,*;STOP,*))**

**Purpose**

Use Inquire Current Element Type and Size to inquire the type and size of the current element.

The graPHIGS API returns the element of the structure element pointed to by the element pointer. If elements of this type have no associated values, then the graPHIGS API returns a value of zero in the element size parameter. If the element pointer is currently zero, then the graPHIGS API returns a *NIL* value in the element type parameter. To retrieve the element contents, use the [Inquire Current Element Content](#) subroutine.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

5   Function Requires State (PHOP,*;STOP,*).

**Language Bindings**

**C**

`pinq_cur_elem_type_size (err_ind, elem_type, elem_size)`

**Output Parameters**

Pint *err_ind

   Error indicator.

Pelem_type *elem_type

   Element type (1=PELEM_NIL, 2=PELEM_POLYLINE3, 3=PELEM_POLYLINE, 4=PELEM_POLYMARKER3, 5=PELEM_POLYMARKER, 6=PELEM_TEXT3, 7=PELEM_TEXT, 8=PELEM_ANNO_TEXT_REL3, 9=PELEM_ANNO_TEXT_REL, 10=PELEM_FILL_AREA3, 11=PELEM_FILL_AREA, 12=PELEM_FILL_AREA_SET3, 13=PELEM_FILL_AREA_SET, 14=PELEM_CELL_ARRAY3, 15=PELEM_CELL_ARRAY, 16=PELEM_GDP3, 17=PELEM_GDP, 18=PELEM_LINE_IND, 19=PELEM_MARKER_IND, 20=PELEM_TEXT_IND, 21=PELEM_INT_IND, 22=PELEM_EDGE_IND, 23=PELEM_LINETYPE, 24=PELEM_LINEWIDTH, 25=PELEM_LINE_COLR_IND, 26=PELEM_MARKER_TYPE, 27=PELEM_MARKER_SIZE, 28=PELEM_MARKER_COLR_IND, 29=PELEM_TEXT_FONT, 30=PELEM_TEXT_PREC, 31=PELEM_CHAR_EXPAN, 32=PELEM_CHAR_SPACE, 33=PELEM_TEXT_COLR_IND, 34=PELEM_CHAR_HT, 35=PELEM_CHAR_UP_VEC, 36=PELEM_TEXT_PATH, 37=PELEM_TEXT_ALIGN, 38=PELEM_ANNO_CHAR_HT, 39=PELEM_ANNO_CHAR_UP_VEC, 40=PELEM_ANNO_CHAR_COLR, 41=PELEM_ANNO_CHAR_PREC, 42=PELEM_ANNO_CHAR_ALIGN)
size_t *elem_size
Element size in bytes.

FORTRAN
PQCETS (errind, eltype, il, rl, sl)

Output Parameters
integer errind
Error indicator.

integer eltype
Element type (1=PENIL, 2=PEPL3, 3=PEPL, 4=PEPM3, 5=PEPM, 6=PETX3, 7=PETX, 8=PEATR3, 9=PEATR, 10=PEFA3, 11=PEFA, 12=PEFAS3, 13=PEFAS, 14=PECA3, 15=PECA, 16=PEGDP3, 17=PEGDP, 18=PEPLI, 19=PEPMI, 20=PETXI, 21=PEII, 22=PEEDI, 23=PELN, 24=PELWSC, 25=PEPLCI, 26=PEMK, 27=PEMKSC, 28=PEPMCI, 29=PETXFN, 30=PETXPR, 31=PECHXP, 32=PECHSP, 33=PETXCI, 34=PECHH, 35=PECHUP, 36=PETXP, 37=PETXAL, 38=PEATCH, 39=PEATCU, 40=PEATP, 41=PEATAL, 42=PEANST, 43=PEIS, 44=PEISI, 45=PEICI, 46=PEEDFG, 47=PEEDT, 48=PEEWSC, 49=PEEDCI, 50=PEPA, 51=PEPRPV, 52=PEPARF, 53=PEADS, 54=PERES, 55=PEIASF, 56=PEHRID, 57=PELMT3, 58=PELMT, 59=PEGMT3, 60=PEGMT, 61=PEMVC3, 62=PEMVC, 63=PEMCLI, 64=PERMCV, 65=PEVWI, 66=PEEXST, 67=PELB, 68=PEAP, 69=PEGSE, 70=PEPKID).

integer il
Dimension of integer array.

integer rl
Dimension of real array.

integer sl
Dimension of character array.

Errors
None

Related Subroutines
• Inquire Current Element Content

INQUIRE DEFAULT CHOICE DEVICE DATA (PHOP,*,*,*)

Purpose
Use Inquire Default Choice Device Data to inquire the default values for a specified choice device for the specified workstation type.

The graPHIGS API returns the default values for the requested choice device. For more information on the defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **Function Requires State (PHOP,*,*,*)**
- **Workstation Type Not Recognized By Implementation**
- **Information Not Available For Generic Workstation Type**
- **Specified Workstation Is Not Of Category Input Or Outin**
- **Specified Device Not Available On Workstation**

**Language Bindings**

**C**

`pinq_def_choice_data (ws_type, choice_num, store, err_ind, max_choices, pet_list, echo_area, choice_data)`

**Input Parameters**

- `Pint ws_type`: Workstation type.
- `Pint choice_num`: Choice device number (>=1).
- `Pstore store`: Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store `CREATE STORE (PHOP,*,*,*)` subroutine for details on how the graPHIGS API uses this parameter on inquiries.

**Output Parameters**

- `Pint *err_ind`: Error indicator.
- `Pint *max_choices`: Maximum number of choice alternatives.
- `Pint_list **pet_list`: List of prompt and echo types. The memory referenced by `*pet_list` is managed by the parameter store.
- `Plimit *echo_area`: Default echo area in DC.
- `Pchoice_data **choice_data`: Default choice data record. The memory referenced by `*choice_data` is managed by the parameter store.

**FORTRAN**

`PQDCH (wtype, devno, n, mldr, errind, malt, ol, pet, earea, ldr, datrec)`

282  *The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference*
Input Parameters

`integer wtype`
Workstation type.

`integer devno`
Choice device number (>=1).

`integer n`
List element requested (>=0).

`integer mldr`
Dimension of data record array (>=0).

Output Parameters

`integer errind`
Error indicator.

`integer malt`
Maximum number of choice alternatives.

`integer ol`
Number of available prompt and echo types.

`integer pet`
$n^{th}$ element of the list of the available prompt and echo types.

`real earea(4)`
Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

`integer ldr`
Number of array elements used in the data record.

`character*80 datrec(mldr)`
Default choice data record.

Errors

None

Related Subroutines

• `Initialize Choice`
• `Initialize Choice 3`
• `Inquire Choice Device State`
• `Inquire Choice Device State 3`
• `Inquire Default Choice Device Data 3`
• `Inquire Workstation Connection And Type`

INQUIRE DEFAULT CHOICE DEVICE DATA 3 (PHOP,*,*,*)

Purpose

Use Inquire Default Choice Device Data 3 to inquire the default values for a specified choice device for the specified workstation type.

The graPHIGS API returns the default values for the requested choice device. For more information on the defaults, see The graPHIGS Programming Interface: Technical Reference.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 2 Function Requires State (PHOP,*,*,*)
- 52 Workstation Type Not Recognized By Implementation
- 51 Information Not Available For Generic Workstation Type
- 61 Specified Workstation Is Not Of Category Input Or Outin
- 250 Specified Device Not Available On Workstation

### Language Bindings

**C**

```c
pinq_def_choice_data3 (ws_type, choice_num, store, err_ind, max_choices, pet_list, echo_vol, choice_data)
```

#### Input Parameters

- `Pint ws_type`
  - Workstation type.
- `Pint choice_num`
  - Choice device number (>=1).
- `Pstore store`
  - Handle to the Store object. The graPHIGS API uses an object of type `Store` to facilitate the use of subroutines which return complex data. See `Create Store` ([CREATE STORE (PHOP,*,*,*)](#)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

#### Output Parameters

- `Pint *err_ind`
  - Error indicator.
- `Pint *max_choices`
  - Maximum number of choice alternatives.
- `Plist **pet_list`
  - List of prompt and echo types. The memory referenced by `*pet_list` is managed by the parameter `store`.
- `Plimit3 *echo_vol`
  - Default echo volume in DC.
- `Pchoice_data3 **choice_data`
  - Default choice data record. The memory referenced by `*choice_data` is managed by the parameter `store`.

**FORTRAN**

```fortran
PQDCH3 (wtype, devno, n, mldr, errind, malt, ol, pet, evol, ldr, datrec)
```

#### Input Parameters

- `integer wtype`
  - Workstation type.
integer devno
   Choice device number (>=1).

integer n
   List element requested (>=0).

integer mldr
   Dimension of data record array (>=0).

Output Parameters

integer errind
   Error indicator.

integer malt
   Maximum number of choice alternatives.

integer ol
   Number of available prompt and echo types.

integer pet
   \( n^{th} \) element of the list of the available prompt and echo types.

real evol(6)
   Default echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Number of array elements used in the data record.

character*80 datrec(mldr)
   Default choice data record.

Errors

None

Related Subroutines

- Initialize Choice
- Initialize Choice 3
- Inquire Choice Device State
- Inquire Choice Device State
- Inquire Default Choice Device Data 3
- Inquire Workstation Connection And Type

INQUIRE DEFAULT DISPLAY UPDATE STATE (PHOP,*,*,*)

Purpose

Use Inquire Default Display Update State to inquire the default values of the deferral state and modification mode for the specified workstation type.

For an explanation of the deferral states and modification modes see The graPHIGS Programming Interface: Understanding Concepts.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2   Function Requires State (PHOP,*,*,*)
Workstation Type Not Recognized By Implementation

Information Not Available For Generic Workstation Type

Specified Workstation Does Not Have Output Capability

This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_def_disp_upd_st (ws_type, err_ind, def_mode, mod_mode)

Input Parameters

Pint ws_type
  Workstation type.

Output Parameters

Pint *err_ind
  Error indicator.

Pdefer_mode *def_mode
  Default value for deferral mode (0=PDEFER_ASP, 1=PDEFER_BNIG, 2=PDEFER_BNIL, 3=PDEFER_ASTI, 4=PDEFER_WAIT).

Pmod_mode *mod_mode
  Default value for modification mode (0=PMODE_NIVE, 1=PMODE_UWOR, 2=PMODE_UQUM).

FORTRAN

PQDDUS (wtype, errind, defmod, modmod)

Input Parameters

integer wtype
  Workstation type.

Output Parameters

integer errind
  Error indicator.

integer defmod
  Default value for deferral mode (0=PASAP, 1=PBNIG, 2=PBNIL, 3=PASTI, 4=PWAITD).

integer modmod
  Default value for modification mode (0=PNIVE, 1=PUWOR, 2=PUQUM).

Errors

None

Related Subroutines

• Inquire Display Update State
• Inquire Workstation Connection And Type
• Set Display Update State
INQUIRE DEFAULT LOCATOR DEVICE DATA (PHOP,*,*,*)

Purpose

Use Inquire Default Locator Device Data to inquire the default value of the specified locator device for the specified workstation type.

The graPHIGS API returns the default values for the requested locator device. The default initial locator position is in view zero, which has the highest view input priority by default. For more information on the defaults, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2   Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_loc_data (ws_type, loc_num, store, err_ind, loc_pos, pet_list, echo_area, loc_data)

Input Parameters

Pint ws_type  
   Workstation type.

Pint loc_num  
   Locator device number (>=1).

Pstore store  
   Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store [CREATE STORE (PHOP,*,*,*)] subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind  
   Error indicator.

Ppoint *loc_pos  
   Default initial locator position in WC.

Pint_list **pet_list  
   List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit *echo_area  
   Default echo area in DC.

Ploc_data **loc_data  
   Default locator data record. The memory referenced by *loc_data is managed by the parameter store.
PQDLC (wtype, devno, n, mldr, errind, dpx, dpy, ol, pet, earea, ldr, datrec)

Input Parameters

integer wtype
    Workstation type.

integer devno
    Locator device number (>=1).

integer n
    List element requested (>=0).

integer mldr
    Dimension of data record array (>=0).

Output Parameters

integer errind
    Error indicator.

integer dpx
    x coordinate of the default initial locator position in WC.

integer dpy
    y coordinate of the default initial locator position in WC.

integer ol
    Number of available prompt and echo types.

integer pet
    nᵗʰ element of the list of the available prompt and echo types.

real earea(4)
    Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    Data record.

Errors

None

Related Subroutines

- Initialize Locator
- Initialize Locator 3
- Inquire Locator Device State
- Inquire Locator Device State 3
- Inquire Default Locator Device Data 3
- Inquire Workstation Connection And Type
Use Inquire Default Locator Device Data 3 to inquire the default value of the specified locator device for the specified workstation type.

The graPHIGS API returns the default values for the requested locator device. The default initial locator position is in view zero, which has the highest view input priority by default. For more information on the defaults, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2   Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
61  Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_loc_data3 (ws_type, loc_num, store, err_ind, loc_pos, pet_list, echo_vol, loc_data)

Input Parameters

Pint ws_type
   Workstation type.

Pint loc_num
   Locator device number (>=1).

Pstore store
   Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store CREATE STORE (PHOP,*,*) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Ppoint3 *loc_pos
   Default initial locator position in WC.

Pint_list **pet_list
   List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit3 *echo_vol
   Default echo volume in DC.

Ploc_data3 **loc_data
   Default locator data record. The memory referenced by *loc_data is managed by the parameter store.

FORTRAN

PQDLC3 (wtype, devno, n, mldr, errind, dpx, dpy, dpz, ol, pet, evol, ldr, datrec)
**Input Parameters**

integer wtype  
Workstation type.

integer devno  
Locator device number (>=1).

integer n  
List element requested (>=0).

integer mldr  
Dimension of data record array (>=0).

**Output Parameters**

integer errind  
Error indicator.

integer dpx  
$x$ coordinate of the default initial locator position in WC.

integer dpy  
$y$ coordinate of the default initial locator position in WC.

integer dpz  
$z$ coordinate of the default initial locator position in WC.

integer ol  
Number of available prompt and echo types.

integer pet  
$n$th element of the list of the available prompt and echo types.

real evol(6)  
Default echo volume in DC ($XMIN$, $XMAX$, $YMIN$, $YMAX$, $ZMIN$, $ZMAX$).

integer ldr  
Number of array elements used in the data record.

character*80 datrec(mldr)  
Data record.

**Errors**

None

**Related Subroutines**

- Initialize Locator
- Initialize Locator 3
- Inquire Locator Device State
- Inquire Locator Device State 3
- Inquire Default Locator Device Data
- Inquire Workstation Connection And Type

**INQUIRE DEFAULT PICK DEVICE DATA (PHOP,*,*,*)**

**Purpose**
Use Inquire Default Pick Device Data to return the default values of the specified pick device for the specified workstation type.

The graPHIGS API returns the default values for the requested pick device. For more information on defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP;*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

`pinq_def_pick_data (ws_type, pick_num, store, err_ind, pet_list, echo_area, pick_data)`

**Input Parameters**

`Pint ws_type`

Workstation type.

`Pint pick_num`

Pick device number (>=1).

`Pstore store`

Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store `CREATE STORE (PHOP;*,*,*)` subroutine for details on how the graPHIGS API uses this parameter on inquiries.

**Output Parameters**

`Pint *err_ind`

Error indicator.

`Pint_list **pet_list`

List of prompt and echo types. The memory referenced by `*pet_list` is managed by the parameter `store`.

`Plimit *echo_area`

Default echo area in DC.

`Ppick_data **pick_data`

Default pick data record. The memory referenced by `*pick_data` is managed by the parameter `store`.

FORTRAN

`PQDPK (wtype, devno, n, mldr, errind, ol, pet, earea, ldr, datrec)`

**Input Parameters**
integer wtype
    Workstation type.

integer devno
    Pick device number (>=1).

integer n
    List element requested (>=0).

integer mldr
    Dimension of data record array (>=0).

Output Parameters

integer errind
    Error indicator.

integer ol
    Number of available prompt and echo types.

integer pet
    $n^{th}$ element of the list of the available prompt and echo types.

real earea(4)
    Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
    Number of array elements used in the data record.

canonical*80 datrec(mldr)
    Data record.

Errors

None

Related Subroutines

• Initialize Pick
• Initialize Pick 3
• Inquire Pick Device State
• Inquire Pick Device State 3
• Inquire Default Pick Device Data
• Inquire Workstation Connection And Type

INQUIRE DEFAULT PICK DEVICE DATA 3 (PHOP,*,*,*)

Purpose

Use Inquire Default Pick Device Data 3 to return the default values of the specified pick device for the specified workstation type.

The graPHIGS API returns the default values for the requested pick device. For more information on defaults, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
Language Bindings

C

pinq_def_pick_data3 (ws_type, pick_num, store, err_ind, pet_list, echo_vol, pick_data)

Input Parameters

Pint ws_type
    Workstation type.

Pint pick_num
    Pick device number(>=1).

Pstore store
    Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
    Error indicator.

Pint_list **pet_list
    List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit3 *echo_vol
    Default echo volume in DC.

Ppick_data3 **pick_data
    Default pick data record. The memory referenced by *pick_data is managed by the parameter store.

FORTRAN

PQDPK3 (wtype, devno, n, mldr, errind, ol, pet, evol, ldr, datrec)

Input Parameters

integer wtype
    Workstation type.

integer devno
    Pick device number (>=1).

integer n
    List element requested (>=0).

integer mldr
    Dimension of data record array (>=0).

Output Parameters
integer errind
   Error indicator.

integer ol
   Number of available prompt and echo types.

integer pet
   \( n^{th} \) element of the list of the available prompt and echo types.

real evol(6)
   Default echo volume in DC \((XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX)\).

integer ldr
   Number of array elements used in the data record.

character*80 datrec(mldr)
   Data record.

Errors

None

Related Subroutines

- Initialize Pick
- Initialize Pick 3
- Inquire Pick Device State
- Inquire Pick Device State 3
- Inquire Default Pick Device Data 3
- Inquire Workstation Connection And Type

INQUIRE DEFAULT STRING DEVICE DATA (PHOP,*,*,*)

Purpose

Use Inquire Default String Device Data to inquire the default values of the specified string device for the specified workstation type.

The graPHIGS API returns the default values for the requested string device. For more information on default values, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 2 Function Requires State (PHOP,*,*,*)
- 52 Workstation Type Not Recognized By Implementation
- 51 Information Not Available For Generic Workstation Type
- 61 Specified Workstation Is Not Of Category Input Or Outin
- 250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_string_data \((ws\text{\_}type, \text{string\_}num, \text{store, \text{err\_}ind, max\_buf\_size, pet\_list, echo\_area, string\_data})\)
Input Parameters

Pint ws_type  
Workstation type.

Pint string_num  
String device number (>=1).

Pstore store  
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP;*;*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind  
Error indicator.

Pint *max_buf_size  
Available input buffer size.

Pint_list **pet_list  
List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit *echo_area  
Default echo area in DC.

Pstring_data **string_data  
Default string data record. The memory referenced by *string_data is managed by the parameter store.

FORTRAN

PQDST (wtype, devno, n, mldr, errind, mbuff, ol, pet, earea, ldr, datrec)

Input Parameters

integer wtype  
Workstation type.

integer devno  
String device number (>=1).

integer n  
List element requested (>=0).

integer mldr  
Dimension of data record array (>=0).

Output Parameters

integer errind  
Error indicator.

integer mbuff  
Available string buffer size.

integer ol  
Number of available prompt and echo types.

integer pet  
nth element of the list of the available prompt and echo types.
real earea(4)
    Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    Data record.

Errors
None

Related Subroutines
• Initialize String
• Initialize String 3
• Inquire String Device State
• Inquire String Device State 3
• Inquire Default String Device Data 3
• Inquire Workstation Connection And Type

INQUIRE DEFAULT STRING DEVICE DATA 3 (PHOP,*,*,*)

Purpose

Use Inquire Default String Device Data 3 to inquire the default values of the specified string device for the specified workstation type.

The graPHIGS API returns the default values for the requested string device. For more information on default values, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_string_data3 (ws_type, string_num, store, err_ind, max_buf_size, pet_list, echo_vol, string_data)

Input Parameters

Pint ws_type
    Workstation type.

Pint string_num
    String device number (>=1).
Pstore stores
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store ([CREATE STORE (PHOP; *, *, *)]) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pint *max_buf_size
   Available input buffer size.

Pint_list **pet_list
   List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit3 *echo_vol
   Default echo volume in DC.

Pstring_data3 **string_data
   Default string data record. The memory referenced by *string_data is managed by the parameter store.

FORTRAN

PQDST3 (wtype, devno, n, mldr, errind, mbuff, ol, pet, evol, ldr, datrec)

Input Parameters

integer wtype
   Workstation type.

integer devno
   String device number (>=1).

integer n
   List element requested (>=0).

integer mldr
   Dimension of data record array (>=0).

Output Parameters

integer errind
   Error indicator.

integer mbuff
   Available string buffer size.

integer ol
   Number of available prompt and echo types.

integer pet
   n\textsuperscript{th} element of the list of the available prompt and echo types.

real evol(6)
   Default echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Number of array elements used in the data record.
Purpose

Use Inquire Default Stroke Device Data to inquire the default values of the specified stroke device for the specified workstation type.

The graPHIGS API returns the default values for the requested stroke device. For more information on defaults, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
61  Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation

Language Bindings

C

`pinq_def_stroke_data (ws_type, stroke_num, store, err_ind, max_buf_size, pet_list, echo_area, stroke_data)`

Input Parameters

`Pint ws_type`
Workstation type.

`Pint stroke_num`
Stroke device number (>=1).

`Pstore store`
Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE_STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquires.
Output Parameters

*err_ind
Error indicator.

*max_buf_size
Available input buffer size in points.

**pet_list
List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

*echo_area
Default echo area in DC.

**stroke_data
Default stroke data record. The memory referenced by *stroke_data is managed by the parameter store.

FORTRAN

PQDSK (wtype, devno, n, mldr, errind, mbuff, ol, pet, earea, ldr, datrec)

Input Parameters

integer wtype
  Workstation type.

integer devno
  Stroke device number (>=1).

integer n
  List element requested (>=0).

integer mldr
  Dimension of data record array (>=0).

Output Parameters

integer errind
  Error indicator.

integer mbuff
  Available stroke buffer size in points.

integer ol
  Number of available prompt and echo types.

integer pet
  nᵗʰ element of the list of the available prompt and echo types.

real earea(4)
  Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
  Number of array elements used in the data record.

character*80 datrec(mldr)
  Data record.

Errors

None
Related Subroutines

- Initialize Stroke
- Initialize Stroke 3
- Inquire Stroke Device State
- Inquire Stroke Device State 3
- Inquire Default Stroke Device Data 3
- Inquire Workstation Connection And Type

INQUIRE DEFAULT STROKE DEVICE DATA 3 (PHOP, *, *, *)

Purpose

Use Inquire Default Stroke Device Data 3 to inquire the default values of the specified stroke device for the specified workstation type.

The graPHIGS API returns the default values for the requested stroke device. For more information on defaults, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 2 Function Requires State (PHOP, *, *, *)
- 52 Workstation Type Not Recognized By Implementation
- 51 Information Not Available For Generic Workstation Type
- 61 Specified Workstation Is Not Of Category Input Or Output
- 250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_stroke_data3 (ws_type, stroke_num, store, err_ind, max_buf_size, pet_list, echo_vol, stroke_data)

Input Parameters

Pint ws_type
  Workstation type.

Pint stroke_num
  Stroke device number (>=1).

Pstore store
  Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP, *, *)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
  Error indicator.

Pint *max_buf_size
  Available input buffer size in points.
**Pint_list**
List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

**Plimit3**
Default echo volume in DC.

**Pstroke_data3**
Default stroke data record. The memory referenced by *stroke_data is managed by the parameter store.

FORTRAN

PQDSK3 (wtype, devno, n, mlrd, errind, mbuff, ol, pet, evol, ldr, datrec)

**Input Parameters**

*integer wtype*
Workstation type.

*integer devno*
Stroke device number (>=1).

*integer n*
List element requested (>=0).

*integer mlrd*
Dimension of data record array (>=0).

**Output Parameters**

*integer errind*
Error indicator.

*integer mbuff*
Available stroke buffer size in points.

*integer ol*
Number of available prompt and echo types.

*integer pet*
nth element of the list of the available prompt and echo types.

*real evol(6)*
Default echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

*integer ldr*
Number of array elements used in the data record.

*character*80 datrec(mldr)*
Data record.

**Errors**

None

**Related Subroutines**

- Initialize Stroke
- Initialize Stroke 3
- Inquire Stroke Device State
- Inquire Stroke Device State 3
INQUIRE DEFAULT VALUATOR DEVICE DATA (PHOP,*,* *)

Purpose

Use Inquire Default Valuator Device Data to inquire the default values of the specified valuator device on the given workstation type.

The graPHIGS API returns the default values for the requested valuator device. For more information on default values, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,* *)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_val_data (ws_type, val_num, store, err_ind, def_value, pet_list, echo_area, val_data)

Input Parameters

Pint ws_type
   Workstation type.

Pint val_num
   Valuator device number (>=1).

Pstore store
   Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,* *)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pfloat *def_value
   Default initial value.

Pint_list **pet_list
   List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit *echo_area
   Default echo area in DC.
\*\*val_data

Default valuator data record. The memory referenced by \*\*val_data is managed by the parameter store.

FORTRAN

PQDVL (wtype, devno, n, mldr, errind, dval, ol, pet, earea, ldr, datrec)

Input Parameters

integer wtype
Workstation type.

integer devno
Valuator device number (>=1).

integer n
List element requested (>=0).

integer mldr
Dimension of data record array (>=0).

Output Parameters

integer errind
Error indicator.

integer dval
Default initial value.

integer ol
Number of available prompt and echo types.

integer pet
\(n^{th}\) element of the list of the available prompt and echo types.

real earea(4)
Default echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
Number of array elements used in the data record.

character*80 datrec(mldr)
Data record.

Errors

None

Related Subroutines

- Initialize Valuator
- Initialize Valuator 3
- Inquire Valuator Device State
- Inquire Valuator Device State 3
- Inquire Default Valuator Device Data 3
- Inquire Workstation Connection And Type
INQUIRE DEFAULT VALUATOR DEVICE DATA 3 (PHOP,*,*,*)

Purpose

Use Inquire Default Valuator Device Data 3 to inquire the default values of the specified valuator device on the given workstation type.

The graPHIGS API returns the default values for the requested valuator device. For more information on default values, see The graPHIGS Programming Interface: Technical Reference.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Recognized By Generic Workstation Type
61 Specified Workstation Is Not Of Category Input Or Out
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_def_val_data3 (ws_type, val_num, store, err_ind, def_value, pet_list, echo_vol, val_data)

Input Parameters

Pint ws_type
    Workstation type.

Pint val_num
    Valuator device number (>=1).

Pstore store
    Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
    Error indicator.

Pfloat *def_value
    Default initial value.

Pint_list **pet_list
    List of prompt and echo types. The memory referenced by *pet_list is managed by the parameter store.

Plimit3 *echo_area
    Default echo volume in DC.

Pval_data3 **val_data
    Default valuator data record. The memory referenced by *val_data is managed by the parameter store.
FORTRAN

PQDVL3 (wtype, devno, n, mldr, errind, dval, ol, pet, evol, ldr, datrec)

Input Parameters

integer wtype
    Workstation type.

integer devno
    Valuator device number (>=1).

integer n
    List element requested (>=0).

integer mldr
    Dimension of data record array (>=0).

Output Parameters

integer errind
    Error indicator.

integer dval
    Default initial value.

integer ol
    Number of available prompt and echo types.

integer pet
    n\textsuperscript{th} element of the list of the available prompt and echo types.

real evol(6)
    Default echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    Data record.

Errors

None

Related Subroutines

- Initialize Valuator
- Initialize Valuator 3
- Inquire Valuator Device State
- Inquire Valuator Device State 3
- Inquire Default Valuator Device Data
- Inquire Workstation Connection And Type

INQUIRE DISPLAY SPACE SIZE (PHOP,*,*,*)

Purpose

Use Inquire Display Space Size to inquire the maximum display space size for the specified workstation type.
The graPHIGS API returns the maximum display space size in Device Coordinates (DC) and address units. Device Coordinate units are either *METER* or *OTHER*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
57  Specified Workstation Is Of Category MI
62  This Information Not Available For MO Workstation Type

**Language Bindings**

**C**

```c
pinq_disp_space_size (ws_type, err_ind, size)
```

**Input Parameters**

*Pint ws_type*

  Workstation type.

**Output Parameters**

*Pint *err_ind*

  Error indicator.

*Pdisp_space_size *size*

  Display size.

**FORTRAN**

```fortran
PQDSP (wtype, errind, dcunit, dx, dy, rx, ry)
```

**Input Parameters**

*integer wtype*

  Workstation type.

**Output Parameters**

*integer errind*

  Error indicator.

*integer dcunit*

  Device coordinate units (*0=METRE, 1=POTHU*).

*real dx*

  Maximum display space size in x direction in DC.

*real dy*

  Maximum display space size in y direction in DC.

*integer rx*

  Maximum display space size in x direction in raster units.
**integer ry**

Maximum display space size in y direction in raster units.

**Errors**

None

**Related Subroutines**

- Initialize Choice
- Initialize Locator
- Initialize Pick
- Initialize String
- Initialize Stroke
- Initialize Valuator
- Inquire Display Space Size 3
- Inquire Workstation Connection And Type

---

**INQUIRE DISPLAY SPACE SIZE 3 (PHOP,*,*,*)**

**Purpose**

Use Inquire Display Space Size 3 to inquire the maximum display space size for the specified workstation type.

The graPHIGS API returns the maximum display space size in Device Coordinates (DC) and address units. Device Coordinate units are either `METER` or `OTHER`.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
57  Specified Workstation Is Of Category MI
62  This Information Not Available For MO Workstation Type

**Language Bindings**

C

`pinq_disp_space_size3 (ws_type, err_ind, size)`

**Input Parameters**

*Pint ws_type*

Workstation type.

**Output Parameters**

*Pint *err_ind*

Error indicator.
$Pdisp\_space\_size3$ *size

Display size.

**FORTRAN**

$PQDSP3$ (wtype, errind, dcunit, dx, dy, dz, rx, ry, rz)

**Input Parameters**

*integer* wtype  
Workstation type.

**Output Parameters**

*integer* errind  
Error indicator.

*integer* dcunit  
Device coordinate units ($0=PMETRE$, $1=POTHU$).

*real* dx  
Maximum display space size in $x$ direction in DC.

*real* dy  
Maximum display space size in $y$ direction in DC.

*real* dz  
Maximum display space size in $z$ direction in DC.

*integer* rx  
Maximum display space size in $x$ direction in raster units.

*integer* ry  
Maximum display space size in $y$ direction in raster units.

*integer* rz  
Maximum display space size in $z$ direction in raster units.

**Errors**

None

**Related Subroutines**

- Initialize Choice 3
- Initialize Locator 3
- Initialize Pick 3
- Initialize String 3
- Initialize Stroke 3
- Initialize Valuator 3
- Inquire Display Space Size
- Inquire Workstation Connection And Type

**INQUIRE DISPLAY UPDATE STATE (PHOP,WSOP,* ,*)**

**Purpose**

Use Inquire Display Update State to inquire the current deferral and update state values for the specified workstation.
Possible deferral modes include: ASAP, BNIG, BNIL, ASTI, and WAIT. Possible modification modes include: NIVE, UWOR, and UQUM. The display surface is either EMPTY or NOT EMPTY. Possible states of visual representation include: CORRECT, DEFERRED, and SIMULATED. If your application specifies both simulated and deferred actions, then the state of visual representation is DEFERRED (the pixel data returned is different from the graphical state of the workstation). For an explanation of the deferral states and modification modes see The graPHIGS Programming Interface: Understanding Concepts.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP, WSOP, *, *)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_disp_upd_st (ws_id, err_ind, def_mode, mod_mode, disp_surf_empty, vis_st)

Input Parameters

Pint ws_id
  Workstation identifier.

Output Parameters

Pint *err_ind
  Error indicator.

Pdeferr_mode *def_mode
  Deferral mode (0=PDEFER_ASAP, 1=PDEFER_BNIG, 2=PDEFER_BNIL, 3=PDEFER_ASTI, 4=PDEFER_WAIT).

Pmod_mode *mod_mode
  Modification mode (0=PMODE_NIVE, 1=PMODE_UWOR, 2=PMODE_UQUM).

Pdisp_surf_empty *disp_surf_empty
  Display surface empty (0=PSURF_NOTEMPTY, 1=PSURF_EMPTY).

Pvis_st *vis_st
  State of visual representation (0=PVISUAL_ST_CORRECT, 1=PVISUAL_ST_DEFER, 2=PVISUAL_ST_SIMULATED).

FORTRAN

PQDUS (wkid, errind, defmod, modmode, dempty, stofvr)

Input Parameters

integer wkid
  Workstation identifier.

Output Parameters

integer errind
  Error indicator.
integer defmod
    Deferral mode (0=PASAP, 1=PBNIG, 2=PBNIL, 3=PASTI, 4=PWAITD).

integer modmod
    Modification mode (0=PNIVE, 1=PUWOR, 2=PUQOM).

integer dempty
    Display surface empty (0=PNEMPT, 1=PEMPTY).

integer stofvr
    State of visual representation (0=PVROK, 1=PVRDRF, 2=PVRSIM).

Errors
None

Related Subroutines
• Inquire Default Display Update State
• Set Display Update State

INQUIRE DYNAMICS OF STRUCTURES (PHOP,*,*,*)

Purpose
Use Inquire Dynamics of Structures to inquire the dynamic modification supported by the specified workstation for the following categories of picture changes:
• structure content modification
• post structure
• unpost structure
• delete structure
• reference modification (structure identifier changes).

Possible dynamic modifications include: IMPLICIT REGENERATION (IRG), meaning that implicit regeneration is necessary; IMMEDIATELY (IMM), meaning that the action is performed immediately; and CAN BE SIMULATED (CBS), meaning that the change can be simulated. If a category has a dynamic modification of IRG or CBS, it is still possible that your application can immediately execute some changes correctly if the state of the picture allows it. (For example, if no structures are posted, most changes can be executed immediately.)

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

Function Requires State (PHOP,*,*,*)
Workstation Type Not Recognized By Implementation
Information Not Available For Generic Workstation Type
Specified Workstation Does Not Have Output Capability
This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_dyns_structs (ws_type, err_ind, dyns)
Input Parameters

\textit{Pint ws\_type}

Workstation type.

Output Parameters

\textit{Pint *err\_ind}

Error indicator.

\textit{Pdyns\_structs *dyns}

Structure dynamics.

FORTRAN

\texttt{PQDSTR (wtype, errind, strcon, post, unpost, delete, refmod)}

Input Parameters

\texttt{integer wtype}

Workstation type.

Output Parameters

\texttt{integer errind}

Error indicator.

\texttt{integer strcon}

Structure content modification (0=PIRG, 1=PIMM, 2=PCBS).

\texttt{integer post}

Post structure (0=PIRG, 1=PIMM, 2=PCBS).

\texttt{integer unpost}

Unpost structure (0=PIRG, 1=PIMM, 2=PCBS).

\texttt{integer delete}

Delete structure (0=PIRG, 1=PIMM, 2=PCBS).

\texttt{integer refmod}

Reference modification (0=PIRG, 1=PIMM, 2=PCBS).

Errors

None

Related Subroutines

\begin{itemize}
  \item Inquire Dynamics Of Workstation Attributes
\end{itemize}

\textbf{INQUIRE DYNAMICS OF WORKSTATION ATTRIBUTES (PHOP\*, **, *\*)}

\textbf{Purpose}

Use Inquire Dynamics of Workstation Attributes to inquire the dynamic modification supported by the specified workstation for the following categories of picture changes:

\begin{itemize}
  \item view representation modification
  \item polyline bundle representation modification
  \item polymarker bundle representation modification
  \item text bundle representation modification
\end{itemize}
• interior bundle representation modification
• edge bundle representation modification
• pattern representation modification
• color representation modification
• workstation transformation modification
• highlighting filter modification
• invisibility filter modification
• HLHSR mode modification.

Possible dynamic modifications include: IMPLICIT REGENERATION (IRG), meaning that implicit regeneration is necessary; IMMEDIATELY (IMM), meaning that the action is performed immediately; and CAN BE SIMULATED (CBS), meaning that the change can be simulated. If a category has a dynamic modification of IRG or CBS, it is still possible that your application can immediately execute some changes correctly if the state of the picture allows it. (For example, if no structures are posted, most changes can be executed immediately.)

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2   Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
59  Specified Workstation Does Not Have Output Capability
62  This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_dyns_ws_attrs (ws_type, err_ind, attr)

Input Parameters

Pint ws_type
  Workstation type.

Output Parameters

Pint *err_ind
  Error indicator.

Pdyns_ws_attrs *attr
  Dynamics of workstation attributes.

FORTRAN

PQDSWA (wtype, errind, plbun, pmbun, txbun, inbun, edbun, parep, colrep, vwrep, wktr, hlfltr, infltr, hlhsr)

Input Parameters

integer wtype
  Workstation type.
Output Parameters

integer errind
   Error indicator.

integer plbun
   Polyl ine bundle representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer pmbun
   Polymarker bundle representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer txbun
   Text bundle representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer inbun
   Interior bundle representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer edbun
   Edge bundle representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer parep
   Pattern representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer colrep
   Color representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer vwrep
   View representation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer wktr
   Workstation transformation changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer hifltr
   Highlighting filter changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer infltr
   Invisibility filter changeable (0=PIRG, 1=PIMM, 2=PCBS).

integer hlhsr
   HLHSR mode changeable (0=PIRG, 1=PIMM, 2=PCBS).

Errors

None

Related Subroutines

* Inquire Dynamics Of Structures

INQUIRE EDGE FACILITIES (PHOP,*,*)

Purpose

Use Inquire Edge Facilities to inquire the edge facilities for the specified workstation type.

The graPHIGS API returns a number indicating the total quantity of available edge types and their identifiers; the available number of edge widths and the nominal, minimum, and maximum values; and the number of predefined edge indexes. The graPHIGS API returns the width of lines in Device Coordinate (DC) units. Possible edge types include: SOLID, DASHED, DOTTED, or DASHED-DOTTED.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:
Function Requires State (PHOP,*,*,*)
Workstation Type Not Recognized By Implementation
Information Not Available For Generic Workstation Type
Specified Workstation Does Not Have Output Capability
This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_edge_facs (ws_type, num elems appl list, start ind, err_ind, fac, num elems impl list)

Input Parameters

Pint ws_type
  Workstation type.

Pint num elems appl list
  Number of elements in the application list (>=0).

Pint index
  Starting index (>=0).

Output Parameters

Pint *err ind
  Error indicator.

Pedge_facs *fac
  Edge facilities.

Pint *num elems impl list
  Number of elements in the implementation list.

FORTRAN

PQEDF (wtype, n, errind, nedt, edt, nedw, nomedw, redwmn, npedi)

Input Parameters

integer wtype
  Workstation type.

integer n
  List element requested (>=0).

Output Parameters

integer errind
  Error indicator.

integer nedt
  Number of available edge types.

integer edt
  nth element of the list of available edge types (PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).

integer nedw
  Number of available edge widths.
real nomedw
   Nominal edge width.
real redwmn
   Minimum edge width.
real redwmx
   Maximum edge width.
integer npedi
   Number of predefined edge indexes.

Errors

None

Related Subroutines
- Inquire Edge Representation
- Set edge Representation

INQUIRE EDGE REPRESENTATION (PHOP,WSOP,*,*)

Purpose

Use Inquire Edge Representation to inquire the current attribute values in the specified entry in the edge bundle table of the specified workstation. Returned values of type SET or REALIZED may be specified.

If the specified edge index is not present in the edge bundle table on the workstation and the specified type of returned values is REALIZED, then the graPHIGS API returns the representation values for edge index 1.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability
100 Bundle Index Value Is Less Than One
101 Specified Representation Has Not Been Defined

Language Bindings

C

pinq_edge_rep (ws_id, index, type, err_ind, edge_rep)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint index
   Edge index (>=1).
Pinq_type type
   Type of returned values (0=PINOQ_SET, 1=PINOQ_REALIZED).

Output Parameters
Pint *err_ind
   Error indicator.
Pedge_bundle *edge_rep
   Edge representation.

FORTRAN
PQEDR (wkid, edi, type, errind, edflag, edtype, edwidth, coli)

Input Parameters
integer wkid
   Workstation identifier.
integer edi
   Edge index (>=1).
integer type
   Type of returned values (0=PSET, 1=PREAL).

Output Parameters
integer errind
   Error indicator.
integer edflag
   Edge flag (0=POFF, 1=PON).
integer edtype
   Edge type (1=PLSOLI, 2=PLDASH, 3=PLDOT, 4=PLDASD).
real edwidth
   Edge width scale factor.
integer coli
   Edge color index (>=0).

Errors
None

Related Subroutines
• Inquire Edge Facilities
• Set edge Representation

INQUIRE EDIT MODE (PHOP,**,**)

Purpose
Use Inquire Edit Mode to inquire the current edit mode. The edit mode is either INSERT or REPLACE.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:
Function Requires State (PHOP,*,*,*)

Language Bindings

C

\texttt{pinq\_edit\_mode (err\_ind, edit\_mode)}

Output Parameters

\textit{Pint} *\texttt{err\_ind}

Error indicator.

\textit{Pedit\_mode} *\texttt{edit\_mode}

Edit mode (0=\texttt{PEDIT\_INSERT}, 1=\texttt{PEDIT\_REPLACE}).

\textbf{FORTRAN}

\texttt{PQEDM (errind, editmo)}

Output Parameters

\textit{integer} \texttt{errind}

Error indicator.

\textit{integer} \texttt{editmo}

Edit mode (0=\texttt{PINSRT}, 1=\texttt{PREPLC}).

Errors

None

Related Subroutines

\begin{itemize}
  \item \texttt{Set Edit Mode}
\end{itemize}

\textbf{INQUIRE ELEMENT CONTENT (PHOP,*,*,*)}

\textbf{Purpose}

Use Inquire Element Content to inquire the contents of the specified structure element.

This subroutine returns the data contained in the specified structure element. The data is returned in a binding specific format. See Chapter 17. "ISO PHIGS C Type and Macro Definitions" for the C binding formats and see Chapter 16. "FORTRAN Structure Content Data Records" for the FORTRAN binding formats. Use the Inquire Element Type and Size subroutine ( \texttt{INQUIRE ELEMENT TYPE AND SIZE (PHOP,*,*,*)} ) to determine the element type and size.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

\begin{itemize}
  \item \textbf{Function Requires State (PHOP,*,*,*)}
  \item \textbf{201 Specified Structure Does Not Exist}
  \item \textbf{202 Specified Element Does Not Exist}
\end{itemize}

Language Bindings
C

pinq_elem_content (struct_id, elem_num, store, err_ind, elem_data)

Input Parameters

Pint struct_id
    Structure identifier.

Pint elem_num
    Element number (>=0).

Pstore store
    Handle to the Store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameter

Pint *err_ind
    Error indicator.

Pelem_data **elem_data
    Data record. The memory referenced by *elem_data is managed by the parameter store.

FORTRAN

PQECO (strid, elenum, iil, isl, errind, il, ia, rl, ra, sl, lstr, str)

Input Parameters

integer strid
    Structure identifier.

integer elenum
    Element position (>=0).

integer iil
    Dimension of integer array (>=0).

integer irl
    Dimension of real array (>=0).

integer isl
    Dimension of character array (>=0).

Output Parameters

integer errind
    Error indicator.

integer il
    Number of integer entries.

integer ia (iil)
    Array containing integer entries.

integer rl
    Number of real entries.

real ra (irl)
    Array containing real entries.
integer sl  
    Number of character string entries.

integer lstr (isl)  
    Length of each character string entry.

character*(*) str(isl)  
    Character string entries.

FORTRAN Subset

PQECO (strid, elenum, iil, irl, isl, errind, il, ia, ra, rl, sl, lstr, str)

Input Parameters

integer strid  
    Structure identifier.

integer elenum  
    Element position (>=0).

integer iil  
    Dimension of integer array (>=0).

integer irl  
    Dimension of real array (>=0).

integer isl  
    Dimension of character array (>=0).

Output Parameters

integer errind  
    Error indicator.

integer il  
    Number of integer entries.

integer ia (iil)  
    Array containing integer entries.

integer rl  
    Number of real entries.

real ra (irl)  
    Array containing real entries.

integer sl  
    Number of character string entries.

integer lstr (isl)  
    Length of each character string entry.

character*80 str(isl)  
    Character string entries.

Errors

None

Related Subroutines

- Inquire Element Type And Size
INQUIRE ELEMENT POINTER (PHOP,*,STOP,*)

Purpose

Use Inquire Element Pointer to inquire the value of the current element pointer in the structure store.

A structure must be open to invoke this subroutine.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to the following error:

5      Function Requires State (PHOP,*,STOP,*)

Language Bindings

C

pinq_elem_ptr (err_ind, elem_ptr_value)

Output Parameters

Pint *err_ind
  Error indicator.

Pint *elem_ptr_value
  Element pointer value.

FORTRAN

PQEP (errind, ep)

Output Parameters

integer errind
  Error indicator.

integer ep
  Element pointer value.

Errors

None

Related Subroutines

• None

INQUIRE ELEMENT TYPE AND SIZE (PHOP,,*,*)

Purpose

Use Inquire Element Type and Size to inquire the type and size of the specified structure element.

This subroutine returns the element type of the specified element. If elements of this type have no associated values, then the graPHIGS API returns a value of zero in the element size parameter. If the element pointer is currently zero, then the graPHIGS API returns a value of NIL in the element type parameter. To retrieve the element contents, use the Inquire Element Content subroutine.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP.*,.*,*)
201 Specified Structure Does Not Exist
202 Specified Element Does Not Exist

Language Bindings

C

pinqElemTypeSize (struct_id, elem_num, err_ind, elem_type, elem_size)

Input Parameters

Pint struct_id
Structure identifier.

Pint elem_num
Element number (>=0).

Output Parameters

Pint *err_ind
Error indicator.

Pelem_type *elem_type
Element type (1=PELEM_NIL, 2=PELEM_POLYLINE3, 3=PELEM_POLYLINE, 4=PELEM_POLYMARKER3, 5=PELEM_POLYMARKER, 6=PELEM_TEXT3, 7=PELEM_TEXT, 8=PELEM_ANNO_TEXT_REL3, 9=PELEM_ANNO_TEXT_REL, 10=PELEM_FILL_AREA3, 11=PELEM_FILL_AREA, 12=PELEM_FILL_AREA_SET3, 13=PELEM_FILL_AREA_SET, 14=PELEM_CELL_ARRAY3, 15=PELEM_CELL_ARRAY, 16=PELEM_GDP3, 17=PELEM_GDP, 18=PELEM_LINE_IND, 19=PELEM_MARKER_IND, 20=PELEM_TEXT_IND, 21=PELEM_INT_IND, 22=PELEM_EDGE_IND, 23=PELEM_LINETYPE, 24=PELEM_LINEWIDTH, 25=PELEM_LINECOLR_IND, 26=PELEM_MARKER_TYPE, 27=PELEM_MARKER_SIZE, 28=PELEM_MARKERCOLR_IND, 29=PELEM_TEXT_FONT, 30=PELEM_TEXT_PRC, 31=PELEM_CHAR_EXPAN, 32=PELEM_CHAR_SPACE, 33=PELEM_TEXTCOLR_IND, 34=PELEM_CHARHT, 35=PELEM_CHARUP_VEC, 36=PELEM_TEXT_PATH, 37=PELEM_TEXT_ALIGN, 38=PELEM_ANNOCHAR_HT, 39=PELEM_ANNOCHAR_UP_VEC, 40=PELEM_ANNO_PATH, 41=PELEM_ANNOALIGN, 42=PELEM_ANNO_STYLE, 43=PELEM_INT_STYLE, 44=PELEM_INT_STYLE_IND, 45=PELEM_INTCOLR_IND, 46=PELEM_EDG_FLAG, 47=PELEM_EDGETYPE, 48=PELEM_EDGWIDTH, 49=PELEM_EDGCOLR_IND, 50=PELEM_PA TIMESTAMP, 51=PELEM_PA TIMESTAMP, 52=PELEM_PA_TIMESTAMP, 53=PELEM_ADD_NAMES_SET, 54=PELEM_REMOVE_NAMES_SET, 55=PELEM_INDIVASF, 56=PELEM_HLHSR_ID, 57=PELEM_LOCALMODEL_TRAN3, 58=PELEM_LOCALMODEL_TRAN, 59=PELEM_GLOBALMODEL_TRAN3, 60=PELEM_GLOBALMODEL_TRAN, 61=PELEM_MODEL_CLIP_VOL3, 62=PELEM_MODEL_CLIP_VOL, 63=PELEM_MODEL_CLIPIND, 64=PELEM_RESTOREMODEL_CLIP_VOL, 65=PELEM_VIEW_IND, 66=PELEM_EXECSUB, 67=PELEM_LABEL, 68=PELEM_APPLDATA, 69=PELEM_GSE, 70=PELEM_PICK_ID).

size_t *elem_size
Element size.
FORTRAN

PQETS (strid, elenum, errind, eltype, il, rl, sl)

Input Parameters

integer strid
  Structure identifier.

integer elenum
  Element position (>=0).

Output Parameters

integer errind
  Error indicator.

integer eltype
  Element type (1=PENIL, 2=PEPL3, 3=PEPL, 4=PEPM3, 5=PEPM, 6=PETX3, 7=PETX, 8=PEATR3, 9=PEATR, 10=PEFA3, 11=PEFA, 12=PEFAS3, 13=PEFAS, 14=PECA3, 15=PECA, 16=PEGDP3, 17=PEGDP, 18=PEPLI, 19=PEPLI, 20=PETXI, 21=PEII, 22=PEEDI, 23=PELPN, 24=PELWSC, 25=PEPLCI, 26=PEMK, 27=PEMKSC, 28=PEPMCI, 29=PETXFN, 30=PETXPR, 31=PECHXP, 32=PECHSP, 33=PETXCI, 34=PECHH, 35=PECHUP, 36=PETXP, 37=PETXAL, 38=PEATCH, 39=PEATCU, 40=PEATP, 41=PEATL, 42=PEANST, 43=PEIS, 44=PEISI, 45=PEICI, 46=PEEDFG, 47=PEEDT, 48=PEEWSC, 49=PEEDCI, 50=PEPA, 51=PEPRPV, 52=PEPARF, 53=PEADS, 54=PERES, 55=PEIASF, 56=PEHRID, 57=PELMT3, 58=PELMT, 59=PEGMT3, 60=PEGMT, 61=PEMCV3, 62=PEMCV, 63=PEMCLI, 64=PEMCL, 65=PEVWI, 66=PEEXST, 67=PELB, 68=PEAT, 69=PEGSE, 70=PEPKID).

integer il
  Dimension of integer array.

integer rl
  Dimension of real array.

integer sl
  Dimension of character array.

Errors

None

Related Subroutines

• Inquire Element Content

INQUIRE ERROR HANDLING MODE (PHOP,*,*,*)

Purpose

Use Inquire Error Handling Mode to inquire if the error handling mode is set to either ON or OFF.

To set the mode, use the Set Error Handling Mode subroutine The default error handling mode is ON.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the value in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to the following error:

2  Function Requires State (PHOP,*,*,*)
Language Bindings

C

pinq_err_hand_mode (err_ind, err_mode)

Output Parameters

Pint *err_ind
   Error indicator.

Perr_mode *err_mode
   Error handling mode (0=PERR_OFF, 1=PERR_ON).

FORTRAN

PQERHM (errind, erhm)

Output Parameters

integer errind
   Error indicator.

integer erhm
   Error handling mode (0=POFF, 1=PON).

Errors

None

Related Subroutines

• Set Error Handling Mode

INQUIRE GENERALIZED DRAWING PRIMITIVE (PHOP,*,*,*)

Purpose

Use Inquire Generalized Drawing Primitive to inquire the list of sets of attributes used by the specified Generalized Drawing Primitive (GDP) on the specified workstation. Possible sets of attributes include: POLYLINE, POLYMARKER, TEXT, INTERIOR and/or EDGE attributes.

The graPHIGS API returns a list of the attributes used by the specified GDP. For registered GDP identifiers, the ISO International Register of Graphical Items defines the list of sets of attributes used. For implementation dependent GDP identifiers, the list of sets of attributes used is workstation dependent.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2   Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
59  Specified Workstation Does Not Have Output Capability
64  Specified Workstation Type Cannot Generate Specified GDP
Language Bindings

C

pinq_gdp (ws_type, gdp, err_ind, num_attr, attr)

Input Parameters

Pint ws_type
   Workstation type.

Pint gdp
   GDP function identifier.

Output Parameters

Pint *err_ind
   Error indicator.

Pint *num_attr
   Number of sets of attributes used.

Pattrs attr[5]
   List of sets of attributes used (0=PATTR_LINE, 1=PATTR_MARKER, 2=PATTR_TEXT, 3=PATTR_INT, 4=PATTR_EDGE).

FORTRAN

PQGDP (wtype, gdp, errind, nbnd, bndl)

Input Parameters

integer wtype
   Workstation type.

integer gdp
   GDP function identifier.

Output Parameters

integer errind
   Error indicator.

integer nbnd
   Number of sets of attributes used.

integer bndl(5)
   List of sets of attributes used (0=PPLATT, 1=PPMATT, 2=PTXATT, 3=PINATT, 4=PEDATT).

Errors

None

Related Subroutines

- Generalized Drawing Primitive
INQUIRE GENERALIZED DRAWING PRIMITIVE 3 (PHOP,*,*,*)

Purpose

Use Inquire Generalized Drawing Primitive 3 to inquire the list of sets of attributes used by the specified Generalized Drawing Primitive 3 (GDP 3) on the specified workstation. Possible sets of attributes include: POLYLINE, POLYMARKER, TEXT, INTERIOR and/or EDGE attributes.

The graPHIGS API returns a list of the attributes used by the specified GDP 3. For registered GDP 3 identifiers, the ISO International Register of Graphical Items defines the list of sets of attributes used. For implementation dependent GDP 3 identifiers, the list of sets of attributes used is workstation dependent.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2    Function Requires State (PHOP,*,*,*)
52   Workstation Type Not Recognized By Implementation
51   Information Not Available For Generic Workstation Type
59   Specified Workstation Does Not Have Output Capability
64   Specified Workstation Type Cannot Generate Specified GDP
62   This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_gdp3 (ws_type, gdp, err_ind, num_attr, attr)

Input Parameters

Pint ws_type
   Workstation type.

Pint gdp
   GDP 3 function identifier.

Output Parameters

Pint *err_ind
   Error indicator.

Pint *num_attr
   Number of sets of attributes used.

Pattrs attr[5]
   List of sets of attributes used (0=PATTR_LINE, 1=PATTR_MARKER, 2=PATTR_TEXTURE, 3=PATTR_INT, 4=PATTR_EDGE).

FORTRAN

PQGDP3 (wtype, gdp, errind, nbnd, bndl)

Input Parameters
integer wttype  
    Workstation type.

integer gdp  
    GDP 3 function identifier.

Output Parameters

integer errind  
    Error indicator.

integer nbnd  
    Number of sets of attributes used.

integer bndl(5)  
    List of sets of attributes used (0=PPLATT, 1=PPMATT, 2=PTXATT, 3=PINATT, 4=PEDATT).

Errors

None

Related Subroutines

• Generalized Drawing Primitive 3

INQUIRE GENERALIZED STRUCTURE ELEMENT FACILITIES
(PHOP,**,**)

Purpose

Use Inquire Generalized Structure Element Facilities to inquire the list of Generalized Structure Element (GSE) identifiers which are supported on the specified workstation. For each GSE identifier, the graPHIGS API returns a workstation dependent indicator which specifies if that particular GSE has actions which are workstation independent or workstation dependent.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,**,**)

Language Bindings

C

pinq_gse_facs (num elems appl list, start ind, err ind, gse, num elems impl list)

Input Parameters

Pint num elems appl list  
    Number of elements in the application list (>=0).

Pint start ind  
    Starting index (>=0).

Output Parameters

Pint *err ind  
    Error indicator.
**Pgse_id_dep_list** `*gse`
List of GSE function identifiers and dependencies.

**Pint** `*num elems_impl_list`
Number of elements in the implementation list.

**FORTRAN**

`PQGSEF (n, errind, ol, gseid, wsdind)`

**Input Parameters**

`integer n`
Element requested from the list of GSEs (>=0).

**Output Parameters**

`integer errind`
Error indicator.

`integer ol`
Number of available GSEs.

`integer gseid`
GSE function identifier of the \( n^{th} \) element in the list of available GSEs.

`integer wsdind`
Workstation dependency indicator of the \( n^{th} \) element in the list of available GSEs (\( 0=PWKI, 1=PWKD \)).

**Errors**

None

**Related Subroutines**

- Generalized Structure Element

**INQUIRE HIGHLIGHTING FILTER (PHOP,WSOP,*,*)**

**Purpose**

Use Inquire Highlighting Filter to inquire the current highlighting inclusion and exclusion filters on the specified workstation.

The graPHIGS API returns the inclusion filter list and the exclusion filter list.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 3 Function Requires State (PHOP,WSOP,*,*)
- 54 Specified Workstation Is Not Open
- 59 Specified Workstation Does Not Have Output Capability

**Language Bindings**
C

\texttt{pinq\_highl\_filter (ws\_id, store, err\_ind, highl\_filter)}

\textbf{Input Parameters}

\textit{Pint ws\_id}

Workstation identifier.

\textit{Pstore store}

Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (\texttt{CREATE\_STORE (PHOP,\*,\*,\*)}) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

\textbf{Output Parameters}

\textit{Pint *err\_ind}

Error indicator.

\textit{Pfilter **highl\_filter}

Highlighting filter. The memory referenced by \texttt{*highl\_filter} is managed by the parameter \texttt{store}.

\textbf{FORTRAN}

\texttt{PQHLFT (wkid, isbsz, esbsz, errind, isn, is, esn, es)}

\textbf{Input Parameters}

\textit{integer wkid}

Workstation identifier.

\textit{integer isbsz}

Inclusion set buffer size (\texttt{>=0}).

\textit{integer esbsz}

Exclusion set buffer size (\texttt{>=0}).

\textbf{Output Parameters}

\textit{integer errind}

Error indicator.

\textit{integer isn}

Number of names in the inclusion set.

\textit{integer is (isbsz)}

Inclusion set.

\textit{integer esn}

Number of names in the exclusion set.

\textit{integer es (esbsz)}

Exclusion set.

\textbf{Errors}

None

\textbf{Related Subroutines}

- Set Highlighting Filter
INQUIRE HLHSR IDENTIFIER FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire HLHSR Identifier Facilities to inquire the list of available Hidden Line/Hidden Surface Removal (HLHSR) identifiers on the specified workstation. See Set HLHSR Identifier subroutine for a listing of the possible HLHSR identifiers.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
57 Specified Workstation Is Of Category MI
62 This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_hlhsr_id_facs (ws_type, num elems appl list, start ind, err ind, hlhsr ids, num elems impl list)

Input Parameters

Pint ws_type
  Workstation type.

Pint num elems appl list
  Number of elements in the application list (>=0).

Pint start ind
  Starting index (>=0).

Output Parameters

Pint *err ind
  Error indicator.

Pint_list *hlhsr ids
  List of available HLHSR identifiers.

Pint *num elems impl list
  Number of elements in the implementation list.

FORTRAN

PQHRIF (wtype, ni, err ind, nhrid, hrid)

Input Parameters

integer wtype
  Workstation type.
integer ni
   List element requested (>=0).

Output Parameters
integer errind
   Error indicator.

integer nhrid
   Number of available HLHSR identifiers.

integer hrid
   Ni\textsuperscript{th} element in the list of available HLHSR identifiers.

Errors
None

Related Subroutines
\begin{itemize}
   \item Inquire HLHSR Mode Facilities
   \item Set HLHSR Identifier
   \item Set HLHSR Mode
\end{itemize}

INQUIRE HLHSR MODE (PHOP,WSOP,*,*)

Purpose

Use Inquire HLHSR Mode to inquire the current and requested Hidden Line/Hidden Surface Removal (HLHSR) mode, and the HLHSR update state on the specified workstation. Possible HLHSR modes include: 0=OFF and 1=ON THE FLY.

The HLHSR update state is \textit{PENDING} if the application has requested an HLHSR mode change but the workstation has not yet provided that change. Otherwise, the update state is \textit{NOT PENDING}.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

\begin{itemize}
   \item 3 Function Requires State (PHOP,WSOP,*,*)
   \item 54 Specified Workstation Is Not Open
   \item 57 Specified Workstation Is Of Category MI
\end{itemize}

Language Bindings

C

\texttt{pinq_hlhsr_mode (ws\_id, err\_ind, upd\_st, cur\_mode, req\_mode)}

Input Parameters

\texttt{Pint ws\_id}
   Workstation identifier.

Output Parameters

\texttt{Pint *err\_ind}
   Error indicator.
Pupd_st *upd_st
    HLHSR update state (0=PUPD_NOT_PEND, 1=PUPD_PEND).

Pint *cur_mode
    Current HLHSR mode.

Pint *req_mode
    Requested HLHSR mode.

FORTRAN

PQHRM (wkid, errind, hupd, chrm, rhrm)

Input Parameters
integer wkid
    Workstation identifier.

Output Parameters
integer errind
    Error indicator.

integer hupd
    HLHSR mode update state (0=PNPEND, 1=PPEND).

integer chrm
    Current HLHSR mode.

integer rhrm
    Requested HLHSR mode.

Errors
None

Related Subroutines
• Inquire HLHSR Mode Facilities
• Set HLHSR Mode

INQUIRE HLHSR MODE FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire HLHSR Mode Facilities to inquire the list of available Hidden Line/Hidden Surface Removal (HLHSR) modes on the specified workstation. Possible HLHSR modes include: 0=OFF and 1=ON THE FLY.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
57 Specified Workstation Is Of Category MI
62 This Information Not Available For MO Workstation Type
Language Bindings

C
pinq_hlhsr_mode_facs (ws_type, num elems appl list, start ind, err ind, hlhsr modes, num elems impl list)

Input Parameters
Pint ws_type
   Workstation type.
Pint num elems appl list
   Number of elements in the application list (>=0).
Pint start_ind
   Starting index (>=0).

Output Parameters
Pint *err ind
   Error indicator.
Pint_list *hlhsr_modes
   List of available HLHSR modes.
Pint *num elems impl list
   Number of elements in the implementation list.

FORTRAN
PQHRMF (wtype, nm, errind, nhrmd, hrmd)

Input Parameters
integer wtype
   Workstation type.
integer nm
   List element requested (>=0).

Output Parameters
integer errind
   Error indicator.
integer nhrmd
   Number of available HLHSR modes.
integer hrmd
   NMth element in the list of available HLHSR modes.

Errors
None

Related Subroutines
• Inquire HLHSR Mode Facilities
• Set HLHSR Identifier
• Set HLHSR Mode
INQUIRE INPUT QUEUE OVERFLOW (PHOP,WSOP,*,*)

Purpose

Use Inquire Input Queue Overflow to inquire identification of the event report causing the event queue overflow.

Once the event queue overflow occurs, the graPHIGS API will not add more events to the event queue until the application clears the overflow situation by emptying the event queue. The application can make the event queue empty by using the Await Event or Flush Device Events subroutines.

If the event queue has overflowed since Open PHIGS or the last invocation of this subroutine, then the graPHIGS API returns the identification of the logical input device that caused the overflow. Logical input device classes include: LOCATOR, STROKE, VALUATOR, CHOICE, PICK, and STRING.

The graPHIGS API does not report the event queue overflow to the application when the overflow occurs. It is reported on the next invocation of the following subroutines which may change the contents of the event queue:
• Await Event
• Flush Device Events
• Close Workstation

The graPHIGS API reports the event queue overflow to the application only once per event queue overflow situation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:
3 Function Requires State (PHOP,WSOP,*,*)
257 Input Queue Has Not Overflowed
258 Input Queue Has Overflowed, But Workstation Is Closed

Language Bindings

C

\texttt{pinq\_in\_overf (err\_ind, ws\_id, in\_class, in\_num)}

Output Parameters

\begin{itemize}
\item \texttt{Pint *err\_ind}  
  Error indicator.
\item \texttt{Pint *ws\_id}  
  Workstation identifier.
\item \texttt{Pin\_class *in\_class}  
  Input class (1=\texttt{PIN\_LOC}, 2=\texttt{PIN\_STROKE}, 3=\texttt{PIN\_VAL}, 4=\texttt{PIN\_CHOICE}, 5=\texttt{PIN\_PICK}, 6=\texttt{PIN\_STRING}).
\item \texttt{Pint *in\_num}  
  Input device number.
\end{itemize}
FORTRAN

PQIQOV (errind, wkid, icl, idn)

Output Parameters

integer errind
Error indicator.

integer wkid
Workstation identifier.

integer icl
Input class (1=PLOCAT, 2=PSTROK, 3=PVALUA, 4=PCHOIC, 5=PPICK, 6=PSTRIN).

integer idn
Input device number.

Errors

None

Related Subroutines

• None

INQUIRE INTERIOR FACILITIES (PHOP,***,*)

Purpose

Use Inquire Interior Facilities to inquire the interior facilities for the specified workstation type.

The graPHIGS API returns data indicating the total number of available interior styles, the number of available hatch styles, and the total number of indexes predefined in the interior bundle table. Possible interior styles include: HOLLOW, SOLID, PATTERN, HATCH, and EMPTY.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,***,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_int_facs (ws_type, hatch_num elems appl_list, hatch_start_ind, err_ind, int_facs, hatch_num elems_impl_list)

Input Parameters

Pint ws_type
Workstation type.
Pint hatch_num_elems_appl_list
   Number of elements in the application hatch style list (>=0).

Pint hatch_start_ind
   Starting index (>=0).

Output Parameters

Pint *err_ind
   Error indicator.

Pint_facs *int_facs
   Interior facilities.

Pint *hatch_num_elems_impl_list
   Number of elements in the implementation list.

FORTRAN

PQIF (wtype, ni, nh, errind, nis, is, nhs, hs, npfai)

Input Parameters

integer wtype
   Workstation type.

integer ni
   List element of interior styles requested (>=0).

integer nh
   List element of hatch styles requested (>=0).

Output Parameters

integer errind
   Error indicator.

integer nis
   Number of available interior styles.

integer is
   $N^{th}$ element in the list of available interior styles (0=PHOLLO, 1=PSOLID, 2=PPATTR, 3=PHATCH, 4=PISEMP).

integer nhs
   Number of available hatch styles.

integer hs
   $NH^{th}$ element in list of available hatch style indexes.

integer npfai
   Number of predefined interior indexes.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type
Use Inquire Interior Representation to inquire the current attribute values in the specified entry in the interior bundle table for the specified workstation. Your application may specify returned values of either type SET or REALIZED. Possible interior styles include: HOLLOW, SOLID, PATTERN, HATCH, and EMPTY.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
59    Specified Workstation Does Not Have Output Capability
100   Bundle Index Value Is Less Than One
101   Specified Representation Has Not Been Defined

Language Bindings

C

pinq_int_rep (ws_id, index, type, err_ind, int_rep)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint index
   Interior index (>=1).

Pint_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Output Parameters

Pint *err_ind
   Error indicator.

Pint_bundle *int_rep
   Interior representation. (See Chapter 17. “ISO PHIGS C Type and Macro Definitions” for the type definitions).

FORTRAN

PQIR (wkid, ii, type, errind, ints, istyli, col)

Input Parameters

integer wkid
   Workstation identifier.

integer ii
   Interior index (>=1).
integer type
Type of returned values (0=PSET, 1=PREAL).

Output Parameters

type errind
Error indicator.
inits
Interior style (0=PHOLLO, 1=PSOLID, 2=PPATTR, 3=PHATCH, 4=PISEMP).
istsli
Interior style index.
coli
Interior color index.

Errors
None

Related Subroutines
• Set Interior Representation

INQUIRE INVISIBILITY FILTER (PHOP,WSOP,*,*)

Purpose

Use Inquire Invisibility Filter to inquire the current invisibility inclusion and exclusion filters on the specified workstation.

The graPHIGS API returns the inclusion filter list and the exclusion filter list.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_invis_filter (ws_id, store, err_ind, invis_filter)

Input Parameters

Pint ws_id
Workstation identifier.

Pstore store
Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters
Pint *err_ind
    Error indicator.

Pfilter **invis_filter
    Invisibility filter. The memory referenced by *invis_filter is managed by the parameter store.

FORTRAN

PQIVFT (wkid, isbsz, esbsz, errind, isn, is, esn, es)

Input Parameters

integer wkid
    Workstation identifier.

integer isbsz
    Inclusion set buffer size (>=0).

integer esbsz
    Exclusion set buffer size (>=0).

Output Parameters

integer errind
    Error indicator.

integer isn
    Number of names in the inclusion set.

integer is (isbsz)
    Inclusion set.

integer esn
    Number of names in the exclusion set.

integer es (esbsz)
    Exclusion set.

Errors

None

Related Subroutines

- Set Invisibility Filter
- Inquire PHIGS Facilities

INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (PHOP,*,*,*)

Purpose

Use Inquire List of Available Generalized Drawing Primitives to inquire the available Generalized Drawing Primitives (GDPs) for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type

Language Bindings

C

\texttt{ping\_list\_avail\_gdp (ws\_type, num\_elems\_appl\_list, start\_ind, err\_ind, gdps, num\_elems\_impl\_list)}

Input Parameters

\textit{Pint ws\_type}  
Workstation type.

\textit{Pint num\_elems\_appl\_list}  
Number of elements in the application list (\(\geq 0\)).

\textit{Pint start\_ind}  
Starting index (\(\geq 0\)).

Output Parameters

\textit{Pint *err\_ind}  
Error indicator.

\textit{Pint\_list gdps}  
List of GDPs.

\textit{Pint *num\_elems\_impl\_list}  
Number of elements in the implementation list.

FORTRAN

\texttt{PQEGDP (wtype, n, errind, ngdp, gdpl)}

Input Parameters

\textit{integer wtype}  
Workstation type.

\textit{integer n}  
List element requested (\(\geq 0\)).

Output Parameters

\textit{integer err\_ind}  
Error indicator.

\textit{integer ngdp}  
Number of available GDPs.

\textit{integer gdpl}  
\(n^{th}\) element in the list of GDP identifiers.

Errors

None
Related Subroutines

- Inquire Workstation Connection And Type

---

**INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES 3 (PHOP,*,*,*)**

**Purpose**

Use Inquire List of Available Generalized Drawing Primitives 3 to inquire the available Generalized Drawing Primitives 3 (GDP 3s) for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- Function Requires State (PHOP,*,*,*)
- Workstation Type Not Recognized By Implementation
- Information Not Recognized By Generic Workstation Type
- Specified Workstation Does Not Have Output Capability
- This Information Not Available For MO Workstation Type

**Language Bindings**

**C**

```c
pinq_list_avail_gdp3 (ws_type, num elems appl list, start ind, err ind, gdps, num elems impl list)
```

**Input Parameters**

- **Pint ws_type**
  - Workstation type.
- **Pint num elems appl list**
  - Number of elements in the application list (>=0).
- **Pint start ind**
  - Starting index (>=0).

**Output Parameters**

- **Pint *err ind**
  - Error indicator.
- **Pint list gdps**
  - List of available 3D GDPs.
- **Pint *num elems impl list**
  - Number of elements in the implementation list.

**FORTRAN**

```fortran
PQEGD3 (wtype, n, errind, ngdp, gdp)
```

**Input Parameters**

...
integer wtype
    Workstation type.

integer n
    List element requested (>=0).

Output Parameters

integer errind
    Error indicator.

integer ngdp
    Number of available 3D GDPs.

integer gdpl
    n\textsuperscript{th} element in the list of 3D GDP identifiers.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE LIST OF AVAILABLE GENERALIZED STRUCTURE ELEMENTS
(\texttt{PHOP,*,*,*})

Purpose

Use Inquire List of Available Generalized Structure Elements to inquire a list of available Generalized Structure Element (GSE) identifiers for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Function Requires State (PHOP,<em>,</em>,*)</td>
</tr>
<tr>
<td>52</td>
<td>Workstation Type Not Recognized By Implementation</td>
</tr>
<tr>
<td>51</td>
<td>Information Not Available For Generic Workstation Type</td>
</tr>
<tr>
<td>59</td>
<td>Specified Workstation Does Not Have Output Capability</td>
</tr>
<tr>
<td>62</td>
<td>This Information Not Available For MO Workstation Type</td>
</tr>
</tbody>
</table>

Language Bindings

C

\texttt{pinq_list_avail_gse} (\texttt{ws\_type, num\_elems\_appl\_list, start\_ind, err\_ind, gses, num\_elems\_impl\_list})

Input Parameters

\texttt{Pint} \texttt{ws\_type}
    Workstation type.

\texttt{Pint} \texttt{num\_elems\_appl\_list}
    Number of elements in the application list (>=0).
Pint start_ind
Starting index (>=0).

Output Parameters
Pint *err_ind
Error indicator.

Pint_list *gses
List of available GSEs.

Pint *num elems_impl_list
Number of elements in the implementation list.

FORTRAN
PQEGSE (wtype, n, errind, ngse, gsel)

Input Parameters
integer wtype
Workstation type.

integer n
List element requested (>=0).

Output Parameters
integer errind
Error indicator.

integer ngse
Number of available GSEs.

integer gsel
nth element in the list of available GSE identifiers.

Errors
None

Related Subroutines
• Inquire Workstation Connection And Type

INQUIRE LIST OF AVAILABLE WORKSTATION TYPES (PHOP, *, *, *)

Purpose
Use Inquire List of Available Workstation Types to inquire a list of available generic workstation types that can be opened. The workstation types returned are those which your application may use as a workstation type parameter to the Open Workstation subroutine.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP, *, *, *)

Language Bindings
C

pinq_list_avail_ws_types (numelems_appl_list, start_ind, err_ind, types, numelems_impl_list)

Input Parameters

Pint numelems_appl_list
  Number of elements in the application list (>=0).

Pint start_ind
  Starting index (>=0).

Output Parameters

Pint *err_ind
  Error indicator.

Pint_list *types
  List of available workstation types.

Pint *numelems_impl_list
  Number of elements in the implementation list.

FORTRAN

PQEWK (n, errind, number, wktyp)

Input Parameters

integer n
  List element requested (>=0).

Output Parameters

integer errind
  Error indicator.

integer number
  Number of workstation types.

integer wktyp
  nth element in the list of available workstation types.

Errors

None

Related Subroutines

• Open Workstation

INQUIRE LIST OF COLOR INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of Color Indices to inquire the list of color indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:
3 Function Requires State (PHOP,WSOP,*,*)

54 Specified Workstation Is Not Open

59 Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_list_colr_inds (ws_id, num elems appl_list, start_ind, err_ind, colr_ind, num elems impl_list)

Input Parameters

Pint ws_id
 Workstation identifier.

Pint num elems appl_list
 Number of elements in the application list (>=0).

Pint start_ind
 Starting index (>=0).

Output Parameters

Pint *err_ind
 Error indicator.

Pint_list *colr_ind
 List of color indexes.

Pint *num elems impl_list
 Number of elements in the implementation list.

FORTRAN

PQECI (wkid, n, errind, ol, coli)

Input Parameters

integer wkid
 Workstation identifier.

integer n
 List element requested (>=0).

Output Parameters

integer errind
 Error indicator.

integer ol
 Number of color table entries.

integer coli
 $n^{th}$ element in the list of color indexes.

Errors

None
Related Subroutines
• Set Color Representation

INQUIRE LIST OF EDGE INDICES (PHOP, WSOP, *, *)

Purpose

Use Inquire List of Edge Indices to inquire the list of defined edge indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP, WSOP, *, *)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_list_edge_inds (ws_id, numelems_appl_list, start_ind, err_ind, def_edge_ind, numelems_impl_list)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint numelems_appl_list
   Number of elements in the application list (>=0).

Pint start_ind
   Starting index (>=0).

Output Parameters

Pint *err_ind
   Error indicator.

Pint_list *def_edge_ind
   List of defined edge indexes.

Pint *numelems_impl_list
   Number of elements in the implementation list.

FORTRAN

PQEEEDI (wkid, n, errind, ol, edi)

Input Parameters

integer wkid
   Workstation identifier.

integer n
   List element requested (>=0).

Output Parameters
integer errind
   Error indicator.

integer ol
   Number of edge bundle table entries.

integer edi
   \( r^{th} \) element in the list of defined edge indexes.

Errors

None

Related Subroutines

- Set edge Representation

INQUIRE LIST OF INTERIOR INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of Interior Indices to inquire the list of defined interior indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 3   Function Requires State (PHOP,WSOP,*,*)
- 54  Specified Workstation Is Not Open
- 59  Specified Workstation Does Not Have Output Capability

Language Bindings

C

\texttt{pinq\_list\_int\_inds} (ws\_id, num\_elems\_appl\_list, start\_ind, \texttt{err\_ind}, def\_int\_ind, num\_elems\_impl\_list)

Input Parameters

\texttt{Pint ws\_id}
   Workstation identifier.

\texttt{Pint num\_elems\_appl\_list}
   Number of elements in the application list (\(>=0\)).

\texttt{Pint start\_ind}
   Starting index (\(>=0\)).

Output Parameters

\texttt{Pint \*err\_ind}
   Error indicator.

\texttt{Pint\_list \*def\_int\_ind}
   List of defined interior indexes.

\texttt{Pint \*num\_elems\_impl\_list}
   Number of elements in the implementation list.
FORTRAN

PQEII (wkid, n, errind, ol, ii)

Input Parameters

integer wkid
   Workstation identifier.

integer n
   List element requested (>=0).

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of interior bundle table entries.

integer ii
   \( n^{th} \) element in the list of defined interior indexes.

Errors

None

Related Subroutines

- Set Interior Representation

INQUIRE LIST OF PATTERN INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of Pattern Indices to inquire the list of defined pattern indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_list_pat_inds (ws_id, num_elems_appl_list, start_ind, err_ind, def_pat_ind, num_elems_impl_list)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint num_elems_appl_list
   Number of elements in the application list (>=0).
Pint start_ind
    Starting index (>=0).

Output Parameters

Pint *err_ind
    Error indicator.

Pint_list *def_pat_ind
    List of defined pattern indexes.

Pint *num elems_impl_list
    Number of elements in the implementation list.

FORTRAN

PQEPAI (wkid, n, errind, ol, pai)

Input Parameters

integer wkid
    Workstation identifier.

integer n
    List element requested (>=0)

Input Parameters

integer errind
    Error indicator.

integer ol
    Number of pattern table entries.

integer pai
    nth element in the list of pattern indexes.

Errors

None

Related Subroutines

• Set Pattern Representation

INQUIRE LIST OF POLYLINE INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of Polyline Indices to inquire the list of defined polyline indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3  Function Requires State (PHOP,WSOP,*,*)
54  Specified Workstation Is Not Open
59  Specified Workstation Does Not Have Output Capability
Language Bindings

C

pinq_list_line inds (ws_id, num elems appl list, start ind, err ind, def line ind, num elems impl list)

Input Parameters

Pint ws_id
  Workstation identifier.

Pint num elems appl list
  Number of elements in the application list (>=0).

Pint start ind
  Starting index (>=0).

Output Parameters

Pint *err ind
  Error indicator.

Pint list *def line ind
  List of defined polyline indexes.

Pint *num elems impl list
  Number of elements in the implementation list.

FORTRAN

PQEPLI (wkid, n, errind, ol, pli)

Input Parameters

integer wkid
  Workstation identifier.

integer n
  List element requested (>=0).

Output Parameters

integer errind
  Error indicator.

integer ol
  Number of polyline bundle table entries.

integer pli
  nth element in the list of defined polyline indexes.

Errors

None

Related Subroutines
  • Set Polyline Representation
INQUIRE LIST OF POLYMARKER INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of Polymarker Indices to inquire the list of defined polymarker indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3    Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
59   Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_list_marker_inds (ws_id, num_elems_appl_list, start_ind, err_ind, def_marker_ind, num_elems_impl_list)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint num_elems_appl_list
    Number of elements in the application list (>=0).

Pint start_ind
    Starting index (>=0).

Output Parameters

Pint *err_ind
    Error indicator.

Pint_list *def_marker_ind
    List of defined polymarker indexes.

Pint *num_elems_impl_list
    Number of elements in the implementation list.

FORTRAN

PQEPMI (wkid, n, errind, ol, pmi)

Input Parameters

integer wkid
    Workstation identifier.

integer n
    List element requested (>=0).

Output Parameters

integer errind
    Error indicator.
integer ol
Number of polymarker bundle table entries.

integer pmi
\(n^{th}\) element in the list of defined polymarker indexes.

Errors
None

Related Subroutines
• Set Polymarker Representation

---

INQUIRE LIST OF TEXT INDICES (PHOP,WSOP,\(^*,*\))

Purpose
Use Inquire List of Text Indices to inquire the list of defined text indexes existing on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,\(^*,*\))
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_list_text_indsws_id, num elems appl list, start ind, err ind, def text ind, num elems impl list

Input Parameters

Pint ws_id
Workstation identifier.

Pint num elems appl list
Number of elements in the application list (\(\geq 0\)).

Pint start ind
Starting index (\(\geq 0\)).

Output Parameters

Pint *err ind
Error indicator.

Pint_list *def text ind
List of defined text indexes.

Pint *num elems impl list
Number of elements in the implementation list.

FORTRAN

PQETXI (wkid, n, errind, ol, tx)
Input Parameters

integer wkid
    Workstation identifier.

integer n
    List element requested (>=0).

Output Parameters

integer errind
    Error indicator.

integer ol
    Number of text bundle table entries.

integer txi
    \( n^{th} \) element in the list of defined text indexes.

Errors

None

Related Subroutines

• Set Text Representation

INQUIRE LIST OF VIEW INDICES (PHOP,WSOP,*,*)

Purpose

Use Inquire List of View Indices to inquire the list of defined view indexes existing on the workstation.

The graPHIGS API returns the view indexes in a list, which is ordered by view transformation input priority, starting with the highest priority view.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
57 Specified Workstation Is Of Category MI

Language Bindings

C

pinq_list_view_inds (ws_id, numelems_appl_list, start_ind, err_ind, view inds, numelems_impl_list)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint num elems_appl_list
    Number of elements in the application list (>=0).
**Pint start_ind**
Starting index (>=0).

**Output Parameters**

**Pint err_ind**
Error indicator.

**Pint_list view_inds**
List of defined view indexes.

**Pint num_elems_impl_list**
Number of elements in the implementation list.

**FORTRAN**

**PQEVWI** (wkid, n, errind, nvwix, viewi)

**Input Parameters**

**integer wkid**
Workstation identifier.

**integer n**
List element requested (>=0).

**Output Parameters**

**integer errind**
Error indicator.

**integer nvwix**
Number of view bundle table entries.

**integer viewi**
\( n^{th} \) element in the list of defined view indexes.

**Errors**

None

**Related Subroutines**

• Set View Representation

---

**INQUIRE LOCATOR DEVICE STATE (PHOP,WSOP,*,*)**

**Purpose**

Use Inquire Locator Device State to inquire the current state of the specified locator device on the specified workstation.

Returned values of type *SET* or *REALIZED* may be specified.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)

54 Specified Workstation Is Not Open
Specified Workstation Is Not Of Category Input Or Outin

Specified Device Not Available On Workstation

Language Bindings

C

pinq_loc_st (ws_id, loc_num, type, store, err_ind, op_mode, echo_switch, init_view_ind, init_loc_pos, prompt_echo, echo_area, loc_data)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint loc_num
   Locator device number (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pop_mode *op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pint *init_view_ind
   Initial view index.

Ppoint *init_loc_pos
   Initial locator position in WC.

Pint *prompt_echo
   Prompt and echo type.

Plimit *echo_area
   Echo area in DC.

Ploc_data **loc_data
   Data record. The memory referenced by *loc_data is managed by the parameter store.

FORTRAN

PQLCS (wkid, lcdnr, type, mldr, errind, mode, esw, iviewi, ipx, ipy, pet, earea, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.
integer lcdnr
   Locator device number (>=1).

integer type
   Type of returned values (0=PSET, 1=PREAL).

integer mldr
   Dimension of data record array (>=0).

Output Parameters

integer errind
   Error indicator.

integer mode
   Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

integer iviewi
   Initial view index.

real ipx
   x coordinate of the initial locator position in WC.

real ipy
   y coordinate of the initial locator position in WC.

integer pet
   Prompt and echo type.

real earea(4)
   Echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
   Number of array elements used in the data record.

integer character*80 datrec(mldr)
   Data record.

Errors

None

Related Subroutines
- Initialize Locator
- Initialize Locator 3
- Inquire Default Locator Device Data
- Inquire Default Locator Device Data 3
- Inquire Locator Device State 3

INQUIRE LOCATOR DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose

Use Inquire Locator Device State 3 to inquire the current state of the specified locator device on the specified workstation. Returned values of type SET or REALIZED may be specified.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

`pinq_loc_st3 (ws_id, loc_num, type, store, err_ind, op_mode, echo_switch, init_view_ind, init_loc_pos, prompt_echo, echo_vol, loc_data)`

Input Parameters

Pint ws_id
   Workstation identifier.

Pint loc_num
   Locator device number (>=1).

Pint type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store ([CREATE STORE (PHOP,*,*,*)](#)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pop_mode *op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pint *init_view_ind
   Initial view index.

Ppoint3 *init_loc_pos
   Initial locator position in WC.

Pint *prompt_echo
   Prompt and echo type.

Plimit3 *echo_vol
   Echo volume in DC.

Ploc_data3 **loc_data
   Data record. The memory referenced by *loc_data is managed by the parameter store.
FORTRAN

PQLCS3 (wkid, lcdnr, type, mldr, errind, mode, esw, iviewi, ipx, ipy, ipz, pet, evol, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.

integer lcdnr
   Locator device number (>=1).

integer type
   Type of returned values (0=PSET, 1=PREALI).

integer mldr
   Dimension of data record array (>=0).

Output Parameters

integer errind
   Error indicator.

integer mode
   Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

integer iviewi
   Initial view index.

real ipx
   x coordinate of the initial locator position in WC.

real ipy
   y coordinate of the initial locator position in WC.

real ipz
   z coordinate of the initial locator position in WC.

integer pet
   Prompt and echo type.

real evol(6)
   Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
   Number of array elements used in the data record.

integer character*80 datrec(mldr)
   Data record.

Errors

None

Related Subroutines

• Initialize Locator
• Initialize Locator 3
• Inquire Default Locator Device Data
• Inquire Default Locator Device Data 3
• Inquire Locator Device State

INQUIRE MODELING CLIPPING FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire Modeling Clipping Facilities to inquire the number and list of modeling clipping operators, and the number of distinct planes in the modeling clipping volume.

Although this inquiry returns two operations (REPLACE and INTERSECT) and six as the number of distinct planes in the modeling clipping volume, support differs by workstation. Therefore, use the graPHIGS API Inquire Workstation Description (GPQWDT) to inquire these values for a specific workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the values returned in the output parameters are invalid and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pinq_model_clip_facs (num elems appl list, start ind, err ind, num planes, ops, num elems impl list)

Input Parameters

Pint num elems appl list
  Number of elements in the application list (>=0).

Pint start ind
  Starting index (>=0).

Output Parameters

Pint *err ind
  Error indicator.

Pint *num planes
  Number of distinct half planes in the modeling clipping volume.

Pint_list *ops
  List of modeling clipping operators (1=REPLACE, 2=INTERSECT).

Pint *num elems impl list
  Number of elements in the implementation list.

FORTRAN

PQMCLF (n, errind, ndpmcv, ol, mclpop)

Input Parameters

integer n
  Element list requested (>=0).

Output Parameters
integer errind  
   Error indicator.

integer ndpmcv  
   Number of distinct half planes in the modeling clipping volume.

integer ol  
   Number of modeling clipping operators

integer mclpop  
   n\textsuperscript{th} element in the list of modeling clipping operators. (1=REPLACE, 2=INTERSECT).

Errors  
None  

Related Subroutines  
• Set Modeling Clipping Indicator  
• Set Modeling Clipping Volume  
• Set Modeling Clipping Volume 3

---

**INQUIRE MORE SIMULTANEOUS EVENTS (PHOP,\*,\*,\*)**

**Purpose**

Use Inquire More Simultaneous Events to inquire whether additional simultaneous events are waiting in the input queue.

The graPHIGS API returns a value indicating whether additional events are waiting that occurred from the same device trigger as the event previously in the current event report (CEV).

Your application can call this subroutine after your application issued the appropriate “Get” subroutine to obtain the current event from the CEV.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the value in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,\*,\*,\*)

**Language Bindings**

**C**

```c
pinq_more_simult_events (err_ind, simult_events)
```

**Output Parameters**

\textit{P}int \texttt{err\_ind}
   Error indicator.

\textit{P}more\_simult\_events \texttt{simult\_events}
   More simultaneous events (0=PSIMULT\_NO\_MORE, 1=PSIMULT\_MORE).

**FORTRAN**

```fortran
PQSIM (errind, flag)
```
Output Parameters

**integer errind**
   Error indicator.

**integer flag**
   More simultaneous events (0=PNMORE, 1=PMORE).

Errors

None

Related Subroutines

- None

---

**INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (PHOP,*,*,*)**

**Purpose**

Use Inquire Number of Available Logical Input Devices to inquire the number of available logical input devices at the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Function Requires State (PHOP,<em>,</em>,*)</td>
</tr>
<tr>
<td>52</td>
<td>Workstation Type Not Recognized By Implementation</td>
</tr>
<tr>
<td>51</td>
<td>Information Not Available For Generic Workstation Type</td>
</tr>
<tr>
<td>61</td>
<td>Specified Workstation Is Not Of Category Input Or Outin</td>
</tr>
</tbody>
</table>

**Language Bindings**

**C**

```c
pinq_num_avail_in (ws_type, err_ind, num_in)
```

**Input Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pint ws_type</td>
<td>Workstation type.</td>
</tr>
</tbody>
</table>

**Input Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pint *err_ind</td>
<td>Error indicator.</td>
</tr>
<tr>
<td>Pnum_in *num_in</td>
<td>Number of input devices.</td>
</tr>
</tbody>
</table>

**FORTRAN**

```fortran
PQLI (wtype, errind, nlcd, nskd, nvld, nchd, npkd, nstd)
```

**Input Parameters**

360 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
integer wtype
   Workstation type.

Output Parameters

integer errind
   Error indicator.

integer nlcd
   Number of locator devices.

integer nskd
   Number of stroke devices.

integer nvld
   Number of valuator devices.

integer nchd
   Number of choice devices.

integer npkd
   Number of pick devices.

integer nstd
   Number of string devices.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE NUMBER OF DISPLAY PRIORITIES SUPPORTED (PHOP,*,*,*)

Purpose

Use Inquire Number of Display Priorities Supported to inquire the number of display priorities supported for the specified workstation type.

When posting a structure to a workstation, the application specifies a structure priority, which is a real number between 0.0 and 1.0. The graPHIGS API traverses the structures in order, from lowest to highest priority.

The graPHIGS API returns values indicating the total number of supported display priorities. For example, if a workstation uses a 4-bit mask to keep track of priorities, then it is able to support only 16 different priorities and must map the real number to one of 16 values. If a workstation can support a continuous range of display priorities, then the inquiry returns a zero.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
59  Specified Workstation Does Not Have Output Capability
62  This Information Not Available For MO Workstation Type
Language Bindings

C

pinq_num_disp_pris (ws_type, err_ind, num_pri)

Input Parameters

Pint ws_type
   Workstation type.

Output Parameters

Pint *err_ind
   Error indicator.

Pint *num_pri
   Number of display priorities supported.

FORTRAN

PQDP (wtype, errind, nspsup)

Input Parameters

integer wtype
   Workstation type.

Output Parameters

integer errind
   Error indicator.

integer nspsup
   Number of display priorities supported.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type
• Post Structure

INQUIRE OPEN STRUCTURE (PHOP,**,**)

Purpose

Use Inquire Open Structure to inquire the identifier of the open structure.

If your application has opened a structure, then the open structure status is OPEN and the graPHIGS API returns the identifier of the open structure as a structure identifier. If your application has not opened a structure, then the graPHIGS API returns the open structure status as NONE and the structure identifier is undefined.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pinq_open_struct (err_ind, status, struct_id)

Output Parameters

Pint *err_ind
   Error indicator.

Popen_struct_status *status
   Open structure status (0=PSTRUCT_NONE, 1=PSTRUCT_OPEN).

Pint *struct_id
   Structure identifier.

FORTRAN

PQOPST (errind, stype, strid)

Output Parameters

integer errind
   Error indicator.

integer stype
   Open structure status (0=PNONST, 1=POPNST).

integer strid
   Structure identifier.

Errors

None

Related Subroutines

• Open Structure
• Close Structure

INQUIRE PATHS TO ANCESTORS (PHOP,*,*,*)

Purpose

Use Inquire Paths to Ancestors to inquire the ancestral paths of a specified structure. A path of ancestors of a structure S is a list of ordered pairs: \( ((A1,E1),(A2,E2),\ldots,(Am,Em), (S,0)) \) where each ordered pair consists of an identifier of a structure (Ax) that is an ancestor of the specified structure (S) and the position of an execute structure-type element (Ex) that references the next structure in the path. Ancestor structure A1 is the top of the path (i.e., it is not referenced by any other structure) and S is the bottom of the path.
The path order and path depth determine the portion of each path to be returned. Your application may specify the path order as *TOP FIRST* or *BOTTOM FIRST*. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation may result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all of the returned path portions are distinct.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2       Function Requires State (PHOP,*,*,*)
201     Specified Structure Does Not Exist
207     Specified Path Depth < Zero

Language Bindings

C

pinq_paths_ances (struct_id, order, depth, store, err_ind, paths)

Input Parameters

Pint struct_id
   Structure identifier.

Ppath_order order
   Path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

Pint depth
   Path depth (>=0).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store ([CREATE STORE (PHOP,*,*,*)]) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pelem_ref_list_list **paths
   Structure path list. The memory referenced by *paths is managed by the parameter store.

FORTRAN

PQPAN (strid, pthord, pthdep, ipthsz, n, errind, ol, apthsz, paths)

Input Parameters

integer strid
   Structure identifier.

integer pthord
   Path order (0=PPOTOP, 1=POBOT).
integer pthdep
   Path depth (>=0).

integer ipthsz
   Maximum number of path entries the buffer can contain.

integer n
   Element of the list of paths.

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of paths available.

integer apthsz
   Actual number of entries of the nth structure path returned.

integer paths(2,ipthsz)
   nth structure path.

Errors
None

Related Subroutines
- Inquire Paths To Descendants

INQUIRE PATHS TO DESCENDANTS (PHOP,*,*,*)

Purpose

Use Inquire Paths To Descendants to inquire the descendant paths of a specified structure. A path of descendants of a structure S is a list of ordered pairs: \((S,E_0),(D_1,E_1),(D_2,E_2),..., (D_n,0)\), where each ordered pair consists of an identifier of a structure \((D_x)\) that is a descendant of the specified structure \((S)\) and the position of an execute structure-type element \((E_x)\) that references the next structure in the path. The specified structure \(S\) is the top of the path and descendant structure \(D_n\) is the bottom of the path (i.e., it does not reference any other structure).

The path order and path depth determine the portion of each path to be returned. Your application may specify the path order as TOP FIRST or BOTTOM FIRST. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation may result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all of the returned path portions are distinct.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
201 Specified Structure Does Not Exist
207 Specified Path Depth < Zero

Language Bindings
C

pinq_paths_descs (struct_id, order, depth, store, err_ind, paths)

Input Parameters

Pint struct_id
   Structure identifier.

Path_order order
   Path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

Pint depth
   Path depth (>=0).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pelem_ref_list_list **paths
   Structure path list. The memory referenced by *paths is managed by the parameter store.

FORTRAN

PQPDE (strid, pthord, pthdep, ipthsz, n, errind, ol, apthsz, paths)

Input Parameters

integer strid
   Structure identifier.

integer pthord
   Path order (0=PPOTOP, 1=PPOBOT).

integer pthdep
   Path depth (>=0).

integer ipthsz
   Maximum number of path entries the buffer can contain.

integer n
   Element of the list of paths.

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of paths available.

integer apthsz
   Actual number of entries of the $n^{th}$ structure path returned.

integer paths(2,ipthsz)
   $n^{th}$ structure path.
Errors

None

Related Subroutines

- Inquire Paths To Ancestors

**INQUIRE PATTERN FACILITIES (PHOP,*,*,*)**

**Purpose**

Use Inquire Pattern Facilities to inquire the pattern facilities for the specified workstation type.

The graPHIGS API returns the number of predefined pattern indexes for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2    Function Requires State (PHOP,*,*,*)
52   Workstation Type Not Recognized By Implementation
51   Information Not Available For Generic Workstation Type
59   Specified Workstation Does Not Have Output Capability
62   This Information Not Available For MO Workstation Type

**Language Bindings**

**C**

\texttt{pinq\_pat\_fac} (\texttt{ws\_type, err\_ind, num\_pred})

**Input Parameters**

\texttt{Pint ws\_type}

Workstation type.

**Output Parameters**

\texttt{Pint *err\_ind}

Error indicator.

\texttt{Pint *num\_pred}

Number of predefined pattern indexes.

**FORTRAN**

\texttt{PQPAF (wtype, errind, nppai)}

**Input Parameters**

\texttt{integer wtype}

Workstation type.

**Output Parameters**
**integer errind**
Error indicator.

**integer nppai**
Number of predefined pattern indexes.

**Errors**
None

**Related Subroutines**
- Inquire Workstation Connection And Type

---

**INQUIRE PATTERN REPRESENTATION (PHOP,WSOP,*,*)**

**Purpose**
Use Inquire Pattern Representation to inquire the current pattern representation in the specified entry in the pattern table of the specified workstation. Returned values of type SET or REALIZED may be specified.

If the specified pattern index is not present in the pattern table on the workstation and the specified type of returned values is REALIZED, then the graPHIGS API returns the representation for pattern index 1. Pattern index 1 is present if your workstation supports interior style PATTERN.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **3** Function Requires State (PHOP,WSOP,*,*)
- **54** Specified Workstation Is Not Open
- **59** Specified Workstation Does Not Have Output Capability
- **112** Pattern Index Value < ONE
- **101** Specified Representation Has Not Been Defined
- **109** Interior Style Pattern Not Supported On Workstation

**Language Bindings**

**C**

```c
pinq_pat_rep (ws_id, index, type, store, err_ind, pat_rep)
```

**Input Parameters**

- **Pint ws_id**
  Workstation identifier.

- **Pint index**
  Pattern index (>=1).

- **Pint type**
  Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

- **Pstore store**
  Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of...
subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,**)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

\[ Pint \ *\text{err\_ind} \]
Error indicator.

\[ P\text{pat\_rep} \ **\text{pat\_rep} \]
Pattern representation. The memory referenced by \text{pat\_rep} is managed by the parameter \text{store}.

FORTRAN

```
PQPAR (wkid, pai, type, dimx, dimy, errind, dx, dy, colia)
```

Input Parameters

\[ \text{integer wkid} \]
Workstation identifier.

\[ \text{integer pai} \]
Pattern index (>=1).

\[ \text{integer type} \]
Type of returned values (0=PSET, 1=PREAL).

\[ \text{integer dimx} \]
Maximum column dimension in pattern array (>=0).

\[ \text{integer dimy} \]
Maximum row dimension in pattern array (>=0).

Output Parameters

\[ \text{integer errind} \]
Error indicator.

\[ \text{integer dx} \]
Number of columns in pattern color index array.

\[ \text{integer dy} \]
Number of rows in pattern color index array.

\[ \text{integer colia}(\text{dimx},\text{dimy}) \]
Pattern color index array.

Errors

None

Related Subroutines

• Set Pattern Representation

**INQUIRE PHIGS FACILITIES (PHOP,*,*,*)**

Purpose

Use Inquire PHIGS Facilities to inquire the graPHIGS API facilities.

The graPHIGS API returns the maximum number of simultaneously open workstations, the maximum number of simultaneously open archive files, the number of available names for name sets, the list of...
available character sets, the maximum length of a normal filter list for Incremental Spatial Search (ISS), and the maximum length of an inverted filter list for ISS.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

**pinq_phigs_facs** (num_elems_appl_list, start_ind, err_ind, max_open_ws, max_open_ar, num_avail_names, char_sets, num_elems_impl_list, iss_norm_max, iss_inv_max)

Input Parameters

*Pint num_elems_appl_list*
Number of elements in the application list (>=0).

*Pint start_ind*
Starting index (>=0).

Output Parameters

*Pint *err_ind*
The error indicator.

*Pint *max_open_ws*
Maximum number of simultaneously opened workstations.

*Pint *max_open_ar*
Maximum number of simultaneously opened archive files.

*Pint *num_avail_names*
Number of available names for name sets.

*Pint_list *char_sets*
List of available character sets.

*Pint *num_elems_impl_list*
Number of elements in the implementation list.

*Pint *iss_norm_max*
Maximum length of the normal filter list for Incremental Spacial Search (ISS).

*Pint *iss_inv_max*
Maximum length of the inverted filter list for Incremental Spacial Search (ISS).

FORTRAN

**pqphf** (ncs, errind, simopw, simopa, namesn olcs, cs, nfln, ifln)

Input Parameters

*integer ncs*
Character set requested.

Output Parameters
integer errind
   Error indicator.

integer simopw
   Maximum number of simultaneously opened workstations.

integer simopa
   Maximum number of simultaneously opened archive files.

integer namesn
   Maximum number of available names for name sets.

integer olcs
   Number of available character sets.

integer cs
   NCS\textsuperscript{th} available character set.

integer nfln
   Maximum length of the normal filter list for Incremental Spacial Search (ISS).

integer ifln
   Maximum length of the inverted filter list for Incremental Spacial Search (ISS).

Errors

None

Related Subroutines

- None

\textbf{INQUIRE PICK DEVICE STATE (PHOP,WSOP,*,*)}

Purpose

Use Inquire Pick Device State to inquire the current state of the specified pick device on the specified workstation. Returned values of type \textit{SET} or \textit{REALIZED} may be specified.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

\begin{itemize}
   \item 3 \quad \text{Function Requires State (PHOP,WSOP,*,*)}
   \item 54 \quad \text{Specified Workstation Is Not Open}
   \item 60 \quad \text{Specified Workstation Is Not Of Category Outin}
   \item 250 \quad \text{Specified Device Not Available On Workstation}
\end{itemize}

Language Bindings

C

\texttt{pinq\_pick\_st (ws\_id, pick\_num, type, store, err\_ind, op\_mode, echo\_switch, pick\_filter, init\_status, init\_pick, prompt\_echo, echo\_area, pick\_data, order)}

Input Parameters

\textit{Pint ws\_id}
   Workstation identifier.
Pint pick_num
   Pick device number (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE_STORE (PHOP,*,* *)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pop_mode *op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pfilter **pick_filter
   Pick filter. The memory referenced by *pick_filter is managed by the parameter store.

Pin_status *init_status
   Initial pick status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Ppick_path **init_pick
   Initial pick path. The memory referenced by *init_pick is managed by the parameter store.

Pint *prompt_echo
   Prompt and echo type.

Plimit *echo_area
   Echo area in DC.

Ppick_data3 **pick_data
   Data record. The memory referenced by *pick_data is managed by the parameter store.

Path_order *order
   Pick path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

FORTRAN

PQPKS (wkid, pkdnr, type, mldr, ipissz, ipessz, ippsz, errind, mode, esw, pissz, pins, pessz, pes, istat, pdd, pp, pet, earea, ldr, datrec, ppordr)

Input Parameters

integer wkid
   Workstation identifier.

integer pkdnr
   Pick device number (>=1).

integer type
   Type of returned values (0=PSET, 1=PREALI).

integer mldr
   Dimension of data record array (>=0).

integer ipissz
   Pick inclusion set buffer size (>=0).
integer ipessz
Pick exclusion set buffer size (>=0).

integer ippsz
Pick path buffer size (>=0).

Output Parameters

integer errind
Error indicator.

integer mode
Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
Echo switch (0=PNECHO, 1=PECHO).

integer pissz
Pick inclusion set size.

integer pins(ipissz)
Pick inclusion set.

integer pessz
Pick exclusion set size.

integer pes(ipessz)
Pick exclusion set.

integer istat
Initial pick status (1=POK, 2=PNPICK).

integer ppd
Initial pick path depth.

integer pp(3,ippsz)
Initial pick path.

integer pet
Prompt and echo type.

real earea(4)
Echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
Number of array elements used in the data record.

character*80 datrec(mldr)
Data record.

integer ppordr
Pick path order (0=PPOTOP, 1=PPOBOT).

Errors

None

Related Subroutines

• Initialize Pick
• Initialize Pick 3
• Inquire Default Pick Device Data
• Inquire Default Pick Device Data 3
• Inquire Pick Device State 3
INQUIRE PICK DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose

Use Inquire Pick Device State 3 to inquire the current state of the specified pick device on the specified workstation. Returned values of type SET or REALIZED may be specified.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3  Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
60 Specified Workstation Is Not Of Category Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_pick_st3 (ws_id, pick_num, type, store, err_ind, op_mode, echo_switch, pick_filter, init_status, init_pick, prompt_echo, echo_vol, pick_data, order)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint pick_num
   Pick device number (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE_STORE (PHOP,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.

Pop_mode *op_mode
   Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
   Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pfilter **pick_filter
   Pick filter. The memory referenced by *pick_filter is managed by the parameter store.

Pin_status *init_status
   Initial pick status (1=PIN_STATUS_OK, 2=PIN_STATUS_NO_IN).

Ppick_path **init_pick
   Initial pick path. The memory referenced by *init_pick is managed by the parameter store.
Pint *prompt_echo
  Prompt and echo type.

Plimit3 *echo_vol
  Echo volume in DC.

Ppick_data3 **pick_data
  Data record. The memory referenced by *pick_data is managed by the parameter store.

Ppath_order *order
  Pick path order (0=PORDER_TOP_FIRST, 1=PORDER_BOTTOM_FIRST).

FORTRAN

PQPKS3 (wkid, pkdnr, type, mldr, ipissz, ipessz, errind, mode, esw, pissz, pins, pessz, pes, istat, ppd, pp, pet, evol, ldr, datrec, ppordr)

Input Parameters

integer wkid
  Workstation identifier.

integer pkdnr
  Pick device number (>=1).

integer type
  Type of returned values (0=PSET, 1=PREAL).

integer mldr
  Dimension of data record array (>=0).

integer ipissz
  Pick inclusion set buffer size (>=0).

integer ipessz
  Pick exclusion set buffer size (>=0).

integer ippsz
  Pick path buffer size (>=0).

Output Parameters

integer errind
  Error indicator.

integer mode
  Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
  Echo switch (0=PNECHO, 1=PECHO).

integer pissz
  Pick inclusion set size.

integer pins(ipissz)
  Pick inclusion set.

integer pessz
  Pick exclusion set size.

integer pes(ipessz)
  Pick exclusion set.
integer istat
    Initial status (1=POK, 2=PNPICK).

integer ppd
    Initial pick path depth.

integer pp(3,iippsz)
    Initial pick path.

integer pet
    Prompt and echo type.

integer evol(6)
    Echo volume in DC. (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    Data record.

integer ppordr
    Pick path order (0=PPOTOP, 1=PPOBOT).

Errors

None

Related Subroutines
- Initialize Pick
- Initialize Pick 3
- Inquire Default Pick Device Data
- Inquire Default Pick Device Data 3
- Inquire Pick Device State

INQUIRE POLYLINE FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire Polyline Facilities to inquire the polyline facilities for the specified workstation type.

The graPHIGS API returns the total number of available line types and their identifiers; the number of available line widths and the nominal, minimum, and maximum line width size; and the number of predefined polyline bundle table indexes for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2    Function Requires State (PHOP,*,*,*)
52   Workstation Type Not Recognized By Implementation
51   Information Not Available For Generic Workstation Type
59   Specified Workstation Does Not Have Output Capability
62   This Information Not Available For MO Workstation Type

Language Bindings
C

pinq_line_facs (ws_type, num elems appl_list, start_ind, err_ind, fac, num elems impl_list)

Input Parameters

Pint ws_type
    Workstation type.

Pint num elems appl_list
    Number of elements in the application list (>=0).

Pint start_ind
    Starting index (>=0).

Output Parameters

Pint *err_ind
    Error indicator.

Pline_facs *fac
    Polyline facilities.

Pint *num elems impl_list
    Number of elements in the implementation list.

FORTRAN

PQPLF (wtype, n, errind, nlt, nlw, nomlw, rlwmin, rlwmax, nppli)

Input Parameters

integer wtype
    Workstation type.

integer n
    List element requested (>=0).

Output Parameters

integer errind
    Error indicator.

integer nlt
    Number of available line types.

integer lt
    n\textsuperscript{th} element in the list of available line types.

integer nlw
    Number of available line widths.

real nomlw
    Nominal line width in DC.

real rlwmin
    Minimum value of a line width in DC.

real rlwmax
    Maximum value of a line width in DC.

integer nppli
    Number of predefined polyline indexes.
Errors

None

Related Subroutines

- Inquire Workstation Connection And Type

**INQUIRE POLYLINE REPRESENTATION (PHOP, WSOP, *, *)**

**Purpose**

Use Inquire Polyline Representation to inquire the current polyline representation in the specified entry in the polyline bundle table of the specified workstation. Returned values of type *SET* or *REALIZED* may be specified.

If the specified polyline index is not present in the polyline bundle table on the workstation and the specified type of returned values is *REALIZED*, then the graPHIGS API returns the representation for polyline index 1.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3         Function Requires State (PHOP, WSOP, *, *)
54        Specified Workstation Is Not Open
59        Specified Workstation Does Not Have Output Capability
100       Bundle Index Value Is Less Than One
101       Specified Representation Has Not Been Defined

**Language Bindings**

C

```c
pinq_line_rep (ws_id, index, type, errind, line_rep)
```

**Input Parameters**

- `Pint ws_id`
  - Workstation identifier.
- `Pint index`
  - Polyline index (>=1).
- `Pinq_type type`
  - Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

**Output Parameters**

- `Pint *errind`
  - Error indicator.
- `Pline_bundle *line_rep`
  - Polyline representation.
FORTRAN

PQPLR (wkid, pli, type, errind, ltype, lwidth, coli)

Input Parameters

integer wkid
    Workstation identifier.

integer pli
    Polyline index (>=1).

integer type
    Type of returned values (0=PSET, 1=PREAL).

Output Parameters

integer errind
    Error indicator.

integer ltype
    Line type.

real lwidth
    Line width scale factor.

integer coli
    Polyline color index.

Errors

None

Related Subroutines

• Set Polyline Representation

INQUIRE POLYMARKER FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire Polymarker Facilities to inquire the polymarker facilities for the specified workstation type.

The graPHIGS API returns data indicating the total number of available marker types and their identifiers; the nominal, minimum, and maximum marker sizes; and the number of predefined polymarker bundle table indexes for the specified workstation type. If the graPHIGS API returns a value of zero for the number of available marker sizes, then the workstation supports a continuous range of marker sizes.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2          Function Requires State (PHOP,*,* ,*)
52         Workstation Type Not Recognized By Implementation
51         Information Not Available For Generic Workstation Type
59         Specified Workstation Does Not Have Output Capability
62         This Information Not Available For MO Workstation Type
Language Bindings

C

pinq_marker_facs (ws_type, num elems appl list, start ind, err ind, fac, num elems impl list)

Input Parameters

Pint ws_type
  Workstation type.

Pint num elems appl list
  Number of elements in the application list (>=0).

Pint start ind
  Starting index (>=0).

Output Parameters

Pint *err ind
  Error indicator.

Pmarker_facs *fac
  Polymarker facilities.

Pint num elems impl list
  Number of elements in the implementation list.

FORTRAN

PQPMF (wtype, n, errind, nmt, mt, nms, nomms, rmsmin, rmsmax, nppmi)

Input Parameters

integer wtype
  Workstation type.

integer n
  List element requested (>=0).

Output Parameters

integer errind
  Error indicator.

integer nmt
  Number of available marker types.

integer mt
  n\textsuperscript{th} element in the list of available marker types.

integer nms
  Number of available marker widths.

real nomms
  Nominal marker width in DC.

real rmsmin
  Minimum value of a marker width in DC.

real rmsmax
  Maximum value of a marker width in DC.
integer nppmi
   Number of predefined polymarker indexes.

Errors

None

Related Subroutines
   • Inquire Workstation Connection And Type

INQUIRE POLYMARKER REPRESENTATION (PHOP,WSOP,*,*)

Purpose

Use Inquire Polymarker Representation to inquire the current polymarker representation in the specified entry in the polymarker bundle table of the specified workstation. Returned values of type SET or REALIZED may be specified.

If the specified polymarker index is not present in the polyline bundle table on the workstation and the specified type of returned values is REALIZED, then the graPHIGS API returns the representation for polymarker index 1.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3    Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
59    Specified Workstation Does Not Have Output Capability
100   Bundle Index Value Is Less Than One
101   Specified Representation Has Not Been Defined

Language Bindings

C

pinq_marker_rep (ws_id, index, type, *err_ind, *marker_rep)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint index
   Polymarker index (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Output Parameters

Pint *err_ind
   Error indicator.

Pmarker_bundle *marker_rep
   Polymarker representation.
FORTRAN

PQPMR (wkid, pmi, type, errind, mtype, mszsf, coli)

Input Parameters

integer wkid
   Workstation identifier.

integer pmi
   Polymarker index (>=1).

integer type
   Type of returned values (0=PSET, 1=PREAL).

Output Parameters

integer errind
   Error indicator.

integer mtype
   Marker type.

real mszsf
   Marker size scale factor.

integer coli
   Polymarker color index.

Errors

None

Related Subroutines

- Set Polymarker Representation

INQUIRE POSTED STRUCTURES (PHOP,WSOP,*;*)

Purpose

Use Inquire Posted Structures to inquire the structure networks which have been identified for display on the specified workstation by the Post Structure subroutine.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3    Function Requires State (PHOP,WSOP,*;*)
54   Specified Workstation Is Not Open
59   Specified Workstation Does Not Have Output Capability

Language Bindings

C

pinq_posted_structs (ws_id, num elems appl_list, start_ind, err_ind, struct_ids, num elems impl_list)
Input Parameters

\( Pint\ ws\_id\)
Workstation identifier.

\( Pint\ num\_elems\_appl\_list\)
Number of elements in the application list (\( \geq 0 \)).

\( Pint\ start\_ind\)
Starting index (\( \geq 0 \)).

Output Parameters

\( Pint\ *err\_ind\)
Error indicator.

\( Pposted\_struct\_list\ *struct\_ids\)
List of structures posted to the workstation.

\( Pint\ *num\_elems\_impl\_list\)
Number of elements in the implementation list.

FORTRAN

\( PQPOST\ (wkid, n, errind, number, strid, prior)\)

Input Parameters

\texttt{integer wkid}
Workstation identifier.

\texttt{integer n}
List element requested (\( \geq 0 \)).

Output Parameters

\texttt{integer errind}
Error indicator.

\texttt{integer number}
Number of structures posted to the workstation.

\texttt{integer strid}
Identifier of the \( n^{\text{th}} \) structure posted to the workstation.

\texttt{real prior}
Input priority of the \( n^{\text{th}} \) structure posted to the workstation.

Errors

None

Related Subroutines

\begin{itemize}
  \item Post Structure
  \item Unpost All Structures
  \item Unpost Structure
\end{itemize}

\textbf{INQUIRE PREDEFINED COLOR REPRESENTATION (PHOP*,**,*)}

Purpose
Use Inquire Predefined Color Representation to inquire the color values in the predefined color table entry in the color table of the specified workstation. The color specification parameters are the coordinates of the color in the default color model as defined in the workstation description table (WDT).

The graPHIGS API returns the predefined color specification corresponding to the specified index.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP;*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type
113 Color Index Value < ZERO
102 Representation Has Not Been Predefined On This Workstation

Language Bindings

C

cinq_pred_colr_rep (ws_type, colr_ind, err_ind, colr_rep)

Input Parameters

Pint ws_type
  Workstation type.

Pint colr_ind
  Predefined color index (>=0).

Output Parameters

Pint *err_ind
  Error indicator.

Pcolr_rep *colr_rep
  Predefined color representation.

FORTRAN

PQPCR (wtype, pci, ccsbsz, errind, ol, cspec)

Input Parameters

integer wtype
  Workstation type.

integer pci
  Predefined color index (>=0).

integer ccsbsz
  Color component specification buffer size (>=0).

Output Parameters

384 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
integer errind
    Error indicator.

integer ol
    Number of color components in the color specification.

real cspec(ccsbsz)
    Color specification.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE PREDEFINED EDGE REPRESENTATION (PHOP,*,*,*)

Purpose

Use Inquire Predefined Edge Representation to inquire the predefined settings for the edge attributes in the edge bundle table of the specified workstation type.

The graPHIGS API returns the edge flag setting, edge line type, edge width scale factor, and edge color for the predefined edge bundle table. The returned attributes correspond to the requested bundle table index.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2  Function Requires State (PHOP,*,*,*)
52  Workstation Type Not Recognized By Implementation
51  Information Not Available For Generic Workstation Type
59  Specified Workstation Does Not Have Output Capability
62  This Information Not Available For MO Workstation Type
100  Bundle Index Value Is Less Than One
102  Representation Has Not Been Predefined On This Workstation

Language Bindings

C

cinq_pred_edge_rep (ws_type, index, err_ind, bundle)

Input Parameters

Pint ws_type
    Workstation type.

Pint index
    Predefined edge index (>=1).

Output Parameters
Pint *err_ind
    Error indicator.

Pedge_bundle *bundle
    Predefined edge representation.

FORTRAN

PQPEDR (wtype, pedi, errind, edflag, edtype, ewidth, coli)

Input Parameters

integer wtype
    Workstation type.

integer pedi
    Predefined edge index (>=1).

Output Parameters

integer errind
    Error indicator.

integer edflag
    Edge flag (0=POFF, 1=PON).

integer edtype
    Edge type.

real ewidth
    Edge width scale factor.

integer coli
    Edge color index.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE PREDEFINED INTERIOR REPRESENTATION (PHOP,*,*,*)

Purpose

Use Inquire Predefined Interior Representation to inquire the predefined settings for the interior attributes in the interior bundle table of the specified workstation type.

The graPHIGS API returns the interior style, the style index, and the interior color index for the predefined interior bundle table. The returned attributes correspond to the requested bundle table index. Possible interior styles include: HOLLOW, SOLID, PATTERN, HATCH, and EMPTY.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type
100 Bundle Index Value Is Less Than One
102 Representation Has Not Been Predefined On This Workstation

Language Bindings

C

cpqq_pred_int_rep (ws_type, index, err_ind, bundle)

Input Parameters

Pint ws_type
    Workstation type.

Pint index
    Predefined interior index (>=1).

Output Parameters

Pint *err_ind
    Error indicator.

Pint_bundle *bundle
    Predefined interior representation.

FORTRAN

PQPIR (wtype, pii, errind, style, stylid, coli)

Input Parameters

integer wtype
    Workstation type.

integer pii
    Predefined interior index (>=1).

Output Parameters

integer errind
    Error indicator.

integer style
    Interior style (0=PHOLLO, 1=PSOLID, 2=PPATTR, 3=PHATCH, 4=PISEMP).

integer stylid
    Interior style index.

integer coli
    Interior color index.

Errors

None
Related Subroutines

- Inquire Workstation Connection And Type

---

**INQUIRE PREDEFINED PATTERN REPRESENTATION (PHOP,**,***)**

**Purpose**

Use Inquire Predefined Pattern Representation to inquire a predefined pattern table entry.

For a given workstation type, the graPHIGS API returns the values corresponding to the specified index of the predefined pattern table. The graPHIGS API returns the pattern color index array.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,**,***)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
59 Specified Workstation Does Not Have Output Capability
62 This Information Not Available For MO Workstation Type
112 Pattern Index Value < ONE
102 Representation Has Not Been Predefined On This Workstation

**Language Bindings**

**C**

```c
pinq_pred_pat_rep (ws_type, index, store, err_ind, pat_rep)
```

**Input Parameters**

- `Pint ws_type`: Workstation type.
- `Pint index`: Predefined pattern index (>=1).
- `Pstore store`: Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store ([CREATE STORE (PHOP,**,***)](#)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

**Output Parameters**

- `Pint *err_ind`: Error indicator.
- `Ppat_rep **pat_rep`: Predefined pattern representation. The memory referenced by `*pat_rep` is managed by the parameter `store`.

**FORTRAN**

```fortran
PQPPAR (wtype, ppai, dimx, dimy, errind, dx, dy, colia)
```

---

388 The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
**Input Parameters**

- `integer wtype`
  - Workstation type.

- `integer ppai`
  - Predefined pattern index (>=1).

- `integer dimx`
  - Maximum x-axis dimension of `colia` (>=0).

- `integer dimy`
  - Maximum y-axis dimension of `colia` (>=0).

**Output Parameters**

- `integer errind`
  - Error indicator.

- `integer dx`
  - x-axis dimension of the pattern color index array.

- `integer dy`
  - y-axis dimension of the pattern color index array.

- `integer colia (dimx,dimy)`
  - Pattern color index array.

**Errors**

None

**Related Subroutines**

- Inquire Workstation Connection And Type

---

**INQUIRE PREDEFINED POLYLINE REPRESENTATION (PHOP,*,*,*)**

**Purpose**

Use Inquire Predefined Polyline Representation to inquire the predefined polyline attributes in an entry of the bundle table for a specified workstation type.

The graPHIGS API returns the polyline type, width, and color for the specified index of the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **2** Function Requires State (PHOP,*,*,*)
- **52** Workstation Type Not Recognized By Implementation
- **51** Information Not Available For Generic Workstation Type
- **59** Specified Workstation Does Not Have Output Capability
- **62** This Information Not Available For MO Workstation Type
- **100** Bundle Index Value Is Less Than One

---

Chapter 15. Inquire Subroutines 389
Language Bindings

C

`pinq_pred_line_rep (ws_type, index, err_ind, bundle)`

Input Parameters

`Pint ws_type`
Workstation type.

`Pint index`
Predefined polyline index (>=1).

Output Parameters

`Pint *err_ind`
Error indicator.

`Pline_bundle *bundle`
Predefined polyline representation.

FORTRAN

`PQPPLR (wtype, pli, errind, ltype, lwidth, coli)`

Input Parameters

`integer wtype`
Workstation type.

`integer pli`
Predefined polyline index (>=1).

Output Parameters

`integer errind`
Error indicator.

`integer ltype`
Line type.

`real lwidth`
Line width scale factor.

`integer coli`
Polyline color index.

Errors

None

Related Subroutines

- Inquire Workstation Connection And Type
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (PHOP,*,*,* )

Purpose

Use Inquire Predefined Polymarker Representation to inquire the predefined polymarker attributes corresponding to the specified entry in the predefined bundle table of the specified workstation type.

The graPHIGS API returns the polymarker type, size, and color for the specified index of the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 2 Function Requires State (PHOP,*,*,* )
- 52 Workstation Type Not Recognized By Implementation
- 51 Information Not Available For Generic Workstation Type
- 59 Specified Workstation Does Not Have Output Capability
- 62 This Information Not Available For MO Workstation Type
- 100 Bundle Index Value Is Less Than One
- 102 Representation Has Not Been Predefined On This Workstation

Language Bindings

C

pinq_pred_marker_rep (ws_type, index, err_ind, bundle)

Input Parameters

Pint ws_type  
Workstation type.

Pint index  
Predefined polymarker index (>=1).

Output Parameters

Pint *errind  
Error indicator.

Pmarker_bundle *bundle  
Predefined polymarker representation.

FORTRAN

PQPPMR (wtype, pmi, errind, mtype, mszsf, coli)

Input Parameters

integer wtype  
Workstation type.

integer pmi  
Predefined polymarker index (>=1).
Output Parameters

*integer* errind
   Error indicator.

*integer* mtype
   Marker type.

*real* mszsf
   Marker size scale factor.

*integer* coli
   Polymarker color index.

Errors

None

Related Subroutines

- Inquire Workstation Connection And Type

---

**INQUIRE PREDEFINED TEXT REPRESENTATION (PHOP,* *,*)**

Purpose

Use Inquire Predefined Text Representation to inquire the predefined text attributes corresponding to the specified entry in the text bundle table for the specified workstation type.

The graPHIGS API returns text font, text precision, character expansion factor, character spacing, and text color for the specified entry. Possible text precisions include: *STRING*, *CHARACTER*, and *STROKE*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **2**: Function Requires State (PHOP,* *,*)
- **52**: Workstation Type Not Recognized By Implementation
- **51**: Information Not Available For Generic Workstation Type
- **59**: Specified Workstation Does Not Have Output Capability
- **62**: This Information Not Available For MO Workstation Type
- **100**: Bundle Index Value Is Less Than One
- **102**: Representation Has Not Been Predefined On This Workstation

Language Bindings

C

`pinq_pred_text_rep (ws_type, index, err_ind, bundle)`

Input Parameters

*Pint* ws_type
   Workstation type.

*Pint* index
   Predefined text index (>=1).
Output Parameters

- *err_ind
  Error indicator.

- *bundle
  Predefined text representation.

**FORTRAN**

- PQPTXR (wtype, ptxi, errind, font, prec, chxp, chsp, coli)

**Input Parameters**

- integer wtype
  Workstation type.

- integer ptxi
  Predefined text index (>=1).

**Output Parameters**

- integer errind
  Error indicator.

- integer font
  Text font.

- integer prec
  Text precision. (0=PSTRP, 1=PCHARP, 2=PSTRKP).

- real chxp
  Character expansion factor.

- real chsp
  Character spacing.

- integer coli
  Text color index.

**Errors**

None

**Related Subroutines**

- Inquire Workstation Connection And Type

---

**INQUIRE PREDEFINED VIEW REPRESENTATION (PHOP,*,*,*)**

**Purpose**

Use Inquire Predefined View Representation to inquire the predefined view attributes corresponding to the specified view table entry for the specified workstation type.

The graPHIGS API returns the view orientation matrix, the view mapping matrix, the view clipping limits, the x to y clipping indicator, and the back and front clipping indicators for the specified entry.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:
Function Requires State (PHOP,*,*,*)

Workstation Type Not Recognized By Implementation

Information Not Available For Generic Workstation Type

Specified Workstation Is Of Category MI

View Index Value < ZERO

Specified Representation Has Not Been Defined

This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_pred_view_rep (ws_type, index, err_ind, view)

Input Parameters

Pint ws_type
  Workstation type.

Pint index
  Predefined view index (>=0).

Output Parameters

Pint *err_ind
  Error indicator.

Pview_rep3 *view
  View representation.

FORTRAN

PQPVWR (wtype, pvwi, errind, vwormt, vwmpmt, vwcplm, xyclpi, bclipi, fclipi)

Input Parameters

integer wtype
  Workstation type

integer pvwi
  Predefined view index (>=0).

Input Parameters

integer errind
  Error indicator.

real vwormt(4,4)
  View orientation matrix.

real vwmpmt(4,4)
  View mapping matrix.

real vwcplm(6)
  View clipping limits in NPC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).
integer xyclpi
    x to y clipping indicator (0=PNCLIP, 1=PCLIP).

integer bclipi
    Back clipping indicator (0=PNCLIP, 1=PCLIP).

integer fclipi
    Front clipping indicator (0=PNCLIP, 1=PCLIP).

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE SET OF OPEN WORKSTATIONS (PHOP,*,*,*)

Purpose

Use Inquire Set of Open Workstations to inquire the set of currently open workstations.

The graPHIGS API returns data indicating the total number of open workstation identifiers and the requested set of workstation identifiers.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pinq_open_wss (num elems appl list, start_ind, err ind, open ws ids, num elems impl list)

Input Parameters

Pint num elems appl list
    Number of elements in the application list (>=0).

Pint start ind
    Starting index (>=0).

Output Parameters

Pint *err ind
    Error indicator.

Pint_list *open ws ids
    List of workstation identifiers.

Pint *num elems impl list
    Number of elements in the implementation list.

FORTRAN

PQOPWK (n, errind, ol, wkid)
Input Parameters

integer n
   Set member requested (>=0).

Output Parameters

integer errind
   Error indicator.

integer ol
   Number of open workstations.

integer wkid
   n\textsuperscript{th} member in the set of open workstations.

Errors

None

Related Subroutines

• Open Workstation
• Close Workstation

INQUIRE SET OF WORKSTATIONS TO WHICH POSTED (PHOP,*,*,*)

Purpose

Use Inquire Set of Workstations to Which Posted to inquire the list of workstations to which the specified structure is posted.

This subroutine returns the number of workstations and the workstation identifiers to which the specified structure is posted.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 \hspace{1cm} \text{Function Requires State (PHOP,*,*,*)}
201 \hspace{1cm} \text{Specified Structure Does Not Exist}

Language Bindings

C

\texttt{pinq\_wss\_posted (struct\_id, num\_elems\_appl\_list, start\_ind, err\_ind, ws, num\_elems\_impl\_list)}

Input Parameters

\textit{Pint} struct\_id
   Structure identifier.

\textit{Pint} num\_elems\_appl\_list
   Number of elements in the application list (>=0).
\textit{Pint start\_ind}  
Starting index (>=0).

\textbf{Output Parameters}
\textit{Pint *err\_ind}  
Error indicator.

\textit{Pint\_list *ws}  
List of workstations to which the structure is posted.

\textit{Pint *num\_elems\_impl\_list}  
Number of elements in the implementation list.

\textbf{FORTRAN}

\texttt{PQWKPO (strid, n, errind, ol, wkid)}

\textbf{Input Parameters}
\textit{integer strid}  
Structure identifier.

\textit{integer n}  
Set member requested (>=0).

\textbf{Output Parameters}
\textit{integer errind}  
Error indicator.

\textit{integer ol}  
Number of workstations to which the structure is posted.

\textit{integer wkid}  
\(n^{th}\) member in the set of workstations to which the structure is posted.

\textbf{Errors}

None

\textbf{Related Subroutines}
\begin{itemize}
  \item None
\end{itemize}

\textbf{INQUIRE STRING DEVICE STATE (PHOP,WSOP,*,*)}

\textbf{Purpose}

Use Inquire String Device State to inquire the current state of the specified string device on the specified workstation.

The graPHIGS API returns the current values of the specified device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

\begin{itemize}
  \item 3 Function Requires State (PHOP,WSOP,*,*)
  \item 54 Specified Workstation Is Not Open
\end{itemize}
Language Bindings

C

`ping_string_st (ws_id, string_num, store, err_ind, op_mode, echo_switch, init_string, prompt_echo, echo_area, string_data)`

Input Parameters

`Pint ws_id`
Workstation identifier.

`Pint string_num`
String device number (>=1).

`Pstore store`
Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (`CREATE STORE (PHOP,*,*,*)`) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

`Pint *err_ind`
Error indicator.

`Pop_mode *op_mode`
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

`Pecho_switch *echo_switch`
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

`char **init_string`
Initial string. The memory referenced by `*init_string` is managed by the parameter `store`.

`Pint *prompt_echo`
Prompt and echo type.

`Plimit *echo_area`
Echo area in DC.

`Pstring_data **string_data`
Data record. The memory referenced by `*string_data` is managed by the parameter `store`.

FORTRAN

`PQSTS (wkid, stdnr, mldr, errind, mode, esw, lostr, istr, pet, earea, ldr, datrec)`

Input Parameters

`integer wkid`
Workstation identifier.

`integer stdnr`
String device number (>=1).

`integer mldr`
Dimension of the data record array (>=0).
Output Parameters

integer errind
   Error indicator.

integer mode
   Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

integer lostr
   Number of characters returned.

character*(*) istr
   Initial string.

integer pet
   Prompt and echo type

real earea(4)
   Echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
   Number of array elements used in the data record.

character*80  datrec(mldr)
   The data record.

FORTRAN Subset

PQSTS (wkid, stdnr, mldr, errind, mode, esw, lostr, istr, pet, earea, ldr, datrec)

Input Parameters

integer wkid
   Workstation identifier.

integer stdnr
   String device number (>=1).

integer mldr
   Dimension of the data record array = ???(>=0).

Output Parameters

integer errind
   Error indicator.

integer mode
   Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

integer lostr
   Number of characters returned.

character*80 istr
   Initial string.

integer pet
   Prompt and echo type.
real earea(4)
   Echo area in DC (XMIN, XMAX, YMIN, YMAX).

integer ldr
   Number of array elements used in the data record.

character*80 datrec(mldr)
   The data record.

Errors
None

Related Subroutines
- Initialize String
- Initialize String 3
- Inquire Default String Device Data
- Inquire Default String Device Data 3
- Inquire String Device State 3

INQUIRE STRING DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose
Use Inquire String Device State 3 to inquire the current state of the specified string device on the specified workstation.

The graPHIGS API returns the current values of the specified device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3     Function Requires State (PHOP,WSOP,*,*)
54    Specified Workstation Is Not Open
61    Specified Workstation Is Not Of Category Input Or Outin
250   Specified Device Not Available On Workstation

Language Bindings

C

pinq_string_st3 (ws_id, string_num, store, err_ind, op_mode, echo_switch, init_string, prompt_echo, echo_vol, string_data)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint string_num
   String device number (>=1).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of
subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

**Output Parameters**

*Pint* `*err_ind`
Error indicator.

*Pop_mode* `*op_mode`
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

*Pecho_switch* `*echo_switch`
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

*char* `**init_string`
Initial string. The memory referenced by `*init_string` is managed by the parameter `store`.

*Pint* `*prompt_echo`
Prompt and echo type.

*Plimit3* `*echo_vol`
Echo volume in DC.

*Pstring_data3* `**string_data`
Data record. The memory referenced by `*string_data` is managed by the parameter `store`.

**FORTRAN**

`PQSTS3 (wkid, stdnr, mldr, errind, mode, esw, lostr, istr, pet, evol, ldr, datrec)`

**Input Parameters**

`integer wkid`
Workstation identifier.

`integer stdnr`
String device number (>=1).

`integer mldr`
Dimension of the data record array (>=0).

**Output Parameters**

`integer errind`
Error indicator.

`integer mode`
Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

`integer esw`
Echo switch (0=PNECHO, 1=PECHO).

`integer lostr`
Number of characters returned.

`character(*) istr`
Initial string.

`integer pet`
Prompt and echo type.

`real evol(6)`
Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).
integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    The data record.

FORTRAN Subset

PQSTS3 (wkid, stdnr, mldr, errind, mode, esw, lostr, istr, pet, evol, ldr, datrec)

Input Parameters

integer wkid
    Workstation identifier.

integer stdnr
    String device number (>=1).

integer mldr
    Dimension of the data record array (>=0).

Output Parameters

integer errind
    Error indicator.

integer mode
    Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
    Echo switch (0=PNECHO, 1=PECHO).

integer lostr
    Number of characters returned.

character*80 istr
    Initial string.

integer pet
    Prompt and echo type

real evol(6)
    Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
    Number of array elements used in the data record.

character*80 datrec(mldr)
    The data record.

Errors

None

Related Subroutines

- Initialize String
- Initialize String 3
- Inquire Default String Device Data
- Inquire Default String Device Data 3
- Inquire String Device State
Purpose

Use Inquire Stroke Device State to inquire the current state of the specified stroke device on the specified workstation. Returned values of type SET or REALIZED may be specified.

The graPHIGS API returns the current values for the specified device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*,*)
54 Specified Workstation Is Not Open
61 Specified Workstation Is Not Of Category Input Or Outin
250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_stroke_st (ws_id, stroke_num, type, store, err_ind, op_mode, echo_switch, init_view_ind, init_stroke, prompt_echo, echo_area, stroke_data)

Input Parameters

Pint ws_id
Workstation identifier.

Pint stroke_num
Stroke device number (>=1).

Pinq_type type
Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store ([CREATE STORE (PHOP,*,*,*)]) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
Error indicator.

Pop_mode *op_mode
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pint *init_view_ind
Initial view index.

Ppoint_list **init_stroke
Initial stroke. The memory referenced by *init_stroke is managed by the parameter store.
Pint *prompt_echo
Prompt and echo type.

Plimit *echo_area
Echo area in DC.

Pstroke_data **stroke_data
Data record. The memory referenced by *stroke_data is managed by the parameter store.

**FORTRAN**

PQSXS (wkid, skdnr, type, n, mldr, errind, mode, esw, iviewi, np, ipxa, ipya, pet, earea, ldr, datrec)

**Input Parameters**

integer wkid
   Workstation identifier.

integer skdnr
   Stroke device number (>=1).

integer type
   Type of returned values (0=PSET, 1=PREAL).

integer n
   Maximum number of points (>=0).

integer mldr
   Dimension of the data record array (>=0).

**Output Parameters**

integer errind
   Error indicator.

integer mode
   Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
   Echo switch (0=PNECHO, 1=PECHO).

integer iviewi
   Initial view index.

integer np
   Number of points.

real ipxa(n)
   x coordinates of the initial stroke in WC.

real ipya(n)
   y coordinates of the initial stroke in WC.

integer pet
   Prompt and echo type.

real earea(4)
   Echo area in DC. (XMIN, XMAX, YMIN, YMAX).

integer ldr
   Number of array elements used in the data record.

character*80 datrec(mldr)
   Data record.
Errors

None

Related Subroutines
- Initialize Stroke
- Initialize Stroke 3
- Inquire Default Stroke Device Data
- Inquire Default Stroke Device Data 3
- Inquire Stroke Device State 3

INQUIRE STROKE DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose

Use Inquire Stroke Device State 3 to inquire the current state of the specified stroke device on the specified workstation. Returned values of type SET or REALIZED may be specified.

The graPHIGS API returns the current values for the specified device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3    Function Requires State (PHOP,WSOP,*,*)
54   Specified Workstation Is Not Open
61   Specified Workstation Is Not Of Category Input Or Outin
250  Specified Device Not Available On Workstation

Language Bindings

C

pinq_stroke_st3 (ws_id, stroke_num, type, store, err_ind, op_mode, echo_switch, init_view_ind, init_stroke, prompt_echo, echo_vol, stroke_data)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint stroke_num
   Stroke device number (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Pstore store
   Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
   Error indicator.
Pop_mode *op_mode
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

Pint *init_view_ind
Initial view index.

Ppoint_list3 **init_stroke
Initial stroke in WC. The memory referenced by *init_stroke is managed by the parameter store.

Pint *prompt_echo
Prompt and echo type.

Plimit3 *echo_vol
Echo volume in DC.

Pstroke_data3 **stroke_data
Data record. The memory referenced by *stroke_data is managed by the parameter store.

FORTRAN

PQSKS (wkid, skdnr, type, n, mldr, errind, mode, esw, iviewi, np, ipxa, ipya, ipza, pet, evol, ldr, datrec)

Input Parameters

integer wkid
Workstation identifier.

integer skdnr
Stroke device number (>=1).

integer type
Type of returned values (0=PSET, 1=PREAL).

integer n
Maximum number of points (>=0).

integer mldr
Dimension of the data record array (>=0).

Output Parameters

integer errind
Error indicator.

integer mode
Operating mode (0=PREQ, 1=PSAMPL, 2=PEVENT).

integer esw
Echo switch (0=PNECHO, 1=PECHO).

integer iviewi
Initial view index.

integer np
Number of points.

real ipxa(n)
x coordinates of the initial stroke in WC.

real ipya(n)
y coordinates of the initial stroke in WC.
real ipza(n)
z coordinates of the initial stroke in WC.

integer pet
Prompt and echo type.

real evol(6)
Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
Number of array elements used in the data record.

character*80 datrec(ldr)
Data record.

Errors
None

Related Subroutines
• Initialize Stroke
• Initialize Stroke 3
• Inquire Default Stroke Device Data
• Inquire Default Stroke Device Data 3
• Inquire Stroke Device State

INQUIRE STRUCTURE IDENTIFIERS (PHOP,*,*,*)

Purpose
Use Inquire Structure Identifiers to inquire the currently existing structure identifiers in the centralized structure store.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pinq_struct_ids (num elems appl list, start ind, err ind, struct ids, num elems impl list)

Input Parameters

Pint num elems appl list
Number of elements in the application list (>=0).

Pint start ind
Starting index (>=0).

Output Parameters

Pint *err ind
Error indicator.
Pint_list *struct_ids
   List of structure identifiers.

Pint *num elems_impl_list
   Number of elements in the implementation list.

FORTRAN

PQSID (n, errind, number, strid)

Input Parameters
integer n
   Set member requested (>=0).

Output Parameters
integer errind
   Error indicator.
integer number
   Number of structure identifiers.
integer strid
   n\textsuperscript{th} structure identifier.

Errors
None

Related Subroutines
• None

INQUIRE STRUCTURE STATE VALUE (PHCL,WSCL,STCL,ARCL)

Purpose
Use Inquire Structure State Value to inquire the structure state of the graPHIGS API. The structure state is either Structure Open (STOP) or Structure Closed (STCL).

Language Bindings

C

pinq_struct_st (struct_st)

Output Parameters
Pstruct_st *struct_st
   Structure state value (0=PSTRUCT_ST_STCL, 1=PSTRUCT_ST_STOP).

FORTRAN

PQSTRS (strsta)

Output Parameters
integer strsta
Structure state value (0=PSTCL, 1=PSTOP).

Errors
None

Related Subroutines
- Open Structure
- Close Structure

INQUIRE STRUCTURE STATUS (PHOP,*,*,*)

Purpose
Use Inquire Structure Status to inquire whether or not the specified structure exists in the centralized structure store. If the specified structure does not exist, then the graPHIGS API sets the structure status indicator to NON-EXISTENT. If the structure exists and contains no elements, then the graPHIGS API sets the structure status indicator to EMPTY; otherwise, the graPHIGS API sets the structure status indicator to NOT EMPTY.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the structure status value in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to the following error:

2 Function Requires State (PHOP,*,*,*)

Language Bindings

C

pinq_struct_status (struct_id, err_ind, status)

Input Parameters
Pint struct_id
Structure identifier.

Output Parameters
Pint *err_ind
Error indicator.
Pstruct_status *status
Structure status identifier (0=PSTRUCT_STATUS_NON_EXISTENT, 1=PSTRUCT_STATUS_EMPTY, 2=PSTRUCT_STATUS_NOT_EMPTY).

FORTRAN
PQSTST (strid, errind, strst)

Input Parameters
integer strid
Structure identifier.

Output Parameters
integer errind
   Error indicator.

integer strsti
   Structure status identifier (0=PSNOEX, 1=PSEMPT, 2=PSNEMP).

Errors
None

Related Subroutines
• None

INQUIRE SYSTEM STATE VALUE (PHCL,WSCL,STCL,ARCL)

Purpose
Use Inquire System State Value to inquire the system state of the graPHIGS API. The system state is either PHIGS Open (PHOP) or PHIGS Closed (PHCL).

Language Bindings

C
pinq_sys_st (sys_st)

Output Parameters

Psys_st *sys_st
   System state value (0=PSYS_ST_PHCL, 1=PSYS_ST_PHOP).

FORTRAN

PQSYS (syssta)

Output Parameters

integer syssta
   System state value (0=PPHCL, 1=PPHOP).

Errors
None

Related Subroutines
• Close PHIGS
• Open PHIGS

INQUIRE TEXT EXTENT (PHOP,*,*,*)

Purpose
Use Inquire Text Extent to inquire the extent of the specified character string in the local 2D text coordinate system. The graPHIGS API uses the specified text attributes for the specified workstation type to compute the extent. STROKE precision is assumed. The text position is (0,0) in the local 2D text coordinate system.
Possible text path attributes include: RIGHT, LEFT, UP or DOWN. The horizontal alignment for the text may be set to NORMAL, LEFT, CENTER, or RIGHT, and the vertical text alignment may be set to NORMAL, TOP, CAP, HALF, BASE or BOTTOM.

The concatenation offset, with a suitable modeling transformation applied to account for the character up vector, indicates the text position for the concatenation of a subsequent text output primitive in the local 2D text coordinate system. This includes for text paths RIGHT and LEFT, a suitable modification to adjust for the intercharacter spacing of the last character as specified by the character spacing parameter.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
62 This Information Not Available For MO Workstation Type
106 Specified Font Not Available For Requested Text Precision

Language Bindings

C

pinq_text_extent (ws_type, text_font, char_expan, char_space, char_ht, text_path, hor_text_align, vert_text_align, char_string, err_ind, rect, offset)

Input Parameters

Pint ws_type
  Workstation type.
Pint text_font
  Text font.
Pfloat char_expan
  Character expansion factor.
Pfloat char_space
  Character spacing.
Pfloat char_ht
  Character height.

Ptext_path text_path
  Text path (0=PPATH_RIGHT, 1=PPATH_LEFT, 2=PPATH_UP, 3=PPATH_DOWN).

Phor_text_align hor_text_align
  Horizontal text alignment (0=PHOR_NORM, 1=PHOR_LEFT, 2=PHOR_CTR, 3=PHOR_RIGHT).

Pvert_text_align vert_text_align
  Vertical text alignment (0=PVERT_NORM, 1=PVERT_TOP, 2=PVERT_CAP, 3=PVERT_HALF, 4=PVERT_BASE, 5=PVERT_BOTTOM).

const char *char_string
  Character string.

Output Parameters
Pint *err_ind
    Error indicator.

Prect *rect
    Extent rectangle.

Ppoint *offset
    Concatenation offset.

FORTRAN

PQTXX (wktype, font, chxp, chsp, chh, txp, txalh, txalv, str, errind, txexrx, txexry, copx, copy)

Input Parameters

integer wktype
    Workstation type.

integer font
    Text font.

real chxp
    Character expansion factor.

real chsp
    Character spacing.

real chh
    Character height.

integer txp
    Text path (0=PRIGHT, 1=PLEFT, 2=PUP, 3=PDOWN).

integer txalh
    Horizontal text alignment (0=PAHNOR, 1=PALEFT, 2=PACENT, 3=PARITE).

integer txalv
    Vertical text alignment (0=PAVNOR, 1=PATOP, 2=PACAP, 3=PAHALF, 4=PABASE, 5=PABOTT).

character(*) str
    Character string.

Output Parameters

integer errind
    Error indicator.

real txexrx(2)
    x coordinates of the text extent rectangle.

real txexry(2)
    y coordinates of the text extent rectangle.

real copx
    x coordinate of the concatenation offset.

real copy
    y coordinate of the concatenation offset.

FORTRAN Subset

PQTXXS (wktype, font, chxp, chsp, chh, txp, txalh, txalv, lstr, str, errind, txexrx, txexry, copx, copy)
Input Parameters

integer wktype
    Workstation type.

integer font
    Text font.

real chxp
    Character expansion factor.

real chsp
    Character spacing.

real chh
    Character height.

integer txp
    Text path (0=PRIGHT, 1=PLEFT, 2=PUP, 3=PDOWN).

integer txalh
    Horizontal text alignment (0=PAHNOR, 1=PALEFT, 2=PACENT, 3=PARITE).

integer txalv
    Vertical text alignment (0=PAVNOR, 1=PATOP, 2=PACAP, 3=PAHALF, 4=PABASE, 5=PABOTT).

integer lstr
    Length of string (in characters).

character*80 str
    Character string.

Output Parameters

integer errind
    Error indicator.

real txexrx(2)
    x coordinates of the text extent rectangle.

real txexry(2)
    y coordinates of the text extent rectangle.

real copx
    x coordinate of the concatenation offset.

real copy
    y coordinate of the concatenation offset.

Errors

None

Related Subroutines
- None

INQUIRE TEXT FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire Text Facilities to inquire the text facilities for the specified workstation type.
Only the highest supported precision for each font is present in the list of text font and precision pairs. Possible text precisions include: STRING, CHARACTER, or STROKE.

If the graPHIGS API returns a value of zero for the number of available character heights, then the workstation supports a continuous range of character heights. If the graPHIGS API returns a value of zero for the number of available character expansion factors, then the workstation supports a continuous range of character expansion factors.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 2 Function Requires State (PHOP,*,*,*)
- 52 Workstation Type Not Recognized By Implementation
- 51 Information Not Available For Generic Workstation Type
- 59 Specified Workstation Does Not Have Output Capability
- 62 This Information Not Available For MO Workstation Type

**Language Bindings**

**C**

`pinq_text_facs (ws_type, numelems_appl_list, start_ind, err_ind, fac, numelems_impl_list)`

**Input Parameters**

- `Pint ws_type`  
  Workstation type.

- `Pint numelems_appl_list`  
  Number of elements in the application list (>=0).

- `Pint start_ind`  
  Starting index (>=0).

**Output Parameters**

- `Pint *err_ind`  
  Error indicator.

- `Ptext_facs *fac`  
  Text facilities.

- `Pint *numelems_impl_list`  
  Number of elements in the implementation list.

**FORTRAN**

`PQTXF (wtype, n, errind, nfpp, font, prec, nchh, minchh, maxchh, nchx, minchx, maxchx, nptxi)`

**Input Parameters**

- `integer wtype`  
  Workstation type.

- `integer n`  
  List element requested (>=0).
Output Parameters

integer errind
   Error indicator.

integer nfpp
   Number of text font and precision pairs.

integer font
   $n^{th}$ element in the list of text fonts.

integer prec
   $n^{th}$ element in the list of text precisions ($0=PSTRP$, $1=PCHARP$, $2=PSTRKP$).

integer nchh
   Number of available character heights.

real minchh
   Minimum character height in DC.

real maxchh
   Maximum character height in DC.

integer nchx
   Number of available character expansion factors.

real minchx
   Minimum character expansion factor.

real maxchx
   Maximum character expansion factor.

integer nptxi
   Number of predefined text indexes.

Errors

None

Related Subroutines

- Inquire Workstation Connection And Type
- Set Text Representation
- Text
- Text 3

INQUIRE TEXT REPRESENTATION (PHOP,WSOP,*,*)

Purpose

Use Inquire Text Representation to inquire the current text attributes contained in the specified entry in the text bundle table of the specified workstation. Returned values of type SET or REALIZED may be specified.

The graPHIGS API returns data indicating the text font and precision, character expansion factor and spacing, and text color of the specified entry. Possible text precisions include: STRING, CHARACTER, and STROKE.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:
3 Function Requires State (PHOP, WSOP, *, *)
54 Specified Workstation Is Not Open
59 Specified Workstation Does Not Have Output Capability
100 Bundle Index Value Is Less Than One
101 Specified Representation Has Not Been Defined

Language Bindings

C

pinq_text_rep (ws_id, index, type, err_ind, text_rep)

Input Parameters

Pint ws_id
   Workstation identifier.

Pint index
   Text index (>=1).

Pinq_type type
   Type of returned values (0=PINQ_SET, 1=PINQ_REALIZED).

Output Parameters

Pint *err_ind
   Error indicator.

Ptext_bundle *text_rep
   Text representation.

FORTRAN

PQTXR (wkid, txi, type, errind, font, prec, chxp, chsp, coli)

Input Parameters

integer wkid
   Workstation identifier.

integer txi
   Text index (>=1).

integer type
   Type of returned values (0=PSET, 1=PREAL).

Output Parameters

integer errind
   Error indicator.

integer font
   Text font.

integer prec
   Text precision (0=PSTRP, 1=PCHARP, 2=PSTRKP).

real chxp
   Character expansion factor.
real chsp
    Character spacing.

integer coli
    Text color index.

Errors
None

Related Subroutines
• Set Text Representation

INQUIRE VALUATOR DEVICE STATE (PHOP,WSOP,*,*)

Purpose

Use Inquire Valuator Device State to inquire the current state of the specified valuator device on the specified workstation.

The graPHIGS API returns the values of the specified device. The format and content of these values returned by the graPHIGS API depends on the prompt/echo type defined in the subroutine that initializes the input device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 3 Function Requires State (PHOP,WSOP,*,*)
- 54 Specified Workstation Is Not Open
- 61 Specified Workstation Is Not Of Category Input Or Outin
- 250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_val_st (ws_id, val_num, store, err_ind, op_mode, echo_switch, init_value, prompt_echo, echo_area, val_data)

Input Parameters

Pint ws_id
    Workstation identifier.

Pint val_num
    Valuator device number (>=1).

Pstore store
    Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
    Error indicator.
**Pop_mode** *op_mode*
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

**Pecho_switch** *echo_switch*
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).

**Pfloat** *init_value*
Initial value.

**Pint** *prompt_echo*
Prompt and echo type.

**Plimit** *echo_area*
Echo area in DC.

**Pval_data** *val_data*
Data record. The memory referenced by *val_data is managed by the parameter store.

**FORTRAN**

**PQVLS** *(wkid, vldnr, mldr, errind, mode, esw, ival, pet, earea, ldr, datrec)*

**Input Parameters**

*integer wkid*
Workstation identifier.

*integer vldnr*
Valuator device number (>=1).

*integer mldr*
Dimension of the data record array (>=0).

**Output Parameters**

*integer errind*
Error indicator.

*integer mode*
Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

*integer esw*
Echo switch (0=PNECHO, 1=PECHO).

*real ival*
Initial value.

*integer pet*
Prompt and echo type.

*real earea(4)*
Echo area in DC (XMIN, XMAX, YMIN, YMAX).

*integer ldr*
Number of array elements in the data record.

*character*80 datrec(mldr)*
Data record.

**Errors**

None
Related Subroutines
- Initialize Valuator
- Initialize Valuator 3
- Inquire Default Valuator Device Data
- Inquire Default Valuator Device Data 3
- Inquire Valuator Device State 3

INQUIRE VALUATOR DEVICE STATE 3 (PHOP,WSOP,*,*)

Purpose

Use Inquire Valuator Device State 3 to inquire the current state of the specified valuator device on the specified workstation.

The graPHIGS API returns the values of the specified device. The format and content of these values returned by the graPHIGS API depends on the prompt/echo type defined in the subroutine that initializes the input device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- 3 Function Requires State (PHOP,WSOP,*,*)
- 54 Specified Workstation Is Not Open
- 61 Specified Workstation Is Not Of Category Input Or Outin
- 250 Specified Device Not Available On Workstation

Language Bindings

C

pinq_val_st3 (ws_id, val_num, store, err_ind, op_mode, echo_switch, init_value, prompt_echo, echo_vol, val_data)

Input Parameters

Pint ws_id
Workstation identifier.

Pint val_num
Valuator device number (>=1).

Pstore store
Handle to the store object. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
Error indicator.

Pop_mode *op_mode
Operating mode (0=POP_REQ, 1=POP_SAMPLE, 2=POP_EVENT).

Pecho_switch *echo_switch
Echo switch (0=PSWITCH_NO_ECHO, 1=PSWITCH_ECHO).
Pfloat *init_value
  Initial value.

Pint *prompt_echo
  Prompt and echo type.

Plimit3 *echo_vol
  Echo volume in DC.

Pval_data **val_data
  Data record. The memory referenced by *val_data is managed by the parameter store.

FORTRAN

PQVLS3 (wkid, vldnr, mldr, errind, mode, esw, ival, pet, evol, ldr, datrec)

Input Parameters
integer wkid
  Workstation identifier.

integer vldnr
  Valuator device number (>=1).

integer mldr
  Dimension of the data record array (>=0).

Output Parameters
integer errind
  Error indicator.

integer mode
  Operating mode (0=PREQU, 1=PSAMPL, 2=PEVENT).

integer esw
  Echo switch (0=PNECHO, 1=PECHO).

real ival
  Initial value.

integer pet
  Prompt and echo type.

real evol(6)
  Echo volume in DC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer ldr
  Number of array elements used in the data record.

character*80 datrec(mldr)
  Data record.

Errors

None

Related Subroutines
- Initialize Valuator
- Initialize Valuator 3
- Inquire Default Valuator Device Data
INQUIRE VIEW FACILITIES (PHOP,*,*,*)

Purpose

Use Inquire View Facilities to inquire the view facilities for the specified workstation type.

The graPHIGS API returns data indicating the number of predefined view indexes on the workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the number in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2    Function Requires State (PHOP,*,*,*)
52   Workstation Type Not Recognized By Implementation
51   Information Not Available For Generic Workstation Type
57   Specified Workstation Is Of Category MI
62   This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_view_facs (ws_type, err_ind, num_view_ind)

Input Parameters

Pint ws_type
  Workstation type.

Output Parameters

Pint *err_ind
  Error indicator.

Pint *num_view_ind
  Number of predefined view indexes.

FORTRAN

PQVWF (wtype, errind, npvwi)

Input Parameters

integer wtype
  Workstation type.

Output Parameters

integer errind
  Error indicator.

integer npvwi
  Number of predefined view indexes.
Errors
None

Related Subroutines
- Inquire Workstation Connection And Type
- Set View Representation

INQUIRE VIEW REPRESENTATION (PHOP, WSOP, *, *)

Purpose
Use Inquire View Representation to inquire the current and/or requested values from the specified view table entry of the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3  Function Requires State (PHOP, WSOP, *, *)
54  Specified Workstation Is Not Open
57  Specified Workstation Is Of Category MI
114  View Index Value < ZERO
101  Specified Representation Has Not Been Defined

Language Bindings

C
pinq_view_rep (ws_id, view_ind, err_ind, upd_st, cur_view, req_view)

Input Parameters

Pint ws_id
  Workstation identifier.

Pint view_ind
  View index requested (>=0).

Output Parameters

Pint *err_ind
  Error indicator.

Pupd_st *upd_st
  Viewing transformation update state (0=PUPD_NOT_PEND, 1=PUPD_PEND).

Pview_rep3 *cur_view
  Current view representation.

Pview_rep3 *req_view
  Requested view representation

FORTRAN

PQVWR (wkid, viewi, curq, errind, vwupd, vwormt, vwmpmt, vwcplm, xyclpi, bclpi, fclpi)
Input Parameters

integer wkid
   Workstation identifier.

integer viewi
   View index requested (>=0).

integer curq
   Specifies whether current or requested values are to be returned (0=PCURVL, 1=PRQSVL).

Output Parameters

integer errind
   Error indicator.

integer vwupd
   Viewing transformation update state (0=PNPEND, 1=PPEND).

real vwormt(4,4)
   View orientation matrix.

real vwmptm(4,4)
   View mapping matrix.

real vwclplm(6)
   View clipping limits in NPC (XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX).

integer xyclpi
   x to y clipping indicator (0=PNCLIP, 1=PCLIP).

integer bclipi
   Back clipping indicator (0=PNCLIP, 1=PCLIP).

integer fclipi
   Front clipping indicator (0=PNCLIP, 1=PCLIP).

Errors

None

Related Subroutines

- Set View Representation

INQUIRE WORKSTATION CATEGORY (PHOP,*,*,*)

Purpose

Use Inquire Workstation Category to inquire the category of the specified workstation type.

The graPHIGS API returns the category of the workstation type indicating whether it is output (OUTPUT), input (INPUT), output and input (OUTIN), metafile output (MO), or metafile input (MI).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the workstation category in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
52 Workstation Type Not Recognized By Implementation
51 Information Not Available For Generic Workstation Type
Language Bindings

C

pinq_ws_cat (ws_type, err_ind, cat)

Input Parameters

Pint ws_type
   Workstation type.

Input Parameters

Pint *err_ind
   Error indicator

Pws_cat *cat
   Workstation category (0=PCAT_OUT, 1=PCAT_IN, 2=PCAT_OUTIN, 3=PCAT_MO, 4=PCAT_MI).

FORTRAN

PQWKCA (wtype, errind, wkcat)

Input Parameters

integer wtype
   Workstation type.

Output Parameters

integer errind
   Error indicator.

integer wkcat
   Workstation category (0=POUTPT, 1=PINPUT, 2=POUTIN, 3=PMO, 4=PMI).

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE WORKSTATION CLASSIFICATION (PHOP,*,*,*)

Purpose

Use Inquire Workstation Classification to inquire the display classification of the specified workstation type.

The graPHIGS API returns a value indicating the type of display technology utilized by the specified workstation type. Possible classifications include: VECTOR, RASTER, and OTHER.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the display classification in the output parameter. If the information is unavailable, then the value returned in the output parameter is unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2 Function Requires State (PHOP,*,*,*)
Language Bindings

C

cinq_ws_class (ws_type, err_ind, ws_class)

Input Parameters

Pint ws_type
    Workstation type.

Output Parameters

Pint *err_ind
    Error indicator.

Pws_class *ws_class
    Workstation classification (0=PCLASS_VEC, 1=PCLASS_RASTER, 2=PCLASS_OTHER).

FORTRAN

PQWKCL (wtype, errind, vrtype)

Input Parameters

integer wtype
    Workstation type.

Output Parameters

integer errind
    Error indicator.

integer vrtype
    Workstation classification (0=PVECTR, 1=PRASTR, 2=POTHWK).

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type

INQUIRE WORKSTATION CONNECTION AND TYPE (PHOP,WSOP,*,*)

Purpose

Use Inquire Workstation Connection and Type to inquire the connection identifier and the realized workstation type of the specified workstation identifier. The realized workstation type is the type assigned by the graPHIGS API when the workstation is created by the Open Workstation subroutine.
If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*)
54 Specified Workstation Is Not Open

Language Bindings

C

pinq_ws_conn_type (ws_id, store, err_ind, conn_id, ws_type)

Input Parameters

Pint ws_id
    Workstation identifier.

Pstore store
    Handle to the Store identifier. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store (CREATE STORE (PHOP,*,*)) subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint *err_ind
    Error indicator.

void **conn_id
    Connection identifier. The memory referenced by *conn_id is managed by the parameter store.

Pint *ws_type
    Workstation type.

FORTRAN

PQWKC (wkid, errind, conid, wtype)

Input Parameters

integer wkid
    Workstation identifier.

Output Parameters

integer errind
    Error indicator.

integer conid
    Connection identifier.

integer wtype
    Workstation type.

Errors

None

Related Subroutines
None

INQUIRE WORKSTATION CONNECTION AND TYPE (PHOP, WSOP, *, *)

Purpose

Use Inquire Workstation Connection and Type to inquire the connection identifier and the realized workstation type of the specified workstation identifier. The realized workstation type is the type assigned by the graPHIGS API when the workstation is created by the Open Workstation subroutine.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 FUNCTION REQUIRES STATE (PHOP, WSOP, *, *)
54 SPECIFIED WORKSTATION IS NOT OPEN

Language Bindings

C

pinq_ws_conn_type (ws_id, store, err_ind, conn_id, ws_type);

Input Parameters

Pint    ws_id
    Workstation identifier.
Pstore  store
    Handle to the Store identifier. The graPHIGS API uses an object of type Store to facilitate the use of subroutines which return complex data. See Create Store subroutine for details on how the graPHIGS API uses this parameter on inquiries.

Output Parameters

Pint    *err_ind
    Error indicator.
void    **conn_id
    Connection identifier. The memory referenced by *conn_id is managed by the parameter store.
Pint    *ws_type
    Workstation type.

FORTRAN

pqwkc (wkid, errind, conid, wtype)

Input Parameters

integer wkid
    Workstation identifier.

Output Parameters

integer errind
    Error indicator.
integer conid
    Connection identifier.

integer wtype
    Workstation type.

Errors
None

Related Subroutines
None

INQUIRE WORKSTATION STATE TABLE LENGTHS (PHOP,*,*,*)

Purpose
Use Inquire Workstation State Table Lengths to inquire the maximum number of entries supported for workstation tables for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

2       Function Requires State (PHOP,*,*,*)
52      Workstation Type Not Recognized By Implementation
51      Information Not Available For Generic Workstation Type
59      Specified Workstation Does Not Have Output Capability
62      This Information Not Available For MO Workstation Type

Language Bindings

C

pinq_ws_st_table (ws_type, err_ind, lengths)

Input Parameters

Pint ws_type
    Workstation type.

Output Parameters

Pint *err_ind
    Error indicator.

Pws_st_tables *lengths
    Lengths of workstation tables.

FORTRAN

PQWKSL (wtype, errind, mpibte, mpmbte, mtxbte, minbte, medbte, mpaib, mcol, vvitbi)

Input Parameters
integer wtype
    Workstation type.

Output Parameters

integer errind
    Error indicator.

integer mplbte
    Maximum number of polyline bundle table entries.

integer mpmbte
    Maximum number of polymarker bundle table entries.

integer mtxbte
    Maximum number of text bundle table entries.

integer minbte
    Maximum number of interior bundle table entries.

integer medbte
    Maximum number of edge bundle table entries.

integer mpai
    Maximum number of pattern indexes.

integer mcoli
    Maximum number of color indexes.

integer vwtbi
    Maximum number of view table indexes.

Errors

None

Related Subroutines

• Inquire Workstation Connection And Type
• Set Color Representation
• Set edge Representation
• Set Interior Representation
• Set Polyline Representation
• Set Polymarker Representation
• Set Text Representation
• Set View Representation

INQUIRE WORKSTATION STATE VALUE (PHCL,WSCL,STCL,ARCL)

Purpose

Use Inquire Workstation State Value to inquire the current workstation state of the graPHIGS API. The workstation state is either Workstation Open (WSOP) or Workstation Closed (WSCL). then at least one workstation is open. If the state is WSCL, then no workstations are open.

Language Bindings
### INQUIRE WORKSTATION TRANSFORMATION (PHOP,WSOP,*,*)

**Purpose**

Use Inquire Workstation Transformation to inquire the current and requested workstation transformation values of a specified workstation.

If your application has not updated the workstation, then the graPHIGS API returns a value of *PENDING*. In this case, the requested values reflect either the default settings or the settings established in the application by the Set Workstation Window, Set Workstation Window 3, Set Workstation Viewport and Set Workstation Viewport 3 subroutines. The current values reflect the workstation’s current transformation values. As soon as the workstation is updated, the requested and current values are the same and the state is *NOT PENDING*. The values returned by the graPHIGS API are the window and viewport definitions.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **3** Function Requires State (PHOP,WSOP,*,*)
- **54** Specified Workstation Is Not Open
- **57** Specified Workstation Is Of Category MI

**Language Bindings**

### C

```
C
pinq_ws_tran (ws_id, err_ind, upd_st, req_win_lim, cur_win_lim, req_vp_lim, cur_vp_lim)
```

### FORTRAN

```
FORTRAN
PQWKST (wksta)
```

**Output Parameters**

- **Pws_st *ws_st**
  - Workstation state value (0=PWS_ST_WSCL, 1=PWS_ST_WSOP).

**Errors**

None

**Related Subroutines**

- Close Workstation
- Open Workstation

---

**INQUIRE WORKSTATION TRANSFORMATION (PHOP,WSOP,*,*)**

**Purpose**

Use Inquire Workstation Transformation to inquire the current and requested workstation transformation values of a specified workstation.

If your application has not updated the workstation, then the graPHIGS API returns a value of *PENDING*. In this case, the requested values reflect either the default settings or the settings established in the application by the Set Workstation Window, Set Workstation Window 3, Set Workstation Viewport and Set Workstation Viewport 3 subroutines. The current values reflect the workstation’s current transformation values. As soon as the workstation is updated, the requested and current values are the same and the state is *NOT PENDING*. The values returned by the graPHIGS API are the window and viewport definitions.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

- **3** Function Requires State (PHOP,WSOP,*,*)
- **54** Specified Workstation Is Not Open
- **57** Specified Workstation Is Of Category MI

**Language Bindings**

### C

```
C
pinq_ws_tran (ws_id, err_ind, upd_st, req_win_lim, cur_win_lim, req_vp_lim, cur_vp_lim)
```
Input Parameters

**Pint ws_id**

Workstation identifier.

Output Parameters

**Pint *err_ind**

Error indicator.

**Pupd_st *upd_st**

Workstation transformation update state (0=PUPD_NOT_PEND, 1=PUPD_PEND).

**Plimit *req_win_lim**

Requested workstation window limits in NPC.

**Plimit *cur_win_lim**

Current workstation window limits in NPC.

**Plimit *req_vp_lim**

Requested workstation viewport limits in DC.

**Plimit *cur_vp_lim**

Current workstation viewport limits in DC.

FORTRAN

**PQWKT (wkid, errind, tus, rwindo, cwindo, rviewp, cviewp)**

Input Parameters

**integer wkid**

Workstation identifier.

Output Parameters

**integer errind**

Error indicator.

**integer tus**

Workstation transformation update state (0=PNPEND, 1=PPEND).

**real rwindo(4)**

Requested workstation window limits in NPC (RWXMIN, RWXMAX, RWYMIN, RWYMAX).

**real cwindo(4)**

Current workstation window limits in NPC. (CWXMIN, CWXMAX, CWYMIN, CWYMAX).

**real rviewp(4)**

Requested workstation viewport limits in DC (RVXMIN, RVXMAX, RVYMIN, RVYMAX).

**real cviewp(4)**

Requested workstation viewport limits in DC (CVXMIN, CVXMAX, CVYMIN, CVYMAX).

Errors

None

Related Subroutines

- Inquire Workstation Transformation 3
- Set Workstation Viewport
- Set Workstation Viewport 3
• Set Workstation Window
• Set Workstation Window 3

INQUIRE WORKSTATION TRANSFORMATION 3 (PHOP,WSOP,*,*)

Purpose

Use Inquire Workstation Transformation 3 to inquire the current and requested workstation transformation values of a specified workstation.

If your application has not updated the workstation, then the graPHIGS API returns a value of PENDING. In this case, the requested values reflect either the default settings or the settings established in the application by the Set Workstation Window, Set Workstation Window 3, Set Workstation Viewport, and Set Workstation Viewport 3, subroutines. The current values reflect the workstation's current transformation values. As soon as the workstation is updated, the requested and current values are the same and the state is NOT PENDING. The values returned by the graPHIGS API are the window and viewport definitions.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the information is unavailable, then the values returned in the output parameters are unpredictable and the graPHIGS API sets the error indicator to one of the following errors:

3 Function Requires State (PHOP,WSOP,*;*)
54 Specified Workstation Is Not Open
57 Specified Workstation Is Of Category MI

Language Bindings

C

pinq_ws_tran3 (ws_id, err_ind, upd_st, req_win_lim, cur_win_lim, req_vp_lim, cur_vp_lim)

Input Parameters

Pint ws_id
    Workstation identifier.

Output Parameters

Pint *err_ind
    Error indicator.

Pupd_st *upd_st
    Workstation transformation update state (0=PUPD_NOT_PEND, 1=PUPD_PEND).

Plimit3 *req_win_lim
    Requested workstation window limits in NPC.

Plimit3 *cur_win_lim
    Current workstation window limits in NPC.

Plimit3 *req_vp_lim
    Requested workstation viewport limits in DC.

Plimit3 *cur_vp_lim
    Current workstation viewport limits in DC.
FORTRAN

PQWK3 (wkid, errind, tus, rwindo, cwindo, rviewp, cviewp)

Input Parameters

integer wkid
Workstation identifier.

Output Parameters

integer errind
Error indicator.

integer tus
Workstation transformation update state (0=PNPEND, 1=PPEND).

real rwindo(6)
Requested workstation window limits in NPC (RWXMIN, RWXMAX, RWYMIN, RWYMAX, RWZMIN, RWZMAX).

real cwindo(6)
Current workstation window limits in NPC (CWXMIN, CWXMAX, CWYMIN, CWYMAX, CWZMIN, CWZMAX).

real rviewp(6)
Requested workstation viewport limits in DC (RVXMIN, RVXMAX, RVYMIN, RVYMAX, RVZMIN, RVZMAX).

real cviewp(6)
Requested workstation viewport limits in DC (CVXMIN, CVXMAX, CVYMIN, CVYMAX, CVZMIN, CVZMAX).

Errors

None

Related Subroutines

• Set Workstation Viewport
• Set Workstation Viewport 3
• Set Workstation Window
• Set Workstation Window 3
Chapter 16. ISO PHIGS Transformations

All coordinate data in an ISO PHIGS implementation is conceptually manipulated as three-dimensional data. An application specifies a coordinate as an x, y, z triplet, or if the application specifies an x, y pair, z=0 is assumed. All points are then represented mathematically as column vectors as prescribed by the ISO PHIGS standard. Storage of transformation matrixes which are applied to these points or column vectors is defined by the individual ISO PHIGS bindings and described below.

3-by-3 Matrix

Let the elements of an ISO PHIGS 3x3 matrix be:

\[
\begin{pmatrix}
a & b & c \\
d & e & f \\
g & h & i \\
\end{pmatrix}
\]

C Binding

The ISO PHIGS C binding specifies that these elements be stored such that:

\[
\begin{align*}
m[0][0] &= a; & m[0][1] &= b; & m[0][2] &= c; \\
m[1][0] &= d; & m[1][1] &= e; & m[1][2] &= f; \\
m[2][0] &= g; & m[2][1] &= h; & m[2][2] &= i;
\end{align*}
\]

where \( m \) is of type \( Pmatrix \).

FORTRAN Binding

The ISO PHIGS FORTRAN binding specifies that these elements be stored such that:

\[
\begin{align*}
p[1,1] &= a; & p[2,1] &= b; & p[3,1] &= c \\
p[1,3] &= g; & p[2,3] &= h; & p[3,3] &= i \\
\end{align*}
\]

where \( p \) is a 3 X 3 real matrix.

4-by-4 Matrix

Let the elements of an ISO PHIGS 4 X 4 matrix be:

\[
\begin{pmatrix}
a & b & c & d \\
e & f & g & h \\
j & k & l & m \\
o & p & n & o \\
\end{pmatrix}
\]

C Binding

The ISO PHIGS C binding specifies that these elements be stored such that:

\[
\begin{align*}
q[0][0] &= a; & q[0][1] &= b; & q[0][2] &= c; & q[0][3] &= d; \\
q[1][0] &= e; & q[1][1] &= f; & q[1][2] &= g; & q[1][3] &= h; \\
q[2][0] &= i; & q[2][1] &= j; & q[2][2] &= k; & q[2][3] &= l; \\
q[3][0] &= m; & q[3][1] &= n; & q[3][2] &= o; & q[3][3] &= p;
\end{align*}
\]
where $q$ is of type $P_{matrix3}$.

**FORTRAN Binding**

The ISO PHIGS FORTRAN binding specifies that these elements be stored such that:

$$
\begin{align*}
  t[1,2] &= e & t[2,2] &= f & t[3,2] &= g & t[4,2] &= h \\
\end{align*}
$$

where $t$ is a $4 \times 4$ real matrix.
Chapter 17. FORTRAN Structure Content Data Records

This appendix contains the output parameters for structure content data records returned by the graPHIGS API for the ISO PHIGS FORTRAN subroutines.

The data is organized numerically by element type.

**PENIL=1: (Nil)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Nil*,
- **IL** = 0
- **IA** = ()
- **RL** = 0
- **RA** = ()
- **SL** = 0
- **LSTR** = ()
- **STR** = ()

**PEPL3=2: (Polyline 3)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Polyline 3*,
- **IL** = 1
- **IA(1)** = number of points in the polyline 3
- **RL** = 3*IA(1)
- **RA** = elements 1 through IA(1) contain the x components of the polyline 3
elements IA(1)+1 through 2*IA(1) contain the y components of the polyline 3elements 2*IA(1)+1 through 3*IA(1) contain the z components of the polyline 3
- **SL** = 0
- **LSTR** = ()
- **STR** = ()

**PEPL=3: (Polyline)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Polyline*,
- **IL** = 1
- **IA(1)** = number of points in the polyline
- **RL** = 2*IA(1)
- **RA** = elements 1 through IA(1) contain the x components of the polylineelements IA(1)+1 through 2*IA(1) contain the y components of the polyline
- **SL** = 0
- **LSTR** = ()
- **STR** = ()

**PEPM3=4: (Polymarker 3)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Polymarker 3*,
- **IL** = 1
- **IA(1)** = number of points in the polymarker 3
- **RL** = 3*IA(1)
• RA = elements 1 through IA(1) contain the x components of the polymarker
  elements IA(1)+1 through 2*IA(1) contain the y components of the polymarker
  elements 2*IA(1)+1 through 3*IA(1) contain the z components of the polymarker

  • SL = 0
  • LSTR = ()
  • STR = ()

PEPM=5: (Polymarker)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Polymarker,
• IL = 1
• IA(1) = number of points in the polymarker
• RL = 2*IA(1)
• RA = elements 1 through IA(1) contain the x components of the polymarker
  elements IA(1)+1 through 2*IA(1) contain the y components of the polymarker

  • SL = 0
  • LSTR = ()
  • STR = ()

PETX3=6: (Text 3)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Text 3,
• IL = 0
• IA = ()
• RL = 9
• RA(1) = x coordinate of text point (MC)
• RA(2) = y coordinate of text point (MC)
• RA(3) = z coordinate of text point (MC)
• RA(4) = x coordinate of first text direction vector (MC)
• RA(5) = y coordinate of first text direction vector (MC)
• RA(6) = z coordinate of first text direction vector (MC)
• RA(7) = x coordinate of second text direction vector (MC)
• RA(8) = y coordinate of second text direction vector (MC)
• RA(9) = z coordinate of second text direction vector (MC)

  • SL = 1
  • LSTR(1) = length of string
  • STR(1) = string

PETX=7: (Text)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Text,
• IL = 0
• IA = ()
• RL = 2
• RA(1) = x coordinate of text point (MC)
• RA(2) = y coordinate of text point (MC)

  • SL = 1
  • LSTR(1) = length of string
• $\text{STR}(1) = \text{string}$

PEATR3=8: (Annotation Text Relative 3)

Output parameters for $\text{STRUCTURE CONTENT DATA RECORD}$ for element type $\text{Annotation Text Relative 3}$,
- $\text{IL} = 0$
- $\text{IA} = ()$
- $\text{RL} = 6$
- $\text{RA}(1) = x$ coordinate of reference point (MC)
- $\text{RA}(2) = y$ coordinate of reference point (MC)
- $\text{RA}(3) = z$ coordinate of reference point (MC)
- $\text{RA}(4) = x$ coordinate of annotation point (NPC)
- $\text{RA}(5) = y$ coordinate of annotation point (NPC)
- $\text{RA}(6) = z$ coordinate of annotation point (NPC)
- $\text{SL} = 1$
- $\text{LSTR}(1) = \text{length of string}$
- $\text{STR}(1) = \text{string}$

PEATR=9: (Annotation Text Relative)

Output parameters for $\text{STRUCTURE CONTENT DATA RECORD}$ for element type $\text{Annotation Text Relative}$,
- $\text{IL} = 0$
- $\text{IA} = ()$
- $\text{RL} = 4$
- $\text{RA}(1) = x$ coordinate of reference point (MC)
- $\text{RA}(2) = y$ coordinate of reference point (MC)
- $\text{RA}(3) = x$ coordinate of annotation point (NPC)
- $\text{RA}(4) = y$ coordinate of annotation point (NPC)
- $\text{SL} = 1$
- $\text{LSTR}(1) = \text{length of string}$
- $\text{STR}(1) = \text{string}$

PEFA3=10: (Fill Area 3)

Output parameters for $\text{STRUCTURE CONTENT DATA RECORD}$ for element type $\text{Fill Area 3}$,
- $\text{IL} = 1$
- $\text{IA}(1) = \text{number of points in the fill area 3}$
- $\text{RL} = 3*\text{IA}(1)$
- $\text{RA} = \text{elements 1 through IA}(1)$ contain the $x$ components of the fill area 3 elements $\text{IA}(1)$ +1 through $2*\text{IA}(1)$ contain the $y$ components of the fill area 3 elements $2* \text{IA}(1)$+1 through $3*\text{IA}(1)$ contain the $z$ components of the fill area 3
- $\text{SL} = 0$
- $\text{LSTR} = ()$
- $\text{STR} = ()$

PEFA=11: (Fill Area)
Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Fill Area*,
- \( IL = 1 \)
- \( IA(1) = \) number of points in the fill area
- \( RL = 2 \times IA(1) \)
- \( RA = \) elements 1 through \( IA(1) \) contain the \( x \) components of the fill area
  elements \( IA(1) + 1 \) through \( 2 \times IA(1) \) contain the \( y \) components of the fill area
- \( SL = 0 \)
- \( LSTR = () \)
- \( STR = () \)

**PEFAS3=12: (Fill Area Set 3)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Fill Area Set 3*,
- \( IL = \) number of point lists in fill area set 3
- \( IA() = \) array of end indexes for point lists in fill area set 3
- \( RL = 3 \times (IA(IL)) \)
- \( RA = \) elements 1 through \( (IA(IL)) \) contain the \( x \) components of the fill area set 3
  elements \( IA(IL) + 1 \) through \( 2 \times (IA(IL)) \) contain the \( y \) components of the fill area set 3
  elements \( 2 \times IA(IL) + 1 \) through \( 3 \times (IA()) \) contain the \( z \) components of the fill area set 3
- \( SL = 0 \)
- \( LSTR = () \)
- \( STR = () \)

**PEFAS=13: (Fill Area Set)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Fill Area Set*,
- \( IL = \) number of point lists in fill area set
- \( IA() = \) array of end indexes for point lists in fill area set
- \( RL = 2 \times (IA(IL)) \)
- \( RA = \) elements 1 through \( (IA(IL)) \) contain the \( x \) components of the fill area set
  elements \( IA(IL) + 1 \) through \( 2 \times (IA(IL)) \) contain the \( y \) components of the fill area set set
- \( SL = 0 \)
- \( LSTR = () \)
- \( STR = () \)

**PECA3=14: (Cell Array 3)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Cell Array 3*,
- \( IL = 2 + (IA(1) \times IA(2)) \)
- \( IA(1) = \) \( x \) dimension of cell index array
- \( IA(2) = \) \( y \) dimension of cell index array
- \( IA(3) \) to \( IA(IA(1) \times IA(2) + 2) = \) cell index array in column major order (e.g., \( IA(3) = COLIA(1,1), IA(4) = COLIA(2,1), \ldots \))
- \( RL = 9 \)
- \( RA(1) = \) \( x \) coordinate of \( P \) (MC)
- \( RA(2) = \) \( y \) coordinate of \( P \) (MC)
- \( RA(3) = \) \( z \) coordinate of \( P \) (MC)
- \( RA(4) = \) \( x \) coordinate of \( Q \) (MC)
• RA(5) = y coordinate of Q (MC)
• RA(6) = z coordinate of Q (MC)
• RA(7) = x coordinate of R (MC)
• RA(8) = y coordinate of R (MC)
• RA(9) = z coordinate of R (MC)
• SL = 0
• LSTR = ()
• STR = ()

PECA=15: (Cell Array)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Cell Array,
• IL = 2+(IA(1)*IA(2))
• IA(1) = x dimension of cell index array
• IA(2) = y dimension of cell index array
• IA(3) to IA((IA(1)*IA(2))+2) = cell index array in column major order (e.g., IA(3) = COLIA(1,1), IA(4) = COLIA(2,1), ... )
• RL = 4
• RA(1) = x coordinate of P (MC)
• RA(2) = y coordinate of P (MC)
• RA(3) = x coordinate of Q (MC)
• RA(4) = y coordinate of Q (MC)
• SL = 0
• LSTR = ()
• STR = ()

PEGDP3=16: (Generalized Drawing Primitive 3)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Generalized Drawing Primitive 3,
• IL = 2
• IA(1) = number of points in the generalized drawing primitive 3
• IA(2) = generalized drawing primitive 3 identifier
• RL = 3*IA(1)
• RA = elements 1 through IA(1) contain the x components of the GDP3 point list elements IA(1) +1 through 2*IA(1) contain the y components of the GDP 3 point list elements 2*IA(1)+1 through 3*IA(1) contain the z components of the GDP 3 point list
• SL = number of 80 character data records (LDR for PGDP3 subroutine)
• LSTR(1) to LSTR(SL) = 80
• STR(1) to STR(SL) = GDP data records (DATREC(1) to DATREC(SL) for PGDP3 subroutine)

PEGDP=17: (Generalized Drawing Primitive)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Generalized Drawing Primitive,
• IL = 2
• IA(1) = number of points in the generalized drawing primitive
• IA(2) = generalized drawing primitive identifier
• $RL = 2^*IA(1)$
• $RA = \text{elements 1 through } IA(1) \text{ contain the } x \text{ components of the GDP point list}$
  $\text{elements } IA(1) + 1 \text{ through } 2^*IA(1) \text{ contain the } y \text{ components of the GDP point list}$
• $SL = \text{number of 80 character data records (LDR for PGDP subroutine)}$
• $LSTR(1) \text{ to } LSTR(SL) = 80$
• $STR(1) \text{ to } STR(SL) = \text{GDP data records (DATREC(1) to } DATREC(SL) \text{ for PGDP subroutine)}$

PEPLI=18: (Set Polyline Index)

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Polyline Index*,
• $IL = 1$
• $IA(1) = \text{polyline index}$
• $RL = 0$
• $RA = ()$
• $SL = 0$
• $LSTR = ()$
• $STR = ()$

PEPMI=19: (Set Polymarker Index)

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Polymarker Index*,
• $IL = 1$
• $IA(1) = \text{polymarker index}$
• $RL = 0$
• $RA = ()$
• $SL = 0$
• $LSTR = ()$
• $STR = ()$

PETXI=20: (Set Text Index)

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Text Index*,
• $IL = 1$
• $IA(1) = \text{text index}$
• $RL = 0$
• $RA = ()$
• $SL = 0$
• $LSTR = ()$
• $STR = ()$

PEII=21: (Set Interior Index)

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Interior Index*,
• $IL = 1$
• $IA(1) = \text{interior index}$
• $RL = 0$
• $RA = ()$
• $SL = 0$
• LSTR = ()
• STR = ()

**PEEDI=22: (Set Edge Index)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Edge Index*,
• IL = 1
• IA(1) = edge index
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

**PELN=23: (Set Linetype)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Linetype*,
• IL = 1
• IA(1) = linetype
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

**PELWSC=24: (Set Linewidth Scale Factor)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Linewidth Scale Factor*,
• IL = 0
• IA = ()
• RL = 1
• RA(1) = linewidth scale factor
• SL = 0
• LSTR = ()
• STR = ()

**PEPLCI=25: (Set Polyline Color Index)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Polyline Color Index*,
• IL = 1
• IA(1) = polyline color index
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()
PEMK=26: (Set Marker Type)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Marker Type,
- IL = 1
- IA(1) = marker type
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PEMKSC=27: (Set Marker Size Scale Factor)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Marker Size Scale Factor,
- IL = 0
- IA = ()
- RL = 1
- RA(1) = marker size scale factor
- SL = 0
- LSTR = ()
- STR = ()

PEPMCI=28: (Set Polymarker Color Index)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Polymarker Color Index,
- IL = 1
- IA(1) = polymarker color index
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PETXFN=29: (Set Text Font)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Text Font,
- IL = 1
- IA(1) = text font
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PETXPR=30: (Set Text Precision)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Text Precision,
Chapter 17. FORTRAN Structure Content Data Records

- IL = 1
- IA(1) = text precision(PSTRP, PCHARP, PSTRKP)
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

**PECHXP=31: (Set Character Expansion Factor)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Character Expansion Factor*,

- IL = 0
- IA = ()
- RL = 1
- RA(1) = character expansion factor
- SL = 0
- LSTR = ()
- STR = ()

**PECHSP=32: (Set Character Spacing)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Character Spacing*,

- IL = 0
- IA = ()
- RL = 1
- RA(1) = character spacing
- SL = 0
- LSTR = ()
- STR = ()

**PETXCI=33: (Set Text Color Index)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Text Color Index*,

- IL = 1
- IA(1) = text color index
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

**PECHH=34: (Set Character Height)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Character Height*,

- IL = 0
- IA = ()
- RL = 1
• RA(1) = character height
• SL = 0
• LSTR = ()
• STR = ()

PECHUP=35: (Set Character Up Vector)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Character Up Vector,
• IL = 0
• IA = ()
• RL = 2
• RA(1) = x component of character up vector
• RA(2) = y component of character up vector
• SL = 0
• LSTR = ()
• STR = ()

PETXP=36: (Set Text Path)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Text Path,
• IL = 1
• IA(1) = text path (PRIGHT,PLEFT,PUP,PDOWN)
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PETXAL=37: (Set Text Alignment)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Text Alignment,
• IL = 2
• IA(1) = horizontal text alignment (PAHNOR,PALEFT,PACENT,PARITE)
• IA(2) = vertical text alignment (PAVNOR,PATOP,PACAP,PAHALF,PABASE,PABOTT)
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEATCH=38: (Set Annotation Text Character Height)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Annotation Text Character Height,
• IL = 0
• IA = ()
• RL = 1
• RA(1) = annotation text character height
• SL = 0
• LSTR = ()
• STR = ()

PEATCU=39: (Set Annotation Text Character Up Vector)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Annotation Text Character Up Vector,
• IL = 0
• IA = ()
• RL = 2
• RA(1) = x component of annotation text character up vector
• RA(2) = y component of annotation text character up vector
• SL = 0
• LSTR = ()
• STR = ()

PEATP=40: (Set Annotation Text Path)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Annotation Text Path,
• IL = 1
• IA(1) = annotation text path (PRIGHT,PLEFT,PUP,PDOWN)
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEATAL=41: (Set Annotation Text Alignment)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Annotation Text Alignment,
• IL = 2
• IA(1) = horizontal text alignment (PAHNOR,PALEFT,PACENT,PARITE)
• IA(2) = vertical text alignment (PAVNOR, PATOP, PACAP, PAHALF, PABASE, PABOTT)
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEANST=42: (Set Annotation Style)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Annotation Style,
• IL = 1
• IA(1) = annotation style
• RL = 0
PEIS=43: (Set Interior Style)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Interior Style,

- IL = 1
- IA(1) = interior style (PHOLLO,PSOLID,PPATTR,PHATCH,PISEMP)
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PEISI=44: (Set Interior Style Index)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Interior Style Index,

- IL = 1
- IA(1) = interior style index
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PEICl=45: (Set Interior Color Index)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Interior Color Index,

- IL = 1
- IA(1) = interior color index
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

PEEDFG=46: (Set Edge Flag)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Edge Flag,

- IL = 1
- IA(1) = edge flag (POFF,PON)
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()
PEEDT=47: (Set Edgetype)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Edgetype,
• IL = 1
• IA(1) = edgetype
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEEWSC=48: (Set Edgewidth Scale Factor)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Edgewidth Scale Factor,
• IL = 0
• IA = ()
• RL = 1
• RA(1) = edgewidth scale factor
• SL = 0
• LSTR = ()
• STR = ()

PEEDCI=49: (Set Edge Color Index)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Edge Color Index,
• IL = 1
• IA(1) = edge color index
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEPA=50: (Set Pattern Size)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Pattern Size,
• IL = 0
• IA = ()
• RL = 2
• RA(1) = x component of pattern size (MC)
• RA(2) = y component of pattern size (MC)
• SL = 0
• LSTR = ()
• STR = ()

PEPRPV=51: (Set Pattern Reference Point and Vectors)
Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Pattern Reference Point and Vectors*,

- IL = 0
- IA = ()
- RL = 9
- RA(1) = x coordinate of pattern reference point (MC)
- RA(2) = y coordinate of pattern reference point (MC)
- RA(3) = z coordinate of pattern reference point (MC)
- RA(4) = x component of pattern reference vector 1 (MC) (i.e., DVX(1))
- RA(5) = y component of pattern reference vector 1 (MC) (i.e., DVY(1))
- RA(6) = z component of pattern reference vector 1 (MC) (i.e., DVZ(1))
- RA(7) = x component of pattern reference vector 2 (MC) (i.e., DVX(2))
- RA(8) = y component of pattern reference vector 2 (MC) (i.e., DVY(2))
- RA(9) = z component of pattern reference vector 2 (MC) (i.e., DVZ(2))
- SL = 0
- LSTR = ()
- STR = ()

**PEPARF=52: (Set Pattern Reference Point)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Pattern Reference Point*,

- IL = 0
- IA = ()
- RL = 2
- RA(1) = x coordinate of pattern reference point (MC)(i.e.RFX)
- RA(2) = y coordinate of pattern reference point (MC)(i.e.RFY)
- SL = 0
- LSTR = ()
- STR = ()

**PEADS=53: (Add Names To Set)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Add Names To Set*,

- IL = number of names in the set
- IA = array of name set elements
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

**PERES=54: (Remove Names From Set)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Remove Names From Set*,

- IL = number of names in the set
- IA = array of name set elements
• RL = 0
• RA =()
• SL = 0
• LSTR =()
• STR =()

**PEIASF=55: (Set Individual ASF)**

Output parameters for `STRUCTURE CONTENT DATA RECORD` for element type *Set Individual ASF*,

• IL = 2
• IA(1) = attribute identifier 
  (PLN, PLWSC, PPLCI, PMK, PMKSC, PPMCI, PTXFN, PTXPR, PCHXP, PCHSP, PTXCI, PIS, PISI, PICI, PEDFG, PEDIT, PEWSC, PEDCI)
• IA(2) = aspect source flag value (PBUNDL, PINDIV)
• RL = 0
• RA =()
• SL = 0
• LSTR =()
• STR =()

**PEHRID=56: (Set HLHSR Identifier)**

Output parameters for `STRUCTURE CONTENT DATA RECORD` for element type *Set HLHSR Identifier*,

• IL = 1
• IA(1) = HLHSR identifier
• RL = 0
• RA =()
• SL = 0
• LSTR =()
• STR =()

**PELMT3=57: (Set Local Transformation 3)**

Output parameters for `STRUCTURE CONTENT DATA RECORD` for element type *Set Local Transformation 3*,

• IL = 1
• IA(1) = composition type (PCPRE, PCPOST, PCREPL)
• RL = 16
• RA(1) = (1,1) component of local transformation matrix
• RA(2) = (2,1) component of local transformation matrix
• RA(3) = (3,1) component of local transformation matrix
• RA(4) = (4,1) component of local transformation matrix
• RA(5) = (1,2) component of local transformation matrix
• RA(6) = (2,2) component of local transformation matrix
• RA(7) = (3,2) component of local transformation matrix
• RA(8) = (4,2) component of local transformation matrix
• RA(9) = (1,3) component of local transformation matrix
• RA(10) = (2,3) component of local transformation matrix
• RA(11) = (3,3) component of local transformation matrix
• RA(12) = (4,3) component of local transformation matrix
• RA(13) = (1,4) component of local transformation matrix
• RA(14) = (2,4) component of local transformation matrix
• RA(15) = (3,4) component of local transformation matrix
• RA(16) = (4,4) component of local transformation matrix
• SL = 0
• LSTR = ()
• STR = ()

PELMT=58: (Set Local Transformation)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Local Transformation,
• IL = 1
• IA(1) = composition type (PCPRE,PCPOST,PCREPL)
• RL = 9
• RA(1) = (1,1) component of local transformation matrix
• RA(2) = (2,1) component of local transformation matrix
• RA(3) = (3,1) component of local transformation matrix
• RA(4) = (1,2) component of local transformation matrix
• RA(5) = (2,2) component of local transformation matrix
• RA(6) = (3,2) component of local transformation matrix
• RA(7) = (1,3) component of local transformation matrix
• RA(8) = (2,3) component of local transformation matrix
• RA(9) = (3,3) component of local transformation matrix
• SL = 0
• LSTR = ()
• STR = ()

PEGMT3=59: (Set Global Transformation 3)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Global Transformation 3,
• IL = 0
• IA = ()
• RL = 16
• RA(1) = (1,1) component of global transformation matrix
• RA(2) = (2,1) component of global transformation matrix
• RA(3) = (3,1) component of global transformation matrix
• RA(4) = (4,1) component of global transformation matrix
• RA(5) = (1,2) component of global transformation matrix
• RA(6) = (2,2) component of global transformation matrix
• RA(7) = (3,2) component of global transformation matrix
• RA(8) = (4,2) component of global transformation matrix
• RA(9) = (1,3) component of global transformation matrix
• RA(10) = (2,3) component of global transformation matrix
• RA(11) = (3,3) component of global transformation matrix
• RA(12) = (4,3) component of global transformation matrix
• RA(13) = (1,4) component of global transformation matrix
• RA(14) = (2,4) component of global transformation matrix
• RA(15) = (3,4) component of global transformation matrix
• RA(16) = (4,4) component of global transformation matrix

\[ v_{RA(11)} = (3,3) \]
\[ v_{RA(12)} = (4,3) \]
\[ v_{RA(13)} = (1,4) \]
\[ v_{RA(14)} = (2,4) \]
\[ v_{RA(15)} = (3,4) \]
\[ v_{RA(16)} = (4,4) \]

PF=SL=0
LSTR = ()
STR = ()

**PEGMT**=60: (Set Global Transformation)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Global Transformation,

• IL = 0
• IA = ()
• RL = 9
• RA(1) = (1,1) component of global transformation matrix
• RA(2) = (2,1) component of global transformation matrix
• RA(3) = (3,1) component of global transformation matrix
• RA(4) = (1,2) component of global transformation matrix
• RA(5) = (2,2) component of global transformation matrix
• RA(6) = (3,2) component of global transformation matrix
• RA(7) = (1,3) component of global transformation matrix
• RA(8) = (2,3) component of global transformation matrix
• RA(9) = (3,3) component of global transformation matrix
• SL = 0
• LSTR = ()
• STR = ()

**PEMCV3**=61: (Set Modeling Clipping Volume 3)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Set Modeling Clipping Volume 3,

• IL = 2
• IA(1) = modeling clipping operator
• IA(2) = number of modeling clipping half-spaces in list
• RL = 6*IA(2)
• FOR i = 0 TO IA(2)—1
• RA((6*i)+1) = x coordinate of point defining plane of half-space (MC)
• RA((6*i)+2) = y coordinate of point defining plane of half-space (MC)
• RA((6*i)+3) = z coordinate of point defining plane of half-space (MC)
• RA((6*i)+4) = dx component of normal vector defining the plane of half-space (MC)
• RA((6*i)+5) = dy component of normal vector defining the plane of half-space (MC)
• RA((6*i)+6) = dz component of normal vector defining the plane of half-space (MC)
• SL = 0
• LSTR = ()
• STR = ()

**PEMCV=62: (Set Modeling Clipping Volume)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Modeling Clipping Volume*,
- IL = 2
- IA(1) = modeling clipping operator
- IA(2) = number of modeling clipping half-spaces in list
- RL = 4*IA(2)
- FOR i = 0 TO IA(2)—1
  - RA((4*i)+1) = x coordinate of point defining plane of half-space (MC)
  - RA((4*i)+2) = y coordinate of point defining plane of half-space (MC)
  - RA((4*i)+3) = dx component of normal vector defining the plane of half-space (MC)
  - RA((4*i)+4) = dy component of normal vector defining the plane of half-space (MC)
- SL = 0
- LSTR = ()
- STR = ()

**PEMCLI=63: (Set Modeling Clipping Indicator)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Modeling Clipping Indicator*,
- IL = 1
- IA(1) = modeling clipping indicator (*PNCLIP*, *PCLIP*)
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

**PERMCV=64: (Restore Modeling Clipping Volume)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Restore Modeling Clipping Volume*,
- IL = 0
- IA = ()
- RL = 0
- RA = ()
- SL = 0
- LSTR = ()
- STR = ()

**PEVWI=65: (Set View Index)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set View Index*,
- IL = 1
- IA(1) = view index
- RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEEXST=66: (Execute Structure)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Execute Structure,
• IL = 1
• IA(1) = structure identifier
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PELB=67: (Label)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Label,
• IL = 1
• IA(1) = label identifier
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()

PEAP=68: (Application Data)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Application Data,
• IL = 0
• IA = ()
• RL = 0
• RA = ()
• SL = number of application data records (LDR for PAP subroutine)
• LSTR(1) to LSTR(SL) = 80
• STR(1) to STR(SL) = application data records (DATREC(1) to DATREC(SL) for PAP subroutine)

PEGSE=69: (Generalized Structure Element)

Output parameters for STRUCTURE CONTENT DATA RECORD for element type Generalized Structure Element,
• IL = 1
• IA(1) = generalized structure element identifier
• RL = 0
• RA = ()
• SL = number of GSE data records (LDR for PGSE subroutine)
• LSTR(1) to LSTR(SL) = 80
• STR(1) to STR(SL) = GSE data records (DATREC(1) to DATREC(SL) for PGSE subroutine)

**PEPKID=70: (Set Pick Identifier)**

Output parameters for *STRUCTURE CONTENT DATA RECORD* for element type *Set Pick Identifier*,
• IL = 1
• IA(1) = pick identifier
• RL = 0
• RA = ()
• SL = 0
• LSTR = ()
• STR = ()
Chapter 18. ISO PHIGS C Type and Macro Definitions

/* ----------------------------------------------------------------------------------*/
/* ISO PHIGS C binding type definitions and macro definitions */
/* ----------------------------------------------------------------------------------*/

/* Environmental type definitions */
typedef float Pfloat;
typedef int Pint;

/* Implementation dependent type definitions */

/*--Pchoice_data CHOICE DATA RECORD---------------------------------*/
typedef struct {
  union Pchoice_pets {
    struct Pchoice_pet_r1 {
      Pint number;  /* number of whatever */
    } pet_r1;
    struct Pchoice_pet_r2 {
      Pint num_prompts;  /* number of prompts */
      Ppr_switch *prompts;  /* array of prompts */
    } pet_r2;
    struct Pchoice_pet_r3 {
      Pint num_strings;  /* number of choice strings */
      char **strings;  /* array of choice strings */
    } pet_r3;
    struct Pchoice_pet_r4 {
      Pint num_strings;  /* number of choice strings */
      char **strings;  /* array of choice strings */
    } pet_r4;
    struct Pchoice_pet_r5 {
      Pint struct_id;  /* structure identifier */
      Pint num_pick_ids;  /* number of pick identifiers */
      Pint *pick_ids;  /* array of pick identifiers */
    } pet_r5;
  } pets;
} Pchoice_data;

/*--Pchoice_data3 CHOICE DATA RECORD 3---------------------------------*/
typedef struct {
  union Pchoice3_pets {
    struct Pchoice3_pet_r1 {
      Pint number;  /* number of whatever */
    } pet_r1;
    struct Pchoice3_pet_r2 {
      Pint num_prompts;  /* number of prompts */
      Ppr_switch *prompts;  /* array of prompts */
    } pet_r2;
    struct Pchoice3_pet_r3 {
      Pint num_strings;  /* number of choice strings */
      char **strings;  /* array of choice strings */
    } pet_r3;
  } pets;
} Pchoice_data3;
struct Pchoice3_pet_r4 {
    Pint num_strings;  /* number of choice strings */
    char **strings;    /* array of choice strings */
} pet_r4;

struct Pchoice3_pet_r5 {
    Pint struct_id;   /* struct identifier */
    Pint num_pick_ids; /* number of pick identifiers */
    Pint *pick_ids;   /* array of pick identifiers */
} pet_r5;

} pets;
}
}

Pchoice_data3;

/\*-Pcolr_rep  COLOR REPRESENTATION-----------------------------*/
typedef union {
    Prgb rgb;           /* Red Green Blue color specification */
    Pcieluv cieluv;     /* CIE L*U*V* color specification */
    Phls hls;           /* Hue Lightness Saturation color specification */
    Phsv hsv;           /* Hue Saturation Value color specification */
    Pdata unsupp;       /* Color in an unsupported color model */
} Pcolr_rep;

/\*-Pescape_in_data  ESCAPE INPUT DATA RECORD--------------------*/
typedef union {
    struct Pescape_in_r1 {
        char *string;       /* escape registration dependent */
    } escape_in_r1;
} Pescape_in_data;

/\*-Pescape_out_data  ESCAPE OUT DATA RECORD---------------------*/
typedef union {
    struct Pescape_out_r1 {
        char *string;       /* escape registration dependent */
    } escape_out_r1;
    struct Pescape_out_r2 {
        Pint ws_id;        /* workstation identifier */
        char *string;      /* escape output data record */
    } escape_out_r2;
} Pescape_out_data;

/\*-Pgdp_data  GDP DATA RECORD------------------------------------*/
typedef union {
    struct Pgdp_r1 {
        char *string;       /* registration dependent */
    } gdp_r1;
    Pdata unsupp;       /* unsupported GDP data record */
} Pgdp_data;

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
/*--Pgdp_data3   GDP DATA RECORD 3------------------------------------*/
typedef union {
    struct Pgdp3_r1 {
        char *string;        /* registration dependent  */
    } gdp3_r1;
    Pdata unsupp;        /* unsupported GDP3 data record  */
} Pgdp_data3;

/*--Pgse_data   GSE DATA RECORD---------------------------------------*/
typedef union {
    struct Pgse_r1 {
        char *string;        /* registration dependent  */
    } gse_r1;
    Pdata unsupp;        /* unsupported GSE data record  */
} Pgse_data;

/*--Pitem_data   ITEM DATA RECORD-------------------------------------*/
typedef union {
    char *string;        /* Metafile Records  */
    Pdata unsupp;        /* unsupported Metafile item data */
} Pitem_data;

/*--Ploc_data   LOCATOR DATA RECORD-----------------------------------*/
typedef struct {
    union Ploc_pets {
        struct Ploc_pet_r1 {
            Pint number;        /* number of whatever  */
        } pet_r1;
        struct Ploc_pet_r2 {
            Pint number;        /* number of whatever  */
        } pet_r2;
        struct Ploc_pet_r3 {
            Pint number;        /* number of whatever  */
        } pet_r3;
        struct Ploc_pet_r4 {
            Pline_attrs line_attrs;        /* polyline attributes  */
        } pet_r4;
        struct Ploc_pet_r5 {
            Pline_fill_ctrl_flag line_fill_ctrl_flag;        /* control flag  */
        } pet_r5;
    } Ploc_pets;
    union Plocattrs {
        PlineAttrs line_attrs;        /* polyline attributes  */
        Pint_attrs int_attrs;        /* interior attributes  */
    } Plocattrs;
} Ploc_data;
```c
struct Ploc_fill_set {
  Pint_attrs  int_attrs; /* interior attributes */
  Pedge_attrs edge_attrs; /* edge attributes */
} fill_set;

} attrs;
} pet_r5;

struct Ploc_pet_r6 {
  Pint number; /* number of whatever */
} pet_r6;

} pets;
} Ploc_data;

/---Ploc_data3  LOCATOR DATA RECORD 3----------------------------------*/

typedef struct {
  union Ploc3_pets {
    struct Ploc3_pet_r1 {
      Pint number; /* number of whatever */
    } pet_r1;

    struct Ploc3_pet_r2 {
      Pint number; /* number of whatever */
    } pet_r2;

    struct Ploc3_pet_r3 {
      Pint number; /* number of whatever */
    } pet_r3;

    struct Ploc3_pet_r4 {
      Pline_attrs line_attrs; /* polyline attributes */
    } pet_r4;

    struct Ploc3_pet_r5 {
      Pline_fill_ctrl_flag line_fill_ctrl_flag; /* control flag */
      union Ploc3_attrs {
        Pline_attrs line_attrs; /* polyline attributes */
        Pint_attrs  int_attrs; /* interior attributes */
      } attrs;
    } pet_r5;

    struct Ploc3_pet_r6 {
      Pint number; /* number of whatever */
    } pet_r6;

  } pets;
} Ploc_data3;

/---Ppick_data  PICK DATA RECORD-------------------------------------*/
typedef struct {
    union Ppick_pets {
        struct Ppick_pet_r1 {
            char *string; /* implementation dependent */
        } pet_r1;
    } pets;
} Ppick_data;

/*--Ppick_data3 PICK DATA RECORD 3-----------------------------*/
typedef struct {
    union Ppick3_pets {
        struct Ppick3_pet_r1 {
            char *string; /* implementation dependent */
        } pet_r1;
    } pets;
} Ppick_data3;

/*--Pstring_data STRING DATA RECORD-------------------------*/
typedef struct {
    Pint in_buf_size;
    Pint init_pos;
    union Pstring_pets {
        struct Pstring_pet_r1 {
            char **string; /* implementation dependent */
        } pet_r1;
    } pets;
} Pstring_data;

/*--Pstring_data3 STRING DATA RECORD 3----------------------*/
typedef struct {
    Pint in_buf_size;
    Pint init_pos;
    union Pstring3_pets {
        struct Pstring3_pet_r1 {
            char **string; /* implementation dependent */
        } pet_r1;
    } pets;
} Pstring_data3;

/*--Pstroke_data STROKE DATA RECORD------------------------*/
typedef struct {
    Pint in_buf_size;
    Pint init_pos;
    union Pstroke_pets {
    } pets;
} Pstroke_data;

Chapter 18. ISO PHIGS C Type and Macro Definitions 461
Pint    in_buf_size;    /* input buffer size */
Pint    init_pos;     /* initial editing position */
PFloat  x_interval;  /* x interval */
PFloat  y_interval;  /* y interval */
PFloat  time_interval;  /* time interval */

union Pstroke_pets {
    struct Pstroke_pet_r1 {
        char **string;    /* implementation dependent */
    } pet_r1;
    struct Pstroke_pet_r2 {
        char **string;    /* implementation dependent */
    } pet_r2;
    struct Pstroke_pet_r3 {
        Pmarker_attrs marker_attrs;    /* marker attributes */
    } pet_r3;
    struct Pstroke_pet_r4 {
        Pline_attrs Time_attrs;    /* line attributes */
    } pet_r4;
}
pets;

Pstroke_data;

/**--Pstroke_data3 STROKE DATA RECORD 3-----------------------------*/

typedef struct {
    Pint    in_buf_size;    /* input buffer size */
Pint    init_pos;     /* initial editing position */
PFloat  x_interval;  /* x interval */
PFloat  y_interval;  /* y interval */
PFloat  z_interval;  /* z interval */
PFloat  time_interval;  /* time interval */
union Pstroke3_pets {
    struct Pstroke3_pet_r1 {
        char **string;    /* implementation dependent */
    } pet_r1;
    struct Pstroke3_pet_r2 {
        char **string;    /* implementation dependent */
    } pet_r2;
    struct Pstroke3_pet_r3 {
        Pmarker_attrs marker_attrs;    /* marker attributes */
    } pet_r3;
    struct Pstroke3_pet_r4 {
        Pline_attrs Time_attrs;    /* line attributes */
    } pet_r4;
}
pets;
Pstroke_data3;

/**--Pval_data VALUATOR DATA RECORD----------------------------------*/
typedef struct {
    Pfloat low_value; /* low value of valuator range */
    Pfloat high_value; /* high value of valuator range*/
}
union Pval_pets {
    struct Pval_pet_r1 {
        char **string; /* implementation dependent */
    } pet_r1;
} pets;
} Pval_data;

/*--Pval_data3 VALUATOR DATA RECORD 3-----------------------------------*/
typedef struct {
    Pfloat low_value; /* low value of valuator range */
    Pfloat high_value; /* high value of valuator range*/
}
union Pval3_pets {
    struct Pval3_pet_r1 {
        char **string; /* implementation dependent */
    } pet_r1;
} pets;
} Pval_data3;

/* Implementation independent type definitions */

/*--Par_file ARCHIVE FILE----------------------------------------*/
typedef struct {
    Pint id; /* archive file identifier */
    char *name; /* archive file name */
} Par_file;

/*--Par_file_list ARCHIVE FILE LIST--------------------------------*/
typedef struct {
    Pint num_ar_files; /* number of archive files */
    Par_file *ar_files; /* list of archive files */
} Par_file_list;

/*--Par_st ARCHIVE STATE----------------------------------------*/
typedef enum {
    PST_ARCL
    PST_AROP
} Par_st;

/*--Pasf ASPECT SOURCE FLAG--------------------------------------*/
typedef enum {
typedef enum {
PASPECT_LINETYPE
PASPECT_LINEWIDTH
PASPECT_LINE_COLR_IND
PASPECT_MARKER_TYPE
PASPECT_MARKER_SIZE
PASPECT_MARKER_COLR_IND
PASPECT_TEXT_FONT
PASPECT_TEXT_PREC
PASPECT_CHAR_EXPAN
PASPECT_CHAR_SPACE
PASPECT_TEXT_COLR_IND
PASPECT_INT_STYLE
PASPECT_INT_STYLE_IND
PASPECT_INT_COLR_IND
PASPECT_EDGE_FLAG
PASPECT_EDGETYPE
PASPECT_EDGEWIDTH
PASPECT_EDGE_COLR_IND
} Paspect;

typedef enum {
PATTR_LINE
PATTR_MARKER
PATTR_TEXT
PATTR_INT
PATTR_EDGE
} Pattrs;

typedef struct {
Pfloat cieluv_x; /* x coefficient */
Pfloat cieluv_y; /* y coefficient */
Pfloat cieluv_y_lum; /* y luminance */
} Pcieluv;

typedef enum {
PIND_NO_CLIP
PIND_CLIP
} Pclip_ind;

typedef enum {
PAVAIL_MONOCHR
PAVAIL_COLR
} Pcolr_avail;

typedef enum {
Pcolr_facs
} Pcolr_avail;

#ifdef PASF_BUNDLED
PASF_INDIV
#endif

/**--Paspect  ASPECT----------------------------------------*/

typedef enum {
PASPECT_LINETYPE
PASPECT_LINEWIDTH
PASPECT_LINE_COLR_IND
PASPECT_MARKER_TYPE
PASPECT_MARKER_SIZE
PASPECT_MARKER_COLR_IND
PASPECT_TEXT_FONT
PASPECT_TEXT_PREC
PASPECT_CHAR_EXPAN
PASPECT_CHAR_SPACE
PASPECT_TEXT_COLR_IND
PASPECT_INT_STYLE
PASPECT_INT_STYLE_IND
PASPECT_INT_COLR_IND
PASPECT_EDGE_FLAG
PASPECT_EDGETYPE
PASPECT_EDGEWIDTH
PASPECT_EDGE_COLR_IND
} Paspect;

/**--Pattrs  ATTRIBUTES USED--------------------------------*/

typedef enum {
PATTR_LINE
PATTR_MARKER
PATTR_TEXT
PATTR_INT
PATTR_EDGE
} Pattrs;

/**--Pcieluv  CIE L*U*V*----------------------------------*/

typedef struct {
Pfloat cieluv_x; /* x coefficient */
Pfloat cieluv_y; /* y coefficient */
Pfloat cieluv_y_lum; /* y luminance */
} Pcieluv;

/**--Pclip_ind  CLIPPING INDICATOR------------------------*/

typedef enum {
PIND_NO_CLIP
PIND_CLIP
} Pclip_ind;

/**--Pcolr_avail  COLOR AVAILABILITY-----------------------*/

typedef enum {
PAVAIL_MONOCHR
PAVAIL_COLR
} Pcolr_avail;

/**--Pcolr_facs  COLOR FACILITIES-------------------------*/

464  The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
typedef struct {
    Pint num_colrs; /* number of colors */
    Pcolr_avail colr_avail; /* color availability */
    Pint num_pred_inds; /* number of predefined color indexes */
    Pcieluv prim_colrs[3]; /* primary colors */
} Pcolr_facs;

/*--Pcompose_type COMPOSITION TYPE-----------------------------*/
typedef enum {
    PTYPE_PRECONCAT
    PTYPE_POSTCONCAT
    PTYPE_REPLACE
} Pcompose_type;

/*--Pconf_res CONFLICT RESOLUTION-------------------------------*/
typedef enum {
    PRES_MAINTAIN
    PRES_ABANDON
    PRES_UPD
} Pconf_res;

/*--Pctrl_flag CONTROL FLAG-------------------------------------*/
typedef enum {
    PFLAG_COND
    PFLAG_ALWAYS
} Pctrl_flag;

/*--Pdata DATA------------------------------------------------*/
typedef struct {
    size_t size; /* size of data */
    void* data; /* pointer to data */
} Pdata;

/*--Pdc_units DEVICE COORDINATE UNITS--------------------------*/
typedef enum {
    PDC_METRES
    PDC_OTHER
} Pdc_units;

/*--Pdefer_mode DEFERRAL MODE------------------------------------*/
typedef enum {
    PDEFER_ASAP
    PDEFER_BNIG
    PDEFER_BNIL
    PDEFER_ASTI
    PDEFER_WAIT
} Pdefer_mode;

/*--Pdisp_space_size DISPLAY SPACE SIZE-------------------------*/
typedef struct {
  Pdc_units  dc_units;  /* device coordinate units  */
  Pfloat_size size_dc;  /* device size in coordinate units */
  Pint_size   size_raster; /* device size in raster units */
} Pdisp_space_size;

/**--Pdisp_space_size3  DISPLAY SPACE SIZE 3--------------------------*/
typedef struct {
  Pdc_units  dc_units;  /* device coordinate units  */
  Pfloat_size3 size_dc;  /* device volume in coordinate units*/
  Pint_size3  size_raster; /* device volume in raster units */
} Pdisp_space_size3;

/**--Pdisp_surf_empty  DISPLAY SURFACE EMPTY--------------------------*/
typedef enum {
  PSURF_NOT_EMPTY
  PSURF_EMPTY
} Pdisp_surf_empty;

/**--Pdysn_structs  DYNAMICS OF STRUCTURES--------------------------*/
typedef struct {
  Pdyn_mod     content;  /* structure content  */
  Pdyn_mod     post;    /* post structure  */
  Pdyn_mod     unpost;  /* unpost structure  */
  Pdyn_mod     del;     /* delete structures */
  Pdyn_mod     ref;     /* structure reference */
} Pdysn_structs;

/**--Pdysn_ws_attr  DYNAMICS OF WORKSTATION ATTRIBUTES--------------*/
typedef struct {
  Pdyn_mod    line_bundle; /* polyline bundle representation */
  Pdyn_mod    marker_bundle; /* polymarker bundle representation*/
  Pdyn_mod    text_bundle;  /* text bundle representation */
  Pdyn_mod    int_bundle;   /* interior bundle representation */
  Pdyn_mod    edge_bundle;  /* edge bundle representation */
  Pdyn_mod    pat_rep;      /* pattern representation */
  Pdyn_mod    colr_rep;     /* color representation */
  Pdyn_mod    view_rep;     /* view representation */
  Pdyn_mod    ws_tran;      /* workstation transform */
  Pdyn_mod    highl_filter; /* highlight filter */
  Pdyn_mod    invis_filter; /* invisibility filter */
  Pdyn_mod    hlhsr_mode;   /* HLHSR mode */
} Pdysn_ws_attr;

/**--Pdyn_mod  DYNAMIC MODIFICATION-----------------------------------*/
typedef enum {
  PDYN_IRG
  PDYN_IMM
  PDYN_CBS
};
typedef enum
{
  PSWITCH_NO_ECHO,
  PSWITCH_ECHO
}
Pecho_switch;

typedef struct
{
  Pasf flag_asf;  /* edge flag asf */
  Pasf type_asf;  /* edge type asf */
  Pasf width_asf; /* edge width asf */
  Pasf colr_ind_asf; /* edge color index asf */
  Pint ind;       /* edge index */
  Pedge_bundle bundle; /* edge bundle */
}
Pedge_attrs;

typedef struct
{
  Pedge_flag flag; /* edge flag */
  Pint type;     /* edgetype */
  Pfloat width;  /* edgewidth scale factor */
  Pint colr_ind; /* edge color index */
}
Pedge_bundle;

typedef struct
{
  Pint_list types; /* list of edge types */
  Pint num_widths; /* number of available edge widths */
  Pfloat nom_width; /* nominal edge width */
  Pfloat min_width; /* min edge width */
  Pfloat max_width; /* max edge width */
  Pint num_pred_inds; /* number of predefined bundle indexes */
}
Pedge_facs;

typedef union
{
  Pint int_data;     /* integer valued data */
  Pfloat float_data; /* float valued data */
  Ppoint_list3 point_list3; /* list of 3d points */
  Ppoint_list point_list; /* list of 2d points */
  Ppoint_list_list3 point_list_list3; /* list of 3d point lists */
  Ppoint_list_list point_list_list; /* list of 2d point lists */
}

struct Pelem_text3
{
  Ppoint3 pos;  /* text position */
  Pvec3 dir[2]; /* direction vectors */
  char *char_string; /* char string */
}
text3;
struct Pelem_text {
    Ppoint pos;     /* text position */
    char  *char_string; /* character string */
} text;

struct Pelem_annot_text_rel3 {
    Ppoint3 ref_point;  /* reference point */
    Pvec3  offset;    /* annotation offset */
    char  *char_string; /* character string */
} anno_text_rel3;

struct Pelem_annot_text_rel {
    Ppoint ref_point;  /* reference point */
    Pvec  offset;     /* annotation offset */
    char  *char_string; /* character string */
} anno_text_rel;

struct Pelem_cell_array3 {
    Pparal paral;    /* parallelogram */
    Ppat_rep colr_array; /* color array */
} cell_array3;

struct Pelem_cell_array {
    Prect  rect;     /* rectangle */
    Ppat_rep colr_array; /* color array */
} cell_array;

struct Pelem_gdp3 {
    Pint id;             /* GDP3 id */
    Ppoint_list3 point_list; /* point list */
    Pgdp_data3 data;     /* data record */
} gdp3;

struct Pelem_gdp {
    Pint id;             /* GDP id */
    Ppoint_list point_list; /* point list */
    Pgdp_data data;     /* data record */
} gdp;

Ptext_prec  text_prec;    /* text precision */
Pvec  char_up_vec; /* character up vector */
Ptext_path  text_path;  /* text path */
Ptext_align text_align; /* text alignment */
Pint_style  int_style; /* interior style */
Pedge_flag  edge_flag; /* edge flag */
Ppoint  pat_ref_point;  /* pattern refer. point */
Pfloat_size  pat_size; /* pattern size */

struct Pelem_pat_ref_point_vecs {
    Ppoint3 ref_point;  /* pattern refer. point */
    Pvec3  ref_vec[2]; /* vectors */
} pat_ref_point_vecs;

Pint_list  names;        /* name sets */

struct Pelem_asf {
    Paspect id;        /* attribute id */
    Pasf  source;      /* attribute source */
} asf;

struct Pelem_local_tran3 {
    Pcompose_type compose_type; /* composition type */
    Pmatrix3  matrix;      /* matrix */
} local_tran3;

struct Pelem_local_tran {
    Pcompose_type compose_type; /* composition type */
} local_tran
Pmatrix matrix; /* matrix */

Pmatrix3 global_tran3; /* global transform3 */
Pmatrix global_tran; /* global transform */

struct Pelem_model_clip3 {
    Pint op; /* operator */
    Phalf_space_list3 half_spaces; /* half space list */
} model_clip3;

struct Pelem_model_clip {
    Pint op; /* operator */
    Phalf_space_list half_spaces; /* half space list */
} model_clip;

Pclip_ind clip_ind; /* clipping indicator */
Pdata appl_data; /* application data */

struct Pelem_gse {
    Pint id; /* GSE id */
    Pgse_data data; /* GSE data record */
} gse;

Pelem_data;

/*---Pelem_ref ELEMENT REFERENCE----------------------------------------*/
typedef struct {
    Pint struct_id; /* structure id */
    Pint elem_pos; /* element position */
} Pelem_ref;

/*---Pelem_ref_list ELEMENT REFERENCE LIST-------------------------------*/
typedef struct {
    Pint num_elem_refs; /* number of element references */
    Pelem_ref *elem_refs; /* list of element references */
} Pelem_ref_list;

/*---Pelem_ref_list_list ELEMENT REFERENCE LIST LIST---------------------*/
typedef struct {
    Pint num_elem_ref_lists; /* no. of element reference lists*/
    Pelem_ref_list *elem_ref_lists; /* list of element refer. lists */
} Pelem_ref_list_list;

/*---Pelem_type ELEMENT TYPE---------------------------------------------*/
typedef enum {
    PELEM_ALL,
    PELEM_NIL,
    PELEM_POLYLINE3,
    PELEM_POLYLINE,
    PELEM_POLYMARKER3,
    PELEM_POLYMARKER,
    PELEM_TEXT3,
    PELEM_TEXT
} Pelem_type;

Chapter 18. ISO PHIGS C Type and Macro Definitions 469
The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
typedef struct {
    Pint num_elem_types;  /* number of element types */
Pelem_type *elem_types;  /* list of element types */
} Pelem_type_list;

/*--Perr_mode  ERROR MODE--------------------------------------------*/
typedef enum {
    PERR_OFF
    PERR_ON
} Perr_mode;

/*--Pfilter  FILTER--------------------------------------------------*/
typedef struct {
    Pint_list incl_set;  /* inclusion set */
Pint_list excl_set;  /* exclusion set */
} Pfilter;

/*--Pfilter_list  FILTER LIST----------------------------------------*/
typedef struct {
    Pint num_filters;  /* number of filters */
Pfilter *filters;  /* list of filters */
} Pfilter_list;

/*--Pfloat_size  FLOAT SIZE------------------------------------------*/
typedef struct {
    Pfloat size_x;  /* x size */
Pfloat size_y;  /* y size */
} Pfloat_size;

/*--Pfloat_size3  FLOAT SIZE 3---------------------------------------*/
typedef struct {
    Pfloat size_x;  /* x size */
Pfloat size_y;  /* y size */
Pfloat size_z;  /* z size */
} Pfloat_size3;

/*--Pgse_id_dep  GSE IDENTIFIER DEPENDENCY--------------------------*/
typedef struct {
    Pint id;  /* GSE identifier */
Pws_dep_ind ind;  /* WS independent/dependent indicator */
} Pgse_id_dep;
typedef struct {
    Pint  num_id_facs;  /* # of identifiers/dependency element*/
    Pgse_id_dep  *id_facs;  /* list of GSE facilities */
} Pgse_id_dep_list;

typedef struct {
    Ppoint  point;  /* point */
    Pvec   norm;  /* normal */
} Phalf_space;

typedef struct {
    Ppoint3  point;  /* point */
    Pvec3   norm;  /* normal */
} Phalf_space3;

typedef struct {
    Pint  num_half_spaces;  /* number of half spaces */
    Phalf_space  *half_spaces;  /* list of half spaces */
} Phalf_space_list;

typedef struct {
    Pint  num_half_spaces;  /* number of half spaces */
    Phalf_space3  *half_spaces;  /* list of half spaces */
} Phalf_space_list3;

typedef struct {
    Pfloat  hue;  /* hue */
    Pfloat  lightness;  /* lightness */
    Pfloat  satur;  /* saturation */
} Phls;

typedef enum {
    PHOR_NORM
    PHOR_LEFT
    PHOR_CTR
    PHOR_RIGHT
} Phor_text_align;
typedef struct {
    Pfloat hue;        /* hue */
    Pfloat satur;      /* saturation */
    Pfloat value;      /* value */
} Phsv;

/*--Pinq_type INQUIRE TYPE------------------------------------------*/
typedef enum {
    PINQ_SET
    PINQ_REALIZED
} Pinq_type;

/*--Pint_attrs INTERIOR ATTRIBUTES----------------------------------*/
typedef struct {
    Pasf style_asf;     /* interior asf */
    Pasf style_ind_asf; /* interior style asf */
    Pasf colr_ind_asf;  /* interior color index asf */
    Pint ind;          /* interior index */
    PintBundle bundle; /* interior bundle */
} Pint_attrs;

/*--Pint_bundle INTERIOR BUNDLE-------------------------------------*/
typedef struct {
    Pint_style style;  /* interior style */
    Pint style_ind;    /* interior style index */
    Pint colr_ind;     /* interior color index */
} Pint_bundle;

/*--Pint_facs INTERIOR FACILITIES-----------------------------------*/
typedef struct {
    Pint num_int_styles; /* number of interior styles */
    Pint_style int_styles[5]; /* list of available interior styles */
    PintList hatch_styles; /* list of available hatch styles */
    Pint num_pred_inds; /* no. of predefined bundle indexes */
} Pint_facs;

/*--Pint_list INTEGER LIST------------------------------------------*/
typedef struct {
    Pint num_ints; /* number of Pints in list */
    Pint *ints;   /* list of integers */
} Pint_list;
typedef struct {
    Pint size_x; /* x size */
    Pint size_y; /* y size */
} Pint_size;

typedef struct {
    Pint size_x; /* x size */
    Pint size_y; /* y size */
    Pint size_z; /* z size */
} Pint_size3;

typedef enum {
PSTYLE_HOLLOW
PSTYLE_SOLID
PSTYLE_PAT
PSTYLE_HATCH
PSTYLE_EMPTY
} Pint_style;

typedef enum {
PIN_NONE
PIN_LOC
PIN_STROKE
PIN_VAL
PIN_CHOICE
PIN_PICK
PIN_STRING
} Pin_class;

typedef enum {
PIN_STATUS_NONE
PIN_STATUS_OK
PIN_STATUS_NO_IN
} Pin_status;

typedef struct {
Pfloat x_min; /* x min */
Pfloat x_max; /* x max */
Pfloat y_min; /* y min */
Pfloat y_max; /* y max */
} Plimit;
typedef struct {
    Pfloat x_min; /* x min */
    Pfloat x_max; /* x max */
    Pfloat y_min; /* y min */
    Pfloat y_max; /* y max */
    Pfloat z_min; /* z min */
    Pfloat z_max; /* z max */
} Plimit3;

/*--Pline attrs  POLYLINE ATTRIBUTES-----------------------------*/
typedef struct {
    Pasf type_asf; /* line type asf */
    Pasf width_asf; /* line width asf */
    Pasf colr_ind_asf; /* line color index asf */
    Pint ind; /* line index */
    Pline_bundle bundle; /* line bundle */
} Pline_attrs;

/*--Pline_bundle  POLYLINE BUNDLE---------------------------------*/
typedef struct {
    Pint type; /* line type */
    Pfloat width; /* linewidth scale factor */
    Pint colr_ind; /* color index */
} Pline_bundle;

/*--Pline_facs  POLYLINE FACILITIES-----------------------------*/
typedef struct {
    Pint_list types; /* list of line types */
    Pint num_widths; /* number of available linewidths */
    Pfloat nom_width; /* nominal linewidth */
    Pfloat min_width; /* min linewidth */
    Pfloat max_width; /* max linewidth */
    Pint num_pred_inds; /* no. of predefined bundle indexes*/
} Pline_facs;

/*--Pline_fill_ctrl_flag  POLYLINE FILL CONTROL FLAG-------------*/
typedef enum {
    PFLAG_LINE
    PFLAG_FILL
    PFLAG_FILL_SET
} Pline_fill_ctrl_flag;

/*--Pmarker_attrs  MARKER ATTRIBUTES-----------------------------*/
typedef struct {
    Pasf type_asf; /* marker type asf */
    Pasf size_asf; /* marker style asf */
    Pasf colr_ind_asf; /* marker color index asf */
    Pint ind; /* marker index */
    Pmarker_bundle bundle; /* marker bundle */
} Pmarker_attrs;
typedef struct {
  Pint type;  /* marker type */
  Pfloat size;  /* marker size scale factor */
  Pint colr_ind;  /* color index */
} Pmarker_bundle;

typedef struct {
  Pint_list types;  /* list of marker types */
  Pint num_sizes;  /* number of available marker sizes */
  Pfloat nom_size;  /* nominal marker size */
  Pfloat min_size;  /* min marker size */
  Pfloat max_size;  /* max marker size */
  Pint num_pred_inds;  /* number of predefined bundle indexes */
} Pmarker_facs;

typedef Pfloat Pmatrix[3][3];

typedef Pfloat Pmatrix3[4][4];

typedef enum {
  PMODE_NIVE,
  PMODE_UWOR,
  PMODE_UQUM
} Pmod_mode;

typedef enum {
  PSIMULT_NO_MORE,
  PSIMULT_MORE
} Pmore_simult_events;

typedef struct {
  Pint loc;  /* locators */
  Pint stroke;  /* strokes */
  Pint val;  /* valuators */
  Pint choice;  /* choices */
  Pint pick;  /* picks */
  Pint string;  /* strings */
} Pnum_in;
/ *-- Popen_struct_status  OPEN STRUCTURE STATUS-----------------------*/

typedef enum {
    PSTRUCT_NONE,
    PSTRUCT_OPEN
} Popen_struct_status;

/ *-- Pop_mode  OPERATING MODE----------------------------------------*/

typedef enum {
    POP_REQ,
    POP_SAMPLE,
    POP_EVENT
} Pop_mode;

/ *-- Pparal  PARALLELOGRAM------------------------------------------*/

typedef struct {
    Ppoint3  p;            /* point p */
    Ppoint3  q;            /* point q */
    Ppoint3  r;            /* point r */
} Pparal;

/ *-- Ppath_order  PATH ORDER----------------------------------------*/

typedef enum {
    PORDER_TOP_FIRST,
    PORDER_BOTTOM_FIRST
} Ppath_order;

/ *-- Ppat_rep  PATTERN REPRESENTATION-------------------------------*/

typedef struct {
    Pint_size  dims;        /* pattern's dimensions */
    Pint      *colr_array;  /* color index array */
} Ppat_rep;

/ *-- Ppick_path  PICK PATH------------------------------------------*/

typedef struct {
    Pint      depth;        /* pick path depth */
    Ppick_path_elem  *path_list;  /* pick path list */
} Ppick_path;

/ *-- Ppick_path_elem  PICK PATH ELEMENT-------------------------------*/

typedef struct {
    Pint      struct_id;    /* structure identifier */
    Pint      pick_id;      /* pick identifier */
    Pint      elem_pos;     /* element sequence number */
} Ppick_path_elem;

Chapter 18. ISO PHIGS C Type and Macro Definitions 477
typedef struct {
    Pfloat x;    /* x coordinate */
    Pfloat y;    /* y coordinate */
} Ppoint;

typedef struct {
    Pfloat x;    /* x coordinate */
    Pfloat y;    /* y coordinate */
    Pfloat z;    /* z coordinate */
} Ppoint3;

typedef struct {
    Pint num_points;    /* number of Ppoints in the list */
    Ppoint *points;     /* list of points */
} Ppoint_list;

typedef struct {
    Pint num_points;    /* number of Ppoint3s in the list */
    Ppoint3 *points;    /* list of points */
} Ppoint_list3;

typedef struct {
    Pint num_point_lists;    /* number of point lists */
    Ppoint_list *point_lists;    /* list of point lists */
} Ppoint_list_list;

typedef struct {
    Pint num_point_lists;    /* number of point lists */
    Ppoint_list3 *point_lists;    /* list of point lists */
} Ppoint_list_list3;

typedef struct {
    Pint num_points;    /* number of Ppoints in the list */
    Ppoint *points;     /* list of points */
} Ppoint_list;

/*--Pposted_struct POSTED STRUCTURE---------------------------------*/
typedef struct {

} Pposted_struct;
Pint id;  /* structure id */
Pfloat disp_pri;  /* display priority */

} Pposted_struct;

/**--Pposted_struct_list POSTED STRUCTURE LIST------------------------*/
typedef struct {
  Pint num_postings;  /* number of structure postings */
  Pposted_struct *postings;  /* list of postings */
} Pposted_struct_list;

/**--Pproj_type PROJECTION TYPE--------------------------------------*/
typedef enum {
  PTYPE_PARAL
  PTYPE_PERSPECT
} Pproj_type;

/**--PPr_switch PROMPT SWITCH----------------------------------------*/
typedef enum {
  PPR_OFF
  PPR_ON
} Ppr_switch;

/**--Prect RECTANGLE--------------------------------------------------*/
typedef struct {
  Ppoint p;  /* point p */
  Ppoint q;  /* point q */
} Prect;

/**--Pref_flag REFERENCE FLAG----------------------------------------*/
typedef enum {
  PFLAG_DEL
  PFLAG_KEEP
} Pref_flag;

/**--Pregen_flag REGENERATION FLAG-----------------------------------*/
typedef enum {
  PFLAG_POSTPONE
  PFLAG_PERFORM
} Pregen_flag;

/**--PPri Relative PRIORITY------------------------------------------*/
typedef enum {
  PPRI_HIGHER
  PPRI_LOWER
} Prel_pri;

/**--Prgb RED GREEN BLUE---------------------------------------------*/
typedef struct {
    Pfloat red;
    Pfloat green;
    Pfloat blue;
} Prgb;

/**--Psearch_dir SEARCH DIRECTION---------------------------------*/
typedef enum {
    PDIR_BACKWARD
    PDIR_FORWARD
} Psearch_dir;

/**--Psearch_status SEARCH STATUS--------------------------------*/
typedef enum {
    PSEARCH_STATUS_FAILURE
    PSEARCH_STATUS_SUCCESS
} Psearch_status;

/**--Pstore STORE------------------------------------------------*/
typedef void *Pstore;

/**--Pstruct_net_source STRUCTURE NETWORK SOURCE-------------------*/
typedef enum {
    PNET_CSS
    PNET_AR
} Pstruct_net_source;

/**--Pstruct_st STRUCTURE STATE----------------------------------*/
typedef enum {
    PSTRUCT_ST_STCL
    PSTRUCT_ST_STOP
} Pstruct_st;

/**--Pstruct_status STRUCTURE STATUS-------------------------------*/
typedef enum {
    PSTRUCT_STATUS_NON_EXISTENT
    PSTRUCT_STATUS_EMPTY
    PSTRUCT_STATUS_NOT_EMPTY
} Pstruct_status;

/**--Psys_st SYSTEM STATE------------------------------------------*/
typedef enum {
    PSYS_ST_PHCL
    PSYS_ST_PHOP
} Psys_st;

/**--Ptext_align TEXT ALIGNMENT------------------------------------*/
typedef struct {

480  The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference
Phor_text_align hor; /* horizontal component */
Pvert_text_align vert; /* vertical component */

} Ptext_align;

/*--Ptext_bundle TEXT BUNDLE-----------------------------------------------*/
typedef struct {
    Pint font;         /* text font */
    Ptext_prec prec;   /* text precision */
    Pfloat char_expan; /* char expansion factor */
    Pfloat char_space; /* character spacing */
    Pint colr_ind;     /* text color index */
} Ptext_bundle;

/*--Ptext_facs TEXT FACILITIES--------------------------------------------*/
typedef struct {
    Pint num_font_precs;        /* number of fonts and precisions */
    Ptext_font_prec *font_precs; /* list of fonts and precisions */
    Pint num_char_hts;          /* number of character heights */
    Pfloat min_char_ht;         /* minimum height */
    Pfloat max_char_ht;         /* maximum height */
    Pint num_char_expans;       /* # of character expansion factors */
    Pfloat min_char_expan;      /* minimum expansion factor */
    Pfloat max_char_expan;      /* maximum expansion factor */
    Pint num_pred inds;         /* no. of predefined bundle indexes */
} Ptext_facs;

/*--Ptext_font_prec TEXT FONT AND PRECISION-------------------------------*/
typedef struct {
    Pint font; /* text font */
    Ptext_prec prec; /* text precision */
} Ptext_font_prec;

/*--Ptext_path TEXT PATH-----------------------------------------------*/
typedef enum {
    PPATH_RIGHT
    PPATH_LEFT
    PPATH_UP
    PPATH_DOWN
} Ptext_path;

/*--Ptext_prec TEXT PRECISION-----------------------------------------------*/
typedef enum {
    PREC_STRING
    PREC_CHAR
    PREC_STROKE
} Ptext_prec;

/*--Pupd_st UPDATE STATE-----------------------------------------------*/
typedef enum {

PUPD_NOT_PEND
PUPD_PEND
} Pupd_st;

/**--Pvec VECTOR---------------------------------------------*/
typedef struct {
Pfloat delta_x; /* delta x value */
Pfloat delta_y; /* delta y value */
} Pvec;

/**--Pvec3 VECTOR 3--------------------------------------------*/
typedef struct {
Pfloat delta_x; /* delta x value */
Pfloat delta_y; /* delta y value */
Pfloat delta_z; /* delta z value */
} Pvec3;

/**--Pvert_text_align VERTICAL TEXT ALIGNMENT--------------------*/
typedef enum {
PVERT_NORM
PVERT_TOP
PVERT_CAP
PVERT_HALF
PVERT_BASE
PVERT_BOTTOM
} Pvert_text_align;

/**--Pview_map VIEW MAPPING--------------------------------------*/
typedef struct {
Plimit win; /* window limits */
Plimit proj_vp; /* projection viewport limits */
} Pview_map;

/**--Pview_map3 VIEW MAPPING 3--------------------------------------*/
typedef struct {
Plimit win; /* window limits */
Plimit3 proj_vp; /* projection viewport limits */
Pproj_type proj_type; /* projection type */
Ppoint3 proj_ref_point; /* projection reference point */
Pfloat view_plane; /* view plane distance */
Pfloat back_plane; /* back plane distance */
Pfloat front_plane; /* front plane distance */
} Pview_map3;

/**--Pview_rep VIEW REPRESENTATION----------------------------------*/
typedef struct {
Pmatrix ori_matrix; /* orientation matrix */
} Pview_rep;
Pmatrix map_matrix; /* mapping matrix */
Plimit clip_limit; /* clipping limits */
Pclip_ind xy_clip; /* X-Y clipping indicator */

} Pview_rep;

/**--Pview_rep3 VIEW REPRESENTATION 3--------------------------------*/
typedef struct {
  Pmatrix3 ori_matrix; /* orientation matrix */
Pmatrix3 map_matrix; /* mapping matrix */
Plimit3 clip_limit; /* clipping limits */
Pclip_ind xy_clip; /* X-Y clipping indicator */
Pclip_ind back_clip; /* back clipping indicator */
Pclip_ind front_clip; /* front clipping indicator */
} Pview_rep3;

/**--Pvisual_st VISUAL STATE----------------------------------------*/
typedef enum {
    PVISUAL_ST_CORRECT
    PVISUAL_ST_DEFER
    PVISUAL_ST_SIMULATED
} Pvisual_st;

/**--Pws_cat WORKSTATION CATEGORY----------------------------------*/
typedef enum {
    PCAT_OUT
    PCAT_IN
    PCAT_OUTIN
    PCAT_MO
    PCAT_MI
} Pws_cat;

/**--Pws_class WORKSTATION CLASS----------------------------------*/
typedef enum {
    PCLASS_VEC
    PCLASS_RASTER
    PCLASS_OTHER
} Pws_class;

/**--Pws_dep_ind WORKSTATION DEPENDENCY INDICATOR----------------*/
typedef enum {
    PWS_INDEP
    PWS_DEP
} Pws_dep_ind;

/**--Pws_st WORKSTATION STATE--------------------------------------*/
typedef enum {
    PWS_ST_WSCL
    PWS_ST_WSOP
} Pws_st;

/**--Pws_st_tables LENGTH OF WORKSTATION STATE TABLES---------------*/
typedef struct {
    Pint line_bundles; /* max.# of polyline table entries*/
    Pint mark_bundles; /* max.# of polymarker tbl entries*/
    Pint text_bundles; /* max.# of text table entries */
    Pint int_bundles; /* max.# of interior table entries*/
    Pint edge_bundles; /* max.# of edge table entries */
    Pint pat_reps; /* max.# of pattern table entries */
    Pint colr_reps; /* max.# of color table entries */
    Pint view_reps; /* max.# of view table entries */
} Pws_st_tables;

Function identifiers
/* Function identifiers */
(Ref #1.)

#define Pfn_open_phigs (0)
#define Pfn_close_phigs (1)
#define Pfn_open_ws (2)
#define Pfn_close_ws (3)
#define Pfn_redraw_all_structs (4)
#define Pfn_upd_ws (5)
#define Pfn_set_disp upd st (6)
#define Pfn_message (7)
#define Pfn_polyline3 (8)
#define Pfn_polyline (9)
#define Pfn_polymarker3 (10)
#define Pfn_polymarker (11)
#define Pfn_text3 (12)
#define Pfn_text (13)
#define Pfn_anno_text rel3 (14)
#define Pfn_anno_text rel (15)
#define Pfn_fill_area3 (16)
#define Pfn_fill_area (17)
#define Pfn_fill_area_set3 (18)
#define Pfn_fill_area_set (19)
#define Pfn_cell_array3 (20)
#define Pfn_cell_array (21)
#define Pfn_gdp3 (22)
#define Pfn_gdp (23)
#define Pfn_set_line_ind (24)
#define Pfn_set_marker_ind (25)
#define Pfn_set_text_ind (26)
#define Pfn_set_int_ind (27)
#define Pfn_set_edge_ind (28)
#define Pfn_set_linetype (29)
#define Pfn_set_linewidth (30)
#define Pfn_set_line_colr_ind (31)
#define Pfn_set_marker_type (32)
#define Pfn_set_marker_size (33)
#define Pfn_set_marker_colr_ind (34)
#define Pfn_set_text_font (35)
#define Pfn_set_text_prec (36)
#define Pfn_set_char_expan (37)
#define Pfn_set_char_space (38)
#define Pfn_set_text_colr_ind (39)
#define Pfn_set_char_ht (40)
#define Pfn_set_char_up_vec (41)
#define Pfn_set_text_path (42)
#define Pfn_set_text_align (43)
#define Pfn_set_anno_char_ht (44)
#define Pfn_set_anno_char_up_vec (45)
#define Pfn_set_anno_path (46)
#define Pfn_set_anno_align (47)
#define Pfn_set_anno_style (48)
Chapter 18. ISO PHIGS C Type and Macro Definitions
#define Pfn_ar_structs (116)
#define Pfn_ar_struct_nets (117)
#define Pfn_ar_all_structs (118)
#define Pfn_set_conf_res (119)
#define Pfn_ret_struct_1ds (120)
#define Pfn_ret_structs (121)
#define Pfn_ret_struct_nets (122)
#define Pfn_ret_all_structs (123)
#define Pfn_ret_paths_ances (124)
#define Pfn_ret_paths_descs (125)
#define Pfn_del_structs_ar (126)
#define Pfn_del_struct_nets_ar (127)
#define Pfn_del_all_structs_ar (128)
#define Pfn_set_pick_id (129)
#define Pfn_set_pick_filter (130)
#define Pfn_init_loc3 (131)
#define Pfn_init_loc (132)
#define Pfn_init_stroke3 (133)
#define Pfn_init_stroke (134)
#define Pfn_init_val3 (135)
#define Pfn_init_val (136)
#define Pfn_init_choice3 (137)
#define Pfn_init_choice (138)
#define Pfn_init_pick3 (139)
#define Pfn_init_pick (140)
#define Pfn_init_string3 (141)
#define Pfn_init_string (142)
#define Pfn_set_loc_mode (143)
#define Pfn_set_stroke_mode (144)
#define Pfn_set_val_mode (145)
#define Pfn_set_choice_mode (146)
#define Pfn_set_pick_mode (147)
#define Pfn_set_string_mode (148)
#define Pfn_req_loc3 (149)
#define Pfn_req_loc (150)
#define Pfn_req_stroke3 (151)
#define Pfn_req_stroke (152)
#define Pfn_req_val (153)
#define Pfn_req_choice (154)
#define Pfn_req_pick (155)
#define Pfn_req_string (156)
#define Pfn_sample_loc3 (157)
#define Pfn_sample_loc (158)
#define Pfn_sample_stroke3 (159)
#define Pfn_sample_stroke (160)
#define Pfn_sample_val (161)
#define Pfn_sample_choice (162)
#define Pfn_sample_pick (163)
#define Pfn_sample_string (164)
#define Pfn_await_event (165)
#define Pfn_flush_events (166)
#define Pfn_get_loc3 (167)
#define Pfn_get_loc (168)
#define Pfn_get_stroke3 (169)
#define Pfn_get_stroke (170)
#define Pfn_get_val (171)
#define Pfn_get_choice (172)
#define Pfn_get_pick (173)
#define Pfn_get_string (174)
#define Pfn_write_item (175)
#define Pfn_get_item_type (176)
#define Pfn_read_item (177)
#define Pfn_interpret_item (178)
#define Pfn_set_err_hand_mode (179)
#define Pfn_set_err_hand (180)
#define Pfn_set_pick_id (181)
Error codes

/* Error codes */

/* <0 Implementation Dependent Errors */

#define PE_NO_ERROR (0) /* No Error */

#define PE_NOT_PHCL (1) /* Ignoring function, function requires state */
/* (PHCL, WSCL, STCL, ARCL) */

#define PE_NOT_PHOP (2) /* Ignoring function, function requires state */
/* (PHOP, *, *, *) */

#define PE_NOT_WSOP (3) /* Ignoring function, function requires state */
/* (PHOP, WSOP, *, *) */

#define PE_NOT_CL (4) /* Ignoring function, function requires state */
/* (PHOP, WSCL, STCL, ARCL) */

#define PE_NOT_STOP (5) /* Ignoring function, function requires state */
/* (PHOP, *, STOP, *) */

#define PE_NOT_STCL (6) /* Ignoring function, function requires state */
/* (PHOP, *, STCL, *) */

#define PE_NOT_AROP (7) /* Ignoring function, function requires state */
/* (PHOP, *, *, AROP) */

/* Workstation Errors */

#define PE_BAD_CONN_ID (50) /* Ignoring function, connection not recognized by the implementation */
/* identifier not recognized by the implementation */

#define PE_WS_TYPE (51) /* Ignoring function, workstation type not recognized by the implementation */
/* this workstation type is not yet available for workstation of this type and use */
/* the specific workstation type */

#define PE_BAD_WS_TYPE (52) /* Ignoring function, workstation type not recognized by the implementation */
/* workstation of this type and use */

#define PE_DUP_WS_ID (53) /* Ignoring function, workstation identifier already in use */

#define PE_WS_NOT_OPEN (54) /* Ignoring function, the specified workstation is not open */

#define PE_NO_OPEN_WS (55) /* Ignoring function, workstation can not be opened for an implementation dependent reason */

#define PE_WS_NOT_MO (56) /* Ignoring function, specified workstation not of category MO*/
#define PE_WS_MI (57) /* Ignoring function, specified */
/* workstation is of category MI */

#define PE_WS_NOT_MI (58) /* Ignoring function, specified */
/* workstation is not of category MI */

#define PE_WS_NO_OUTPUT (59) /* Ignoring function, specified */
/* workstation does not have output */
/* capability (i.e., the workstation */
/* category is neither OUTPUT, OUTIN*/
/* nor MO) */

#define PE_WS_NOT_OUTIN (60) /* Ignoring function, specified */
/* workstation is not of category */
/* OUTIN */

#define PE_WS_NO_INPUT (61) /* Ignoring function, specified */
/* workstation is neither category */
/* INPUT nor category OUTIN */

#define PE_WS_NOT_OUT (62) /* Ignoring function, specified */
/* workstation is neither category */
/* OUTPUT nor category OUTIN */

#define PE_MAX_WS (63) /* Ignoring function, opening this */
/* workstation would exceed the */
/* maximum number of simultaneously */
/* open workstations */

#define PE_NO_GDP (64) /* Ignoring function, the specified */
/* workstation type is not able to */
/* generate the specified */
/* generalized drawing primitive */

*/ Output Attribute Errors */
#define PE_BUN_IND_LT_1 (100) /* Ignoring function, the bundle */
/* index value is less than one */

#define PE_REP_UNDEF (101) /* Ignoring function, the specified */
/* representation has not been */
/* defined. */

#define PE_REP_NOT_PREDEF (102) /* Ignoring function, the specified */
/* representation has not been */
/* predefined on this workstation */

#define PE_MAX_BUN (103) /* Ignoring function, setting this */
/* bundle table entry would exceed */
/* the maximum number of entries */
/* allowed in the workstation bundle*/
/* table */

#define PE_BAD_LINETYPE (104) /* Ignoring function, the specified */
/* linetype is not available on the */
/* specified workstation */

#define PE_BAD_MARKER_TYPE (105) /* Ignoring function, the specified */
/* marker type is not available on */
/* the specified workstation */

#define PE_BAD_FONT (106) /* Ignoring function, the specified */
/* font is not available for the */
/* requested text precision on the */
/* specified workstation */
#define PE_BAD_EDGETYPE (107) /* Ignoring function, the specified */
/* edgetype is not available on the */
/* specified workstation */

#define PE_BAD_INT_STYLE (108) /* Ignoring function, the specified */
/* interior style is not available */
/* on the workstation */

#define PE_NO_PAT (109) /* Ignoring function, interior style*/
/* PATTERN is not supported on the */
/* workstation */

#define PE_BAD_COLR_MODEL (110) /* Ignoring function, the specified */
/* color model is not available on */
/* the workstation */

#define PE_BAD_HLHSR_MODE (111) /* Ignoring function, the specified */
/* HLHSR mode is not available on */
/* the specified workstation */

#define PE_PAT_IND_LT_1 (112) /* Ignoring function, the pattern */
/* index value is less than one */

#define PE_COLR_IND_LT_0 (113) /* Ignoring function, the color */
/* index value is less than zero */

#define PE_VIEW_IND_LT_0 (114) /* Ignoring function, the view index*/
/* value is less than zero */

#define PE_VIEW_IND_LT_1 (115) /* Ignoring function, the view index*/
/* value is less than one */

#define PE_BAD_PAT_DIM (116) /* Ignoring function, one of the */
/* dimensions of pattern color */
/* array is less than one */

#define PE_BAD_COLR_DIM (117) /* Ignoring function, one of the */
/* dimensions of the color index */
/* array is less than zero */

#define PE_BAD_COLR (118) /* Ignoring function, one of the */
/* components of the color specifi- */
/* cation is out of range. The valid*/
/* range is dependent upon the */
/* current color model */

/* Transformations and Viewing Errors */
#define PE_MAX_VIEW (150) /* Ignoring function, setting this */
/* view table entry would exceed */
/* the maximum number of entries */
/* allowed in the workstations */
/* view table */

#define PE_INVALID_WINDOW (151) /* Ignoring function, */
/* invalid window; */
/* XMIN >= XMAX, YMIN >= YMAX or */
/* ZMIN > ZMAX */

#define PE_INVALID_VIEWPORT (152) /* Ignoring function, invalid view-*/
/* port; XMIN >= XMAX, YMIN >= YMAX*/
/* or ZMIN > ZMAX */

Chapter 18. ISO PHIGS C Type and Macro Definitions 489
#define PE_INVALID_CLIP
/* Ignoring function, invalid view */
/* clipping limits; XMIN >= XMAX, */
/* YMIN >= YMAX or ZMIN > ZMAX */

#define PE_BAD_CLIP
/* Ignoring function, the view */
/* clipping limits are not within */
/* NPC range */

#define PE_BAD_PROJ_VIEWPORT
/* Ignoring function, the projection viewport limits are not */
/* within NPC range */

#define PE_BAD_WS_WINDOW
/* Ignoring function, the workstation window limits are not */
/* within NPC range */

#define PE_BAD_WS_VIEWPORT
/* Ignoring function, the workstation viewport is not */
/* within display space */

#define PE_BAD_PLANES
/* Ignoring function, front plane */
/* and back plane distances are */
/* equal when z-extent of the */
/* projection viewport is zero */

#define PE_BAD_VPN
/* Ignoring function, the view */
/* plane normal vector has length */
/* zero */

#define PE_BAD_VUP
/* Ignoring function, the view up */
/* vector has length zero */

#define PE_BAD_VUP_VPN
/* Ignoring function, the view up */
/* and view plane normal vectors */
/* are parallel thus the viewing */
/* coordinate system cannot be */
/* established */

#define PE_BAD_PRP
/* Ignoring function, the projection */
/* reference point is between the */
/* front and back planes */

#define PE_PRP_VIEW_PLANE
/* Ignoring function, the projection */
/* reference point cannot be posi- */
/* tioned on the view plane */

#define PE_FRONT_BACK
/* Ignoring function, the back plane */
/* is in front of the front plane */

/* Structure Errors */
#define PE_IGNORE_STRUCTS
/* Warning, ignoring structures */
/* that do not exist */

#define PE_BAD_STRUCT
/* Ignoring function, the specified */
/* structure does not exist */

#define PE_BAD_ELEMENT
/* Ignoring function, the specified */
/* element does not exist */

#define PE_BAD_PATH
/* Ignoring function, specified */
/* starting path not found in CSS */

#define PE_BAD_CEILING_IND
/* Ignoring function, specified */
/* search ceiling index out of */
/* range */
<table>
<thead>
<tr>
<th>Macro Description</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE_NO_LABEL</td>
<td>#define PE_NO_LABEL (205) /* Ignoring function, the label */</td>
</tr>
<tr>
<td></td>
<td>/* does not exist in the open */</td>
</tr>
<tr>
<td></td>
<td>/* structure between the element */</td>
</tr>
<tr>
<td></td>
<td>/* pointer and the end of the */</td>
</tr>
<tr>
<td></td>
<td>/* structure */</td>
</tr>
<tr>
<td>PE_NO_LABELS</td>
<td>#define PE_NO_LABELS (206) /* Ignoring function, one or both */</td>
</tr>
<tr>
<td></td>
<td>/* of the labels does not exist in */</td>
</tr>
<tr>
<td></td>
<td>/* the open structure between the */</td>
</tr>
<tr>
<td></td>
<td>/* element pointer and the end of */</td>
</tr>
<tr>
<td></td>
<td>/* the structure */</td>
</tr>
<tr>
<td>PE_BAD_PATHDEPTH</td>
<td>#define PE_BAD_PATHDEPTH (207) /* Ignoring function, the specified */</td>
</tr>
<tr>
<td></td>
<td>/* path depth is less than zero (0) */</td>
</tr>
<tr>
<td>PE_BAD_DISP_PRI</td>
<td>#define PE_BAD_DISP_PRI (208) /* Ignoring function, the display */</td>
</tr>
<tr>
<td></td>
<td>/* priority is out of range */</td>
</tr>
<tr>
<td>Input Errors</td>
<td>/* */</td>
</tr>
<tr>
<td>PE_NO_DEVICE</td>
<td>#define PE_NO_DEVICE (250) /* Ignoring function, the specified device */</td>
</tr>
<tr>
<td></td>
<td>/* is not available on the */</td>
</tr>
<tr>
<td></td>
<td>/* specified workstation */</td>
</tr>
<tr>
<td>PE_NOT_REQUEST</td>
<td>#define PE_NOT_REQUEST (251) /* Ignoring function, the function */</td>
</tr>
<tr>
<td></td>
<td>/* requires the input device to be */</td>
</tr>
<tr>
<td></td>
<td>/* in REQUEST mode */</td>
</tr>
<tr>
<td>PE_NOT_SAMPLE</td>
<td>#define PE_NOT_SAMPLE (252) /* Ignoring function, the function */</td>
</tr>
<tr>
<td></td>
<td>/* requires the input device to be */</td>
</tr>
<tr>
<td></td>
<td>/* in SAMPLE mode */</td>
</tr>
<tr>
<td>PE_BAD_PET</td>
<td>#define PE_BAD_PET (253) /* Warning, the specified prompt */</td>
</tr>
<tr>
<td></td>
<td>/* echo type is not available on */</td>
</tr>
<tr>
<td></td>
<td>/* the specified workstation */</td>
</tr>
<tr>
<td></td>
<td>/* Prompt/echo type one will be */</td>
</tr>
<tr>
<td></td>
<td>/* used in its place */</td>
</tr>
<tr>
<td>PE_INVALID_ECHO</td>
<td>#define PE_INVALID_ECHO (254) /* Ignoring function, invalid echo */</td>
</tr>
<tr>
<td></td>
<td>/* area/volume; XMIN &gt;= XMAX, */</td>
</tr>
<tr>
<td></td>
<td>/* YMIN &gt;= YMAX or ZMIN &gt;= ZMAX */</td>
</tr>
<tr>
<td>PE_BAD_ECHO</td>
<td>#define PE_BAD_ECHO (255) /* Ignoring function, one of the */</td>
</tr>
<tr>
<td></td>
<td>/* echo area/volume boundary points */</td>
</tr>
<tr>
<td></td>
<td>/* is outside the range of the */</td>
</tr>
<tr>
<td></td>
<td>/* device */</td>
</tr>
<tr>
<td>PE_QUEUE_OFLOW</td>
<td>#define PE_QUEUE_OFLOW (256) /* Warning, the input queue has */</td>
</tr>
<tr>
<td></td>
<td>/* overflowed */</td>
</tr>
<tr>
<td>PE_NO_QUEUE_OFLOW</td>
<td>#define PE_NO_QUEUE_OFLOW (257) /* Ignoring function, input queue */</td>
</tr>
<tr>
<td></td>
<td>/* has not overflowed */</td>
</tr>
<tr>
<td>PE_OFLOW_NO_GO</td>
<td>#define PE_OFLOW_NO_GO (258) /* Ignoring function, input queue */</td>
</tr>
<tr>
<td></td>
<td>/* has overflowed, but associated */</td>
</tr>
<tr>
<td></td>
<td>/* workstation has been closed */</td>
</tr>
<tr>
<td>PE_BAD_CLASS</td>
<td>#define PE_BAD_CLASS (259) /* Ignoring function, the input */</td>
</tr>
<tr>
<td></td>
<td>/* device class of the current */</td>
</tr>
<tr>
<td></td>
<td>/* input report does not match the */</td>
</tr>
<tr>
<td></td>
<td>/* class being requested */</td>
</tr>
<tr>
<td>PE_BAD_DATA_REC</td>
<td>#define PE_BAD_DATA_REC (260) /* Ignoring function, one of the */</td>
</tr>
<tr>
<td></td>
<td>/* fields within the input device */</td>
</tr>
<tr>
<td></td>
<td>/* data record is in error */</td>
</tr>
</tbody>
</table>
#define PE_INVALID_VALUE (261) /* Ignoring function, initial value*/
    /* is invalid */
#define PE_STROKE_BUF_SIZE (262) /* Ignoring function, number of */
    /* points in the initial stroke */
    /* greater than the buffer size */
#define PE_STRING_BUF_SIZE (263) /* Ignoring function, length of */
    /* the initial string is greater */
    /* than the buffer size */

/* Metafile Errors */
#define PE_ILLEGAL_ITEM_TYPE (300) /* Ignoring function, item type is*/
    /* not allowed for user items */
#define PE_INVALID_ITEM_LENGTH (301) /* Ignoring function, item length*/
    /* is invalid */
#define PE_METAFILE_EMPTY (302) /* Ignoring function, no item is */
    /* left in metafile input */
#define PE_INVALID_ITEM (303) /* Ignoring function, metafile */
    /* item is invalid */
#define PE_BAD_ITEM_TYPE (304) /* Ignoring function, item type is */
    /* unknown */
#define PE_BAD_ITEM_REC (305) /* Ignoring function, content of */
    /* item data record is invalid for */
    /* the specified item type */
#define PE_MAX_ITEM_LENGTH (306) /* Ignoring function, maximum item */
    /* data record length is invalid */
#define PE_USER_ITEM (307) /* Ignoring function, user item */
    /* can not be interpreted */

/* Escape Errors */
#define PE_ESCAPE_NOT_AVAIL (350) /* Warning, the specified escape */
    /* is not available on one or more */
    /* workstations in this implementa- */
    /* tion. The escape will be */
    /* processed by those workstations */
    /* on which it is available */
#define PE_BAD_ESCAPE_DATA (351) /* Ignoring function, one of the */
    /* fields within the escape data */
    /* record is in error */

/* Archival and Retrieval Errors */
#define PE_AR_CANT_OPEN (400) /* Ignoring function, the archive */
    /* file cannot be opened */
#define PE_MAX_AR (401) /* Ignoring function, opening */
    /* this archive file would exceed */
    /* the maximum number of */
    /* simultaneously open archive */
    /* files */
#define PE_DUP_AR_ID (402) /* Ignoring function, archive */
    /* file identifier already in use */
#define PE_BAD_AR (403) /* Ignoring function, the archive */
    /* file is not a PHIGS archive */
#define PE_AR_NOT_OPEN (404) /* Ignoring function, the specified archive file is not open */

#define PE_NAME_CONFLICT (405) /* Ignoring function, the specified archive file name conflict occurred */

#define PE_AR_FULL (406) /* Warning, the archive file is full. Any structures that were archived were archived in total */

#define PE_AR_NO_STRUCT (407) /* Warning, some of the specified structures do not exist on the archive file */

#define PE_AR_NO_STRUCT_EMPTY (408) /* Warning, some of the specified structures do not exist on the archive file. graPHIGS will create empty structures in their places */

// Miscellaneous Errors */
#define PE_BAD_ERROR_FILE (450) /* Ignoring function, the specified error file is invalid */

// System Errors */
#define PE_OFLOW_PHIGS (900) /* Storage overflow has occurred in PHIGS */

#define PE_OFLOW_CSS (901) /* Storage overflow has occurred in CSS */

#define PE_IO_ERROR_READ (902) /* Input/Output error has occurred while reading */

#define PE_IO_ERROR_WRITE (903) /* Input/Output error has occurred while writing */

#define PE_IO_ERROR_TO_WS (904) /* Input/Output error has occurred while sending data to a workstation */

#define PE_IO_ERROR_FROM_WS (905) /* Input/Output error has occurred while receiving data from a workstation */

#define PE_IO_ERROR_LIB (906) /* Input/Output error has occurred during program library management */

#define PE_IO_ERROR_WDT (907) /* Input/Output error has occurred while reading workstation description table */

#define PE_ARITHMETIC_ERROR (908) /* Arithmetic error has occurred */

// Binding Specific Errors */
#define PE_START_IND_INVAL (2200) /* Ignoring function, start index is out of range */
# define PE_LIST_LENGTH_LT_ZERO (2201) /* Ignoring function, the * /
     /* length of the application's * /
     /* list is negative * /

#define PE_ENUM_TYPE_INVAL (2202) /* Ignoring function, enumeration */
     /* type is out of range */

#define PE_ALLOC_STORE (2203) /* Ignoring function, error while */
     /* allocating a Store */

#define PE_ALLOC_STORE_MEM (2204) /* Ignoring function, error while */
     /* allocating memory for a Store */

/* Miscellaneous */

/* Linetypes */
#define PLINE_SOLID (1)
#define PLINE_DASH (2)
#define PLINE_DOT (3)
#define PLINE_DASH_DOT (4)

/* Marker types */
#define PMARKER_DOT (1)
#define PMARKER_PLUS (2)
#define PMARKER_ASTERISK (3)
#define PMARKER_CIRCLE (4)
#define PMARKER_CROSS (5)

/* Annotation styles */
#define PANNO_STYLE_UNCONNECTED (1)
#define PANNO_STYLE_LEAD_LINE (2)

/* Color models */
#define PMODEL_RGB (1)
#define PMODEL_CIELUV (2)
#define PMODEL_HSV (3)
#define PMODEL_HLS (4)

/* Prompt and Echo Types */
#define PLOC_DEF (1)
#define PLOC_CROSS_HAIR (2)
#define PLOC_TRACK_CROSS (3)
#define PLOC_RUB_BAND (4)
#define PLOC_RECT (5)
#define PLOC_DIGIT (6)

#define PSTROKE_DEF (1)
#define PSTROKE_DIGIT (2)
#define PSTROKE_MARKER (3)
#define PSTROKE_LINE (4)

#define PVAL_DEF (1)
#define PVAL_GRAPH (2)
#define PVAL_DIGIT (3)

#define PCHOICE_DEF (1)
/* Default parameters of Open PHIGS */
#define PDEF_MEM_SIZE ((size_t) (-1))
#define PDEF_ERR_FILE ((char*) (0))

/* Element enumeration */
#define PFIRST_PHIGS_ELEM PELEM_POLYLINE3
#define PLAST_PHIGS_ELEM PELEM_PICK_ID
Chapter 19. ISO PHIGS FORTRAN Enumeration Types

C **-----------------------------------------------**
C * ISO PHIGS FORTRAN Enumeration Types *
C * ** All the enumeration types of PHIGS are mapped **
C * to Fortran INTEGERS. The correspondence between *
C * PHIGS scalars and FORTRAN INTEGERS is as follows *
C * in a list of symbolic FORTRAN constants that may *
C * be included by an application program. Also *
C * included is a mapping of PHIGS enumeration types *
C * to FORTRAN variable names. Line type, marker type,*
C * and color model are included for convenience even *
C * though PHIGS defines them as integer rather than *
C * as enumerations. The numbering of all PHIGS *
C * functions is also given for use in the error *
C * handling procedures. *
C **-----------------------------------------------**
C
C **-----------------------------------------------**
C ---- Annotation Style ----
C INTEGER PUNCON, PLDLN
PARAMETER(PUNCON=1, PLDLN=2)

C ---- Archive State ----
C INTEGER PARCL, PAROP
PARAMETER(PARCL=0, PAROP=1)

C **** Aspect Identifier ****
C INTEGER PLN, PLWSC, PPLCI, PMK, PMKSC,
1     PPMCI, PTXFN, PTXPR, PCHXP, PCHSP,
2     PTXCI, PIS, PISI, PICI, PEDFG,
3     PEDT, PEWSC, PEDCI
PARAMETER(PLN=0, PLWSC=1, PPLCI=2, PMK=3, PMKSC=4,
1     PPMCI=5, PTXFN=6, PTXPR=7, PCHXP=8, PCHSP=9,
2     PTXCI=10, PIS=11, PISI=12, PICI=13, PEDFG=14,
3     PEDT=15, PEWSC=16, PEDCI=17)

C ---- Aspect Source ----
C INTEGER PBUNDL, PINDIV
PARAMETER(PBUNDL=0, PINDIV=1)

C ---- Clipping Indicator ----
C INTEGER PNCLIP, PCLIP
PARAMETER(PNCLIP=0, PCLIP=1)

C ---- Color Available ----
C INTEGER PMONOC, PCOLOR
PARAMETER(PMONOC=0, PCOLOR=1)

C ---- Color Model ----
C INTEGER PRGB, PCIE, PHSV, PHLS
PARAMETER(PRGB=1, PCIE=2, PHSV=3, PHLS=4)

C ---- Composition Type ----
C INTEGER PCPRE, PCPOST, PCREPL
PARAMETER(PCPRE=0, PCPOST=1, PCREPL=2)

C ---- Conflict Resolution ----
C INTEGER PCRMNT, PCRABA, PCRUPD
PARAMETER(PCRMNT=0, PCRABA=1, PCRUPD=2)

C ---- Control Flag ----
C INTEGER PCONDI, PALWAY
PARAMETER(PCONDI=0, PALWAY=1)

C ---- Deferral Mode ----
 integer pasap, pnig, pnil, pasti, pwaitd
 parameter(pasap=0, pnig=1, pnil=2, pasti=3, pwaitd=4)

c ---- device coordinate units ----
 integer pmetre, pthu
 parameter(pmetre=0, pthu=1)

c ---- display surface empty ----
 integer pnempt, pempty
 parameter(pnempt=0, pempty=1)

c ---- dynamic modification ----
 integer pirq, pimm, pcb
 parameter(pirq=0, pimm=1, pcb=2)

c ---- echo switch ----
 integer pneo, pcho
 parameter(pneo=0, pcho=1)

c ---- edit mode ----
 integer pinsrt, preplc
 parameter(pinsrt=0, preplc=1)

c ---- element type ----
 integer pea1, peni, pepl3, pepl,
 1 pepm3, pepm, petx3, petx,
 2 petr3, petr, pefa3, pefa,
 3 pefas3, pefas, peca3, peca,
 4 pegdp3, pegdp, pepli, pempm,
 5 petxi, peii, peedi, peln,
 6 pelwsc, pelpci, pemk, pemks,
 7 pemmic, petxfn, petxpr, pecho,
 8 pechsp, petxc1, pechh, pechup,
 9 petxp, petxal, peatch, petauc,
a peaf, peatal, peanst, peis,
b peisi, peici, peedg, peedt,
c peewsc, peedic, pepa, peprpv,
d peparf, peads, peres, peiasf,
e pehrid, pelmt3, pelm, pegm3,
f pemgt, pemcv3, pemcv, pemcli,
g permcv, pewj, peexst, pelb,
h peap, pegse, pepkid
 parameter(pea1=0, peni=1, pepl3=2, pepl=3,
 1 pem3=4, pepm=5, petx3=6, petx=7,
 2 petr3=8, petr=9, pefa3=10, pefa=11,
 3 pefas3=12, pefas=13, peca3=14, peca=15,
 4 pegdp3=16, pegdp=17, pepli=18, pempm=19,
 5 petxi=20, peii=21, peedi=22, peln=23,
 6 pelwsc=24, pelpci=25, pemk=26, pemks=27,
 7 pemmic=28, petxfn=29, petxpr=30, pecho=31,
 8 pechsp=32, petxc1=33, pechh=34, pechup=35,
 9 petxp=36, petxal=37, peatch=38, petauc=39,
a peaf=40, peatal=41, peanst=42, peis=43,
b peisi=44, peici=45, peedg=46, peedt=47,
c peewsc=48, peedic=49, pepa=50, peprpv=51,
d peparf=52, peads=53, peres=54, peiasf=55,
e pehrid=56, pelmt3=57, pelm=58, pegm3=59,
f pemgt=60, pemcv3=61, pemcv=62, pemcli=63,
g permcv=64, pewj=65, peexst=66, pelb=67,
h peap=68, pegse=69, pepkid=70)

c ---- gdp attributes ----
 integer pplatt, ppmatt, ptatt, pinatt, pedatt
 parameter(pplatt=0, ppmatt=1, ptatt=2, pinatt=3, pedatt=4)

c ---- input class ----
INTEGER PNCLAS, PLOCAT, PSTROK, PVALUA, PCHOIC,
1 PPICK, PSTRIN
PARAMETER(PNCLAS=0, PLOCAT=1, PSTROK=2, PVALUA=3, PCHOIC=4,
1 PPICK=5, PSTRIN=6)

C ---- Input Device Status ----
INTEGER PNONE, POK, PNpick, PNCHOI
PARAMETER(PNONE=0, POK=1, PNpick=2, PNCHOI=2)

C ---- Interior Style ----
INTEGER PHOLLO, PSOLID, PPATTR, PHATCH, PISEMP
PARAMETER(PHOLLO=0, PSOLID=1, PPATTR=2, PHATCH=3, PISEMP=4)

C ---- Linetype ----
INTEGER PLsoli, PLdash, PLdot, PLdASD
PARAMETER(PLsoli=1, PLdash=2, PLdot=3, PLdASD=4)

C ---- Marker Type ----
INTEGER Ppoint, Pplus, PaST, POMark, PXMARK
PARAMETER(Ppoint=1, Pplus=2, PaST=3, POMark=4, PXMARK=5)

C ---- Modeling Clip Operator ----
INTEGER PMCREP, PMCINT
PARAMETER(PMCREP=1, PMCINT=2)

C ---- Modification Mode ----
INTEGER PNIVE, PUWOR, PUQUM
PARAMETER(PNIVE=0, PUWOR=1, PUQUM=2)

C ---- More Simultaneous Events ----
INTEGER PNMORE, PMORE
PARAMETER(PNMORE=0, PMORE=1)

C ---- Off/On Switch for Edge Flag and Error Handling ----
INTEGER POFF, PON
PARAMETER(POFF=0, PON=1)

C ---- Open Structure Status ----
INTEGER PNONST, POPNST
PARAMETER(PNONST=0, POPNST=1)

C ---- Operating Mode ----
INTEGER PREQU, PSAMPL, PEVENT
PARAMETER(PREQU=0, PSAMPL=1, PEVENT=2)

C ---- Path Order ----
INTEGER PPOTOP, PPOBOT
PARAMETER(PPOTOP=0, PPOBOT=1)

C ---- Polyline/Fill Area Control Flag ----
INTEGER PPLINE, PFILLA, PFIAs
PARAMETER(PPLINE=0, PFILLA=1, PFIAs=2)

C ---- Presence of Invalid Values ----
INTEGER PABSNT, PPRSNT
PARAMETER(PABSNT=0, PPRSNT=1)

C ---- Reference Handling Flag ----
INTEGER PDELE, PKEEP
PARAMETER(PDELE=0, PKEEP=1)

C ---- Regeneration Flag ----
INTEGER PPOSTP, PPERFO
PARAMETER(PPOSTP=0, PPERFO=1)

C ---- Relative Input Priority ----
INTEGER PHIGHR, PLOWER
PARAMETER(PHIGHR=0, PLOWER=1)

C ---- Search Direction ----
INTEGER PBWD, PFWD
PARAMETER(PBWD=0, PFWD=1)

C ---- Search Success Indicator ----
INTEGER PFAIL, PSUCC
PARAMETER(PFAIL=0, PSUCC=1)

C ---- State of Visual Representation ----
INTEGER PVROK, PVRDFR, PVRSIM
PARAMETER(PVROK=0, PVRDFR=1, PVRSIM=2)

C ---- Structure Network Source ----
INTEGER PCSS, PARCHV
PARAMETER(PCSS=0, PARCHV=1)

C ---- Structure State Value ----
INTEGER PSTCL, PSTOP
PARAMETER(PSTCL=0, PSTOP=1)

C ---- Structure Status Indicator ----
INTEGER PSNOEX, PSEMPT, PSEMPT
PARAMETER(PSNOEX=0, PSEMPT=1, PSEMPT=2)

C ---- System State Value ----
INTEGER PPHCL, PPHOP
PARAMETER(PPHCL=0, PPHOP=1)

C ---- Text Alignment Horizontal ----
INTEGER PAHNOR, PALEFT, PACENT, PARITE
PARAMETER(PAHNOR=0, PALEFT=1, PACENT=2, PARITE=3)

C ---- Text Alignment Vertical ----
INTEGER PAVNOR, PATOP, PACAP, PASHALF, PABASE, PABOTT
PARAMETER(PAVNOR=0, PATOP=1, PACAP=2, PASHALF=3, PABASE=4, PABOTT=5)

C ---- Text Path ----
INTEGER PRIGHT, PLEFT, PUP, PDOWN
PARAMETER(PRIGHT=0, PLEFT=1, PUP=2, PDOWN=3)

C ---- Text Precision ----
INTEGER PSTRP, PCHARP, PSTRKP
PARAMETER(PSTRP=0, PCHARP=1, PSTRKP=2)

C ---- Type of Returned Values ----
INTEGER PSET, PREALI
PARAMETER(PSET=0, PREALI=1)

C ---- Update State ----
INTEGER PNPEND, PPEND
PARAMETER(PNPEND=0, PPEND=1)

C ---- Vector/Raster/Other Type ----
INTEGER PVECTR, PRASTR, POTHNK
PARAMETER(PVECTR=0, PRASTR=1, POTHNK=2)

C ---- View Type ----
INTEGER PPARL, PPERS
PARAMETER(PPARL=0, PPERS=1)

C ---- Workstation Category ----
INTEGER POUTPT, PINPUT, POUTIN, PMO, PMI
PARAMETER(POUTPT=0, PINPUT=1, POUTIN=2, PMO=3, PMI=4)
--- Workstation Dependency Indicator ----
INTEGER PWKI, PWKD
PARAMETER(PWKI=0, PWKD=1)

--- Workstation State Value ----
INTEGER PWSCL, PWSOP
PARAMETER(PWSCL=0, PWSOP=1)

--- Current/Requested ----
INTEGER PCURVL, PRQSVL
PARAMETER(PCURVL=0, PRQSVL=1)

--- PHIGS Function Identifiers ---- (Ref 
#2.)

INTEGER EOPPH, ECLPH, EOPWK, ECLWK, ERST
PARAMETER(EOPPH=0, ECLPH=1, EOPWK=2, ECLWK=3, ERST=4)
INTEGER EUNK, EUSDUS, EMSG, EPL3, EPL
PARAMETER(EUNK=5, EUSDUS=6, EMSG=7, EPL3=8, EPL=9)
INTEGER EPM3, EPM, ETX3, ETX, EATR3
PARAMETER(EPM3=10, EPM=11, ETX3=12, ETX=13, EATR3=14)
INTEGER EATR, EFA3, EFA, EFA3, EFA3=16, EFA3=17, EFA3=18, EFA3=19
PARAMETER(ECA3, ECA, EGD03, EGD0, ESPLI
PARAMETER(ECA3=20, ECA=21, EGD03=22, EGD0=23, ESPLI=24)
PARAMETER(EOPST=60, ECLST=61, EEXST=62, ELB=63, EAP=64)
PARAMETER(EGSE=70, ESEDM=71, ECELST=72, ESEP=73, EOSEP=74)
PARAMETER(ESEPLB=80, EDEL=81, EDELRA=82, EDELLB=83, EEMST=84)
PARAMETER(ECLARF=90, EARST=91, EARSN=92, EARSN=93, ESCNRS=94)
PARAMETER(ERSID=100, EREST=101, ERESN=102, EREST=103, EREPM=104)
PARAMETER(ECLARF=110, ECLARF=111, EPOST=112, EPOST=113, EOPARF=114)
PARAMETER(EREPDE=120, EEDSTAR=121, EEDSNAR=122, EEDSAR=123, ESPIKID=124)
PARAMETER(EERPE=130, EADSTAR=131, EADSNAR=132, EADSNAR=133, EADSAR=134)
PARAMETER(ESPKFT=130, EINLC3=131, EINLC=132, EINSK3=133, EINSK=134)
PARAMETER(EINVL3=135, EINVL=136, EINCH3=137, EINCH=138, EINPK3=139)
PARAMETER(EINPK=140, EINST3=141, EINST=142, ESLCM=143, ESSKM=144)
PARAMETER(EINVL3=145, EINVL=146, EINCH3=147, EINCH=148, EINPK3=149)
PARAMETER(EQRLC=150, EQRLC3=151, EQRLC=152, EQRLC3=153, EQRLC=154)
PARAMETER(EQRLC=155, EQRLC3=156, EQRLC3=157, EQRLC3=158, EQRLC3=159)
PARAMETER(EQRLC=160, EQRLC3=161, EQRLC3=162, EQRLC3=163, EQRLC3=164)
PARAMETER(EQRLC=165, EQRLC3=166, EQRLC3=167, EQRLC3=168, EQRLC3=169)
PARAMETER(EQRLC=170, EQRLC3=171, EQRLC3=172, EQRLC3=173, EQRLC3=174)
PARAMETER(EQRLC=175, EQRLC3=176, EQRLC3=177, EQRLC3=178, EQRLC3=179)
PARAMETER(EQRLC=180, EQRLC3=181, EQRLC3=182, EQRLC3=183, EQRLC3=184)
Chapter 20. ISO PHIGS Subroutines to GPxxxx Subroutines

The following table associates ISO PHIGS subroutine calls with equivalent or approximately equivalent GPxxxx calls. Note that the calls may be approximately equivalent; there may be some variation in the call interface or in the functionality. Refer to The graPHIGS Programming Interface: Subroutine Reference and the individual ISO PHIGS subroutine calls for details.

The table is arranged alphabetically by ISO PHIGS subroutine names. A missing ISO PHIGS subroutine implies that either it is not supported by the graPHIGS API, or there is no approximately equivalent GPxxxx subroutine call.

<table>
<thead>
<tr>
<th>ISO PHIGS Subroutine Call</th>
<th>Associated GPxxxx Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Names to Set</td>
<td>GPADCN</td>
</tr>
<tr>
<td>Annotation Text Relative</td>
<td>GPANR2</td>
</tr>
<tr>
<td>Annotation Text Relative 3</td>
<td>GPANR3</td>
</tr>
<tr>
<td>Application Data</td>
<td>GPINAD</td>
</tr>
<tr>
<td>Archive All Structures</td>
<td>GPARAS</td>
</tr>
<tr>
<td>Archive Structure Networks</td>
<td>GPARSN</td>
</tr>
<tr>
<td>Archive Structures</td>
<td>GPARST</td>
</tr>
<tr>
<td>Await Event</td>
<td>GPAWEV</td>
</tr>
<tr>
<td>Change Structure Identifier</td>
<td>GCSI</td>
</tr>
<tr>
<td>Change Structure Identifier And References</td>
<td>GCSIR</td>
</tr>
<tr>
<td>Change Structure References</td>
<td>GCSRS</td>
</tr>
<tr>
<td>Close Archive File</td>
<td>GCLAR</td>
</tr>
<tr>
<td>Close PHIGS</td>
<td>GCPH</td>
</tr>
<tr>
<td>Close Structure</td>
<td>GCLST</td>
</tr>
<tr>
<td>Close Workstation</td>
<td>GCLST</td>
</tr>
<tr>
<td>Compose Matrix</td>
<td>GCMT2</td>
</tr>
<tr>
<td>Compose Matrix 3</td>
<td>GCMT3</td>
</tr>
<tr>
<td>Copy All Elements From Structure</td>
<td>GCPST</td>
</tr>
<tr>
<td>Delete All Structures</td>
<td>GPDLE</td>
</tr>
<tr>
<td>Delete All Structures from Archive</td>
<td>GPDLE</td>
</tr>
<tr>
<td>Delete Element</td>
<td>GPDLE</td>
</tr>
<tr>
<td>Delete Element Range</td>
<td>GPDLEG</td>
</tr>
<tr>
<td>Delete Elements Between Labels</td>
<td>GPDLEG</td>
</tr>
<tr>
<td>Delete Structure</td>
<td>GPDLSLT</td>
</tr>
<tr>
<td>Delete Structure Network</td>
<td>GPDLN, GPDLNC</td>
</tr>
<tr>
<td>Delete Structure Networks from Archive</td>
<td>GPDNSA</td>
</tr>
<tr>
<td>Delete Structures from Archive</td>
<td>GPDASAR</td>
</tr>
<tr>
<td>Element Search</td>
<td>GPELS</td>
</tr>
<tr>
<td>Emergency Close PHIGS</td>
<td>GCLPH</td>
</tr>
<tr>
<td>Empty Structure</td>
<td>GPEST</td>
</tr>
<tr>
<td>Error Handling</td>
<td>GPEHND</td>
</tr>
<tr>
<td>Error Logging</td>
<td>GPELOG</td>
</tr>
<tr>
<td>Escape</td>
<td>GPE</td>
</tr>
<tr>
<td>Evaluate View Mapping Matrix</td>
<td>GPEVM2</td>
</tr>
<tr>
<td>Evaluate View Mapping Matrix 3</td>
<td>GPEVM3</td>
</tr>
<tr>
<td>Evaluate View Orientation Matrix</td>
<td>GCVMT, GPVPLN, GVR, GPVUP</td>
</tr>
<tr>
<td>Evaluate View Orientation Matrix 3</td>
<td>GCVMT, GPVPLN, GVR, GPVUP</td>
</tr>
<tr>
<td>Execute Structure</td>
<td>GEXST</td>
</tr>
<tr>
<td>Fill Area Set</td>
<td>GPPG2</td>
</tr>
<tr>
<td>Fill Area Set 3</td>
<td>GPPG3</td>
</tr>
<tr>
<td>Flush Device Events</td>
<td>GPFLEV</td>
</tr>
<tr>
<td>Get Choice</td>
<td>GPGTCH</td>
</tr>
</tbody>
</table>

© Copyright IBM Corp. 1994, 2002

503
<table>
<thead>
<tr>
<th>ISO PHIGS Subroutine Call</th>
<th>Associated GPxxxx Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Locator</td>
<td>GPGTLC</td>
</tr>
<tr>
<td>Get Locator 3</td>
<td>GPGTLC</td>
</tr>
<tr>
<td>Get Pick</td>
<td>GPGTPK</td>
</tr>
<tr>
<td>Get String</td>
<td>GPGTST</td>
</tr>
<tr>
<td>Get Stroke</td>
<td>GPGTST</td>
</tr>
<tr>
<td>Get Stroke 3</td>
<td>GPGTST</td>
</tr>
<tr>
<td>Get Valuator</td>
<td>GPGTVL</td>
</tr>
<tr>
<td>Initialize Choice 3</td>
<td>GPINCH</td>
</tr>
<tr>
<td>Initialize Locator 3</td>
<td>GPINLC</td>
</tr>
<tr>
<td>Initialize Pick 3</td>
<td>GPINPK</td>
</tr>
<tr>
<td>Initialize String 3</td>
<td>GPINST</td>
</tr>
<tr>
<td>Initialize Stroke 3</td>
<td>GPINSK</td>
</tr>
<tr>
<td>Initialize Valuator 3</td>
<td>GPINVL</td>
</tr>
<tr>
<td>Inquire All Conflicting Structures</td>
<td>GPQACA</td>
</tr>
<tr>
<td>Inquire Annotation Facilities</td>
<td>GPQANF, GPQXAF</td>
</tr>
<tr>
<td>Inquire Choice Device State</td>
<td>GPQCH</td>
</tr>
<tr>
<td>Inquire Choice Device State 3</td>
<td>GPQCH</td>
</tr>
<tr>
<td>Inquire Color Facilities</td>
<td>GPQLCF</td>
</tr>
<tr>
<td>Inquire Color Model</td>
<td>GPQCML</td>
</tr>
<tr>
<td>Inquire Color Representation</td>
<td>GPQXCR</td>
</tr>
<tr>
<td>Inquire Conflicting Structures in Network</td>
<td>GPQCNA</td>
</tr>
<tr>
<td>Inquire Current Element Content</td>
<td>GPQED</td>
</tr>
<tr>
<td>Inquire Current Element Type and Size</td>
<td>GPQEDH</td>
</tr>
<tr>
<td>Inquire Default Choice Device Data</td>
<td>GPQDCH</td>
</tr>
<tr>
<td>Inquire Default Choice Device Data 3</td>
<td>GPQDCH</td>
</tr>
<tr>
<td>Inquire Default Display Update State</td>
<td>GPQDDV</td>
</tr>
<tr>
<td>Inquire Default Locator Device Data</td>
<td>GPQDLC</td>
</tr>
<tr>
<td>Inquire Default Locator Device Data 3</td>
<td>GPQDLC</td>
</tr>
<tr>
<td>Inquire Default Pick Device Data</td>
<td>GPQDPK</td>
</tr>
<tr>
<td>Inquire Default Pick Device Data 3</td>
<td>GPQDPK</td>
</tr>
<tr>
<td>Inquire Default String Device Data</td>
<td>GPQDST</td>
</tr>
<tr>
<td>Inquire Default String Device Data 3</td>
<td>GPQDST</td>
</tr>
<tr>
<td>Inquire Default Stroke Device Data</td>
<td>GPQDSK</td>
</tr>
<tr>
<td>Inquire Default Stroke Device Data 3</td>
<td>GPQDSK</td>
</tr>
<tr>
<td>Inquire Default Valuator Device Data</td>
<td>GPQDVL</td>
</tr>
<tr>
<td>Inquire Default Valuator Device Data 3</td>
<td>GPQDVL</td>
</tr>
<tr>
<td>Inquire Display Space Size</td>
<td>GPQDS</td>
</tr>
<tr>
<td>Inquire Display Space Size 3</td>
<td>GPQDS</td>
</tr>
<tr>
<td>Inquire Display Update State</td>
<td>GPQDV</td>
</tr>
<tr>
<td>Inquire Edge Facilities</td>
<td>GPQEF</td>
</tr>
<tr>
<td>Inquire Edge Representation</td>
<td>GPQER</td>
</tr>
<tr>
<td>Inquire Edit Mode</td>
<td>GPQEDM</td>
</tr>
<tr>
<td>Inquire Element Content</td>
<td>GPQEDA</td>
</tr>
<tr>
<td>Inquire Element Pointer</td>
<td>GPQEP</td>
</tr>
<tr>
<td>Inquire Element Type and Size</td>
<td>GPQEHA</td>
</tr>
<tr>
<td>Inquire Error Handling Mode</td>
<td>GPQEMO</td>
</tr>
<tr>
<td>Inquire Highlighting Filter</td>
<td>GPOHLF</td>
</tr>
<tr>
<td>Inquire HLHSR Mode</td>
<td>GPQRVR GPQCVR</td>
</tr>
<tr>
<td>Inquire HLHSR(Mode) Facilities</td>
<td>GPQHMO</td>
</tr>
<tr>
<td>Inquire Input Queue Overflow</td>
<td>GPQIQO</td>
</tr>
<tr>
<td>Inquire Interior Facilities</td>
<td>GPQIF</td>
</tr>
<tr>
<td>Inquire Interior Representation</td>
<td>GPQXIR</td>
</tr>
<tr>
<td>Inquire Invisibility Filter</td>
<td>GPQIVF</td>
</tr>
<tr>
<td>Inquire List of Available Workstation Types</td>
<td>GPQWST</td>
</tr>
</tbody>
</table>
Table 2. ISO PHIGS Subroutines and Their Associated GPxxx Subroutine Calls (continued)

<table>
<thead>
<tr>
<th>ISO PHIGS Subroutine Call</th>
<th>Associated GPxxx Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inquire List of View Indices</td>
<td>GPQRVE</td>
</tr>
<tr>
<td>Inquire Locator Device State</td>
<td>GPQLC</td>
</tr>
<tr>
<td>Inquire Locator Device State 3</td>
<td>GPQLC</td>
</tr>
<tr>
<td>Inquire Modeling Clipping Facilities</td>
<td>GPWQD</td>
</tr>
<tr>
<td>Inquire More Simultaneous Events</td>
<td>GPQSEV</td>
</tr>
<tr>
<td>Inquire Number of Available Logical Input Devices</td>
<td>GPQLI</td>
</tr>
<tr>
<td>Inquire Number of Display Priorities Supported</td>
<td>GPQNSP</td>
</tr>
<tr>
<td>Inquire Open Structure</td>
<td>GPQOPS</td>
</tr>
<tr>
<td>Inquire Paths to Ancestors</td>
<td>GPQPAS</td>
</tr>
<tr>
<td>Inquire Paths to Descendants</td>
<td>GPQPDS</td>
</tr>
<tr>
<td>Inquire Pattern Facilities</td>
<td>GPQPAF</td>
</tr>
<tr>
<td>Inquire Pattern Representation</td>
<td>GPQPAR</td>
</tr>
<tr>
<td>Inquire Pick Device State</td>
<td>GPQPK</td>
</tr>
<tr>
<td>Inquire Pick Device State 3</td>
<td>GPQPK</td>
</tr>
<tr>
<td>Inquire Polyline Facilities</td>
<td>GPQPLF</td>
</tr>
<tr>
<td>Inquire Polyline Representation</td>
<td>GPQLR</td>
</tr>
<tr>
<td>Inquire Polymarker Facilities</td>
<td>GPQPMF</td>
</tr>
<tr>
<td>Inquire Polymarker Representation</td>
<td>GPQMR</td>
</tr>
<tr>
<td>Inquire Posted Structures</td>
<td>GPQRV</td>
</tr>
<tr>
<td>Inquire Predefined Color Representation</td>
<td>GPQPCR</td>
</tr>
<tr>
<td>Inquire Predefined Edge Representation</td>
<td>GPQPER</td>
</tr>
<tr>
<td>Inquire Predefined Interior Representation</td>
<td>GPQPIR</td>
</tr>
<tr>
<td>Inquire Predefined Pattern Representation</td>
<td>GPQPPR</td>
</tr>
<tr>
<td>Inquire Predefined Polyline Representation</td>
<td>GPQPLR</td>
</tr>
<tr>
<td>Inquire Predefined Polymarker Representation</td>
<td>GPQPMR</td>
</tr>
<tr>
<td>Inquire Predefined Text Representation</td>
<td>GPQPTR</td>
</tr>
<tr>
<td>Inquire Set of Open Workstations</td>
<td>GPQOPW</td>
</tr>
<tr>
<td>Inquire Set of Workstations To Which Posted</td>
<td>GPQWSA</td>
</tr>
<tr>
<td>Inquire String Device State</td>
<td>GPQST</td>
</tr>
<tr>
<td>Inquire String Device State 3</td>
<td>GPQST</td>
</tr>
<tr>
<td>Inquire Stroke Device State</td>
<td>GPQSK</td>
</tr>
<tr>
<td>Inquire Stroke Device State 3</td>
<td>GPQSK</td>
</tr>
<tr>
<td>Inquire Structure Identifiers</td>
<td>GPQSTI</td>
</tr>
<tr>
<td>Inquire Structure State Value</td>
<td>GPQSTV</td>
</tr>
<tr>
<td>Inquire Structure Status</td>
<td>GPQSTS</td>
</tr>
<tr>
<td>Inquire System State Value</td>
<td>GPQSTV</td>
</tr>
<tr>
<td>Inquire Text Facilities</td>
<td>GPQTXF, GPQGFC</td>
</tr>
<tr>
<td>Inquire Text Representation</td>
<td>GPQTR</td>
</tr>
<tr>
<td>Inquire Valuator Device State</td>
<td>GPQVL</td>
</tr>
<tr>
<td>Inquire Valuator Device State 3</td>
<td>GPQVL</td>
</tr>
<tr>
<td>Inquire View Representation</td>
<td>GPQCVR, GPQVR</td>
</tr>
<tr>
<td>Inquire Workstation Category</td>
<td>GPWQC</td>
</tr>
<tr>
<td>Inquire Workstation Classification</td>
<td>GPQWD</td>
</tr>
<tr>
<td>Inquire Workstation Connection and Type</td>
<td>GPQRCT</td>
</tr>
<tr>
<td>Inquire Workstation State Table Lengths</td>
<td>GPQLW</td>
</tr>
<tr>
<td>Inquire Workstation State Value</td>
<td>GPQWSV</td>
</tr>
<tr>
<td>Inquire Workstation Transformation</td>
<td>GPQWSX</td>
</tr>
<tr>
<td>Inquire Workstation Transformation 3</td>
<td>GPQWSX</td>
</tr>
<tr>
<td>Label</td>
<td>GPINLB</td>
</tr>
<tr>
<td>Message</td>
<td>GPMSG</td>
</tr>
<tr>
<td>Offset Element Pointer</td>
<td>GPOEP</td>
</tr>
<tr>
<td>Open Archive File</td>
<td>GOPPAR</td>
</tr>
<tr>
<td>Open PHIGS</td>
<td>GOPPH</td>
</tr>
<tr>
<td>Open Structure</td>
<td>GOPST</td>
</tr>
<tr>
<td>ISO PHIGS Subroutine Call</td>
<td>Associated GPxxxx Subroutine Call(s)</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Open Workstation</td>
<td>GPOPWS</td>
</tr>
<tr>
<td>Pack Data Record</td>
<td>GPPREC</td>
</tr>
<tr>
<td>Polyl ine</td>
<td>GPPL2</td>
</tr>
<tr>
<td>Polyl ine 3</td>
<td>GPPL3</td>
</tr>
<tr>
<td>Polymarker</td>
<td>GPPM2</td>
</tr>
<tr>
<td>Polymarker 3</td>
<td>GPPM3</td>
</tr>
<tr>
<td>Post Structure</td>
<td>GPARV</td>
</tr>
<tr>
<td>Redraw All Structures</td>
<td>GPRAST</td>
</tr>
<tr>
<td>Remove Names from Set</td>
<td>GPRCN</td>
</tr>
<tr>
<td>Request Choice</td>
<td>GPRQCH</td>
</tr>
<tr>
<td>Request Locator</td>
<td>GPRQLC</td>
</tr>
<tr>
<td>Request Locator 3</td>
<td>GPRQLC</td>
</tr>
<tr>
<td>Request Pick</td>
<td>GPRQPK</td>
</tr>
<tr>
<td>Request String</td>
<td>GPRQST</td>
</tr>
<tr>
<td>Request Stroke</td>
<td>GPRQSK</td>
</tr>
<tr>
<td>Request Stroke 3</td>
<td>GPRQSK</td>
</tr>
<tr>
<td>Request Valuator</td>
<td>GPRQVL</td>
</tr>
<tr>
<td>Restore Modeling Clipping Volume</td>
<td>GPRMCV</td>
</tr>
<tr>
<td>Retrieve All Structures</td>
<td>GPRVAS</td>
</tr>
<tr>
<td>Retrieve Paths to Ancestors</td>
<td>GPQPAS</td>
</tr>
<tr>
<td>Retrieve Paths to Descendants</td>
<td>GPQPDS</td>
</tr>
<tr>
<td>Retrieve Structure Identifiers</td>
<td>GPRSTI</td>
</tr>
<tr>
<td>Retrieve Structure Networks</td>
<td>GPRVSN</td>
</tr>
<tr>
<td>Retrieve Structures</td>
<td>GPRVST</td>
</tr>
<tr>
<td>Rotate</td>
<td>GPROT2</td>
</tr>
<tr>
<td>Rotate X</td>
<td>GPROTX</td>
</tr>
<tr>
<td>Rotate Y</td>
<td>GPROTY</td>
</tr>
<tr>
<td>Rotate Z</td>
<td>GPROTZ</td>
</tr>
<tr>
<td>Sample Choice</td>
<td>GPSMCH</td>
</tr>
<tr>
<td>Sample Locator</td>
<td>GPSMLC</td>
</tr>
<tr>
<td>Sample Locator 3</td>
<td>GPSMLC</td>
</tr>
<tr>
<td>Sample Pick</td>
<td>GPSMPK</td>
</tr>
<tr>
<td>Sample String</td>
<td>GPSMST</td>
</tr>
<tr>
<td>Sample Stroke</td>
<td>GPSMSK</td>
</tr>
<tr>
<td>Sample Stroke 3</td>
<td>GPSMSK</td>
</tr>
<tr>
<td>Sample Valuator</td>
<td>GPSMVL</td>
</tr>
<tr>
<td>Scale</td>
<td>GPSC2</td>
</tr>
<tr>
<td>Scale 3</td>
<td>GPSC3</td>
</tr>
<tr>
<td>Set Annotation Text Alignment</td>
<td>GPAAL</td>
</tr>
<tr>
<td>Set Annotation Text Character Height</td>
<td>GPAH</td>
</tr>
<tr>
<td>Set Annotation Text Character Up Vector</td>
<td>GPAUP</td>
</tr>
<tr>
<td>Set Annotation Text Path</td>
<td>GPAAPT</td>
</tr>
<tr>
<td>Set Annotation Text Style</td>
<td>GPAS</td>
</tr>
<tr>
<td>Set Character Expansion Factor</td>
<td>GPCCHXP</td>
</tr>
<tr>
<td>Set Character Height</td>
<td>GPCCHH</td>
</tr>
<tr>
<td>Set Character Spacing</td>
<td>GPCCHSP</td>
</tr>
<tr>
<td>Set Character Up Vector</td>
<td>GPCCHUP</td>
</tr>
<tr>
<td>Set Choice Mode</td>
<td>GPCHMO</td>
</tr>
<tr>
<td>Set Color Model</td>
<td>GPCML</td>
</tr>
<tr>
<td>Set Color Representation</td>
<td>GPCR</td>
</tr>
<tr>
<td>Set Conflict Resolution</td>
<td>GPCNRS</td>
</tr>
<tr>
<td>Set Display Update State</td>
<td>GPDF</td>
</tr>
<tr>
<td>Set Edge Color Index</td>
<td>GPECI</td>
</tr>
<tr>
<td>Set Edge Flag</td>
<td>GPEF</td>
</tr>
<tr>
<td>ISO PHIGS Subroutine Call</td>
<td>Associated GPxxxx Subroutine Call(s)</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Set Edge Index</td>
<td>GPEI</td>
</tr>
<tr>
<td>Set Edge Representation</td>
<td>GPER</td>
</tr>
<tr>
<td>Set Edgetype</td>
<td>GPELT</td>
</tr>
<tr>
<td>Set Edgewidth Scale Factor</td>
<td>GPESC</td>
</tr>
<tr>
<td>Set Edit Mode</td>
<td>GPEDMO</td>
</tr>
<tr>
<td>Set Element Pointer</td>
<td>GPEP</td>
</tr>
<tr>
<td>Set Element Pointer at Label</td>
<td>GPEPLG</td>
</tr>
<tr>
<td>Set Error Handling Mode</td>
<td>GPEMO</td>
</tr>
<tr>
<td>Set Global Transformation</td>
<td>GPGLX2</td>
</tr>
<tr>
<td>Set Global Transformation 3</td>
<td>GPGLX3</td>
</tr>
<tr>
<td>Set Highlighting Filter</td>
<td>GPHLF</td>
</tr>
<tr>
<td>Set HLHSR Identifier</td>
<td>GPHID</td>
</tr>
<tr>
<td>Set HLHSR Mode</td>
<td>GPXVR</td>
</tr>
<tr>
<td>Set Individual ASF</td>
<td>GPASF</td>
</tr>
<tr>
<td>Set Interior Color Index</td>
<td>GPICI</td>
</tr>
<tr>
<td>Set Interior Index</td>
<td>GPII</td>
</tr>
<tr>
<td>Set Interior Representation</td>
<td>GPIR</td>
</tr>
<tr>
<td>Set Interior Style</td>
<td>GPIS</td>
</tr>
<tr>
<td>Set Interior Style Index</td>
<td>GPISI</td>
</tr>
<tr>
<td>Set Invisibility Filter</td>
<td>GPIVF</td>
</tr>
<tr>
<td>Set Linetype</td>
<td>GPLT</td>
</tr>
<tr>
<td>Set Linewidth Scale Factor</td>
<td>GPLWSC</td>
</tr>
<tr>
<td>Set Local Transformation</td>
<td>GPMLX2</td>
</tr>
<tr>
<td>Set Local Transformation 3</td>
<td>GPMLX3</td>
</tr>
<tr>
<td>Set Locator Mode</td>
<td>GPLCMO</td>
</tr>
<tr>
<td>Set Marker Size Scale Factor</td>
<td>GPMSSC</td>
</tr>
<tr>
<td>Set Marker Type</td>
<td>GPMT</td>
</tr>
<tr>
<td>Set Modeling Clipping Indicator</td>
<td>GPMCI</td>
</tr>
<tr>
<td>Set Modeling Clipping Volume</td>
<td>GPMCV2</td>
</tr>
<tr>
<td>Set Modeling Clipping Volume 3</td>
<td>GPMCV3</td>
</tr>
<tr>
<td>Set Pattern Representation</td>
<td>GPPAR</td>
</tr>
<tr>
<td>Set Pick Filter</td>
<td>GPPKF</td>
</tr>
<tr>
<td>Set Pick Identifier</td>
<td>GPPKID</td>
</tr>
<tr>
<td>Set Pick Mode</td>
<td>GPPKMO</td>
</tr>
<tr>
<td>Set Polyline Color Index</td>
<td>GPPLCI</td>
</tr>
<tr>
<td>Set Polyline Index</td>
<td>GPPLI</td>
</tr>
<tr>
<td>Set Polyline Representation</td>
<td>GPPLR</td>
</tr>
<tr>
<td>Set Polymarker Color Index</td>
<td>GPPMCI</td>
</tr>
<tr>
<td>Set Polymarker Index</td>
<td>GPPMI</td>
</tr>
<tr>
<td>Set Polymarker Representation</td>
<td>GPPMR</td>
</tr>
<tr>
<td>Set String Mode</td>
<td>GPSTMO</td>
</tr>
<tr>
<td>Set Stroke Mode</td>
<td>GPSKMO</td>
</tr>
<tr>
<td>Set Text Alignment</td>
<td>GPTXAL</td>
</tr>
<tr>
<td>Set Text Color Index</td>
<td>GPTXCI</td>
</tr>
<tr>
<td>Set Text Font</td>
<td>GPTXFO</td>
</tr>
<tr>
<td>Set Text Index</td>
<td>GPTXI</td>
</tr>
<tr>
<td>Set Text Path</td>
<td>GPTXPT</td>
</tr>
<tr>
<td>Set Text Precision</td>
<td>GPTXPR</td>
</tr>
<tr>
<td>Set Text Representation</td>
<td>GPTXR</td>
</tr>
<tr>
<td>Set Valuator Mode</td>
<td>GPVLMO</td>
</tr>
<tr>
<td>Set View Index</td>
<td>GPVWI</td>
</tr>
<tr>
<td>Set View Representation</td>
<td>GPXVR</td>
</tr>
<tr>
<td>Set View Representation 3</td>
<td>GPXVR</td>
</tr>
<tr>
<td>Set View Transformation Input Priority</td>
<td>GPVIP</td>
</tr>
<tr>
<td>ISO PHIGS Subroutine Call</td>
<td>Associated GPxxxx Subroutine Call(s)</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Set Workstation Viewport</td>
<td>GPWSX2</td>
</tr>
<tr>
<td>Set Workstation Viewport 3</td>
<td>GPWSX3</td>
</tr>
<tr>
<td>Set Workstation Window</td>
<td>GPWSX2</td>
</tr>
<tr>
<td>Set Workstation Window 3</td>
<td>GPWSX3</td>
</tr>
<tr>
<td>Text</td>
<td>GPTX2</td>
</tr>
<tr>
<td>Text 3</td>
<td>GPTX3</td>
</tr>
<tr>
<td>Transform Point</td>
<td>GPXF2</td>
</tr>
<tr>
<td>Transform Point 3</td>
<td>GPXF3</td>
</tr>
<tr>
<td>Translate</td>
<td>GPTRL2</td>
</tr>
<tr>
<td>Translate 3</td>
<td>GPTRL3</td>
</tr>
<tr>
<td>Unpost All Structures</td>
<td>GPDARW</td>
</tr>
<tr>
<td>Unpost Structure</td>
<td>GPDRW</td>
</tr>
<tr>
<td>Update Workstation</td>
<td>GPUPWS</td>
</tr>
</tbody>
</table>

Table 2. ISO PHIGS Subroutines and Their Associated GPxxxx Subroutine Calls (continued)
Chapter 21. GPxxxx Subroutines to ISO PHIGS Subroutines

The following table associates GPxxxx subroutine calls with equivalent or approximately equivalent ISO PHIGS subroutines. Note that the calls may be approximately equivalent; there may be some variation in the call interface or in the functionality. Refer to _The graPHIGS Programming Interface: Subroutine Reference_ and the ISO PHIGS subroutines for details on individual calls.

The tables are arranged alphabetically by GPxxxx subroutine calls. A missing GPxxxx subroutine call implies that there is no approximately equivalent ISO PHIGS subroutine.

*Table 3. GPxxxx Subroutine Calls and Their Associated ISO PHIGS Subroutines*

<table>
<thead>
<tr>
<th>xxxx Subroutine Call</th>
<th>xxxx Subroutine Call Description</th>
<th>Associated ISO PHIGS Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPAAL</td>
<td>Set Annotation Alignment</td>
<td>Set Annotation Text Alignment</td>
</tr>
<tr>
<td>GPADCN</td>
<td>Add Class Name to Set</td>
<td>Add Names to Set</td>
</tr>
<tr>
<td>GPAH</td>
<td>Set Annotation Height</td>
<td>Set Annotation Text Character Height</td>
</tr>
<tr>
<td>GPAN2</td>
<td>Annotation Text 2</td>
<td>Annotation Text Relative</td>
</tr>
<tr>
<td>GPAN3</td>
<td>Annotation Text 3</td>
<td>Annotation Text Relative 3</td>
</tr>
<tr>
<td>GPANR2</td>
<td>Annotation Text Relative 2</td>
<td>Annotation Text Relative 3</td>
</tr>
<tr>
<td>GPANR3</td>
<td>Annotation Text Relative 3</td>
<td>Annotation Text Relative 3</td>
</tr>
<tr>
<td>GAPT</td>
<td>Set Annotation Path</td>
<td>Set Annotation Text Path</td>
</tr>
<tr>
<td>GPARAS</td>
<td>Archive All Structures</td>
<td>Archive All Structures</td>
</tr>
<tr>
<td>GPARSN</td>
<td>Archive Structure Networks</td>
<td>Archive Structure Networks</td>
</tr>
<tr>
<td>GPARST</td>
<td>Archive Structures</td>
<td>Archive Structures</td>
</tr>
<tr>
<td>GPARV</td>
<td>Associate Root with View</td>
<td>Post Structure</td>
</tr>
<tr>
<td>GNAS</td>
<td>Set Annotation Style</td>
<td>Set Annotation Style</td>
</tr>
<tr>
<td>GPASF</td>
<td>Set Aspect Source Flag</td>
<td>Set Individual ASF</td>
</tr>
<tr>
<td>GPAUP</td>
<td>Set Annotation Up Vector</td>
<td>Set Annotation Text Character Up Vector</td>
</tr>
<tr>
<td>GPAWEV</td>
<td>Await Event</td>
<td>Await Event</td>
</tr>
<tr>
<td>GPCCHH</td>
<td>Set Character Height</td>
<td>Set Character Height</td>
</tr>
<tr>
<td>GPCCHMO</td>
<td>Set Choice Mode</td>
<td>Set Choice Mode</td>
</tr>
<tr>
<td>GPCCHSP</td>
<td>Set Character Spacing</td>
<td>Set Character Spacing</td>
</tr>
<tr>
<td>GPCCHUP</td>
<td>Set Character Up Vector</td>
<td>Set Character Up Vector</td>
</tr>
<tr>
<td>GPCCHXP</td>
<td>Set Character Expansion Factor</td>
<td>Set Character Expansion Factor</td>
</tr>
<tr>
<td>GPCLAR</td>
<td>Close Archive File</td>
<td>Close Archive File</td>
</tr>
<tr>
<td>GPCLPH</td>
<td>Close graPHIGS</td>
<td>Close PHIGS, Emergency Close PHIGS</td>
</tr>
<tr>
<td>GPCLST</td>
<td>Close Structure</td>
<td>Close Structure</td>
</tr>
<tr>
<td>GPCLWS</td>
<td>Close Workstation</td>
<td>Close Workstation</td>
</tr>
<tr>
<td>GPCLCH</td>
<td>Set Color Model</td>
<td>Set Color Model</td>
</tr>
<tr>
<td>GCPMT2</td>
<td>Compose Matrix 2</td>
<td>Compose Matrix</td>
</tr>
<tr>
<td>GCPMT3</td>
<td>Compose Matrix 3</td>
<td>Compose Matrix 3</td>
</tr>
<tr>
<td>GPCNRS</td>
<td>Set Conflict Resolution</td>
<td>Set Conflict Resolution</td>
</tr>
<tr>
<td>GCPST</td>
<td>Copy Structure</td>
<td>Copy All Elements from Structure</td>
</tr>
<tr>
<td>GPCR</td>
<td>Set Color Representation</td>
<td>Set Color Representation</td>
</tr>
<tr>
<td>GCRS</td>
<td>Create Workstation</td>
<td>Open Workstation</td>
</tr>
<tr>
<td>GCSII</td>
<td>Change Structure Identifier</td>
<td>Change Structure Identifier</td>
</tr>
<tr>
<td>GCSIR</td>
<td>Change Structure Identifier and References</td>
<td>Change Structure Identifier and References</td>
</tr>
<tr>
<td>GCSRS</td>
<td>Change Structure References</td>
<td>Change Structure References</td>
</tr>
<tr>
<td>GCVVMT</td>
<td>Compute View Matrix</td>
<td>Evaluate View Orientation Matrix 3</td>
</tr>
<tr>
<td>GPDARW</td>
<td>Disassociate All Roots from Workstation</td>
<td>Unpost All Structures</td>
</tr>
<tr>
<td>GPDASA</td>
<td>Delete All Structures from Archive</td>
<td>Delete All Structures from Archive</td>
</tr>
</tbody>
</table>
Table 3. GPxxxx Subroutine Calls and Their Associated ISO PHIGS Subroutines (continued)

<table>
<thead>
<tr>
<th>xxxx Subroutine Call</th>
<th>xxxx Subroutine Call Description</th>
<th>Associated ISO PHIGS Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDPAST</td>
<td>Delete All Structures</td>
<td>Delete All Structures</td>
</tr>
<tr>
<td>GPDELB</td>
<td>Delete Element Between Labels</td>
<td>Delete Elements Between Labels</td>
</tr>
<tr>
<td>GPDF</td>
<td>Set Deferral State</td>
<td>Set Display Update State</td>
</tr>
<tr>
<td>GPDE</td>
<td>Delete Element</td>
<td>Delete Element</td>
</tr>
<tr>
<td>GPDELEG</td>
<td>Delete Element Group</td>
<td>Delete Elements Between Labels</td>
</tr>
<tr>
<td>GPDLR</td>
<td>Delete Element Range</td>
<td>Delete Element Range</td>
</tr>
<tr>
<td>GPDLNC</td>
<td>Delete Structure Network Conditionally</td>
<td>Delete Structure Network</td>
</tr>
<tr>
<td>GPDLNT</td>
<td>Delete Structure Network</td>
<td>Delete Structure Network</td>
</tr>
<tr>
<td>GPDLST</td>
<td>Delete Structure</td>
<td>Delete Structure</td>
</tr>
<tr>
<td>GPDRAV</td>
<td>Disassociate Root from All Views</td>
<td>Unpost Structure</td>
</tr>
<tr>
<td>GPDKV</td>
<td>Disassociate Root from View</td>
<td>Unpost Structure</td>
</tr>
<tr>
<td>GPDKW</td>
<td>Disassociate Root from Workstation</td>
<td>Unpost Structure</td>
</tr>
<tr>
<td>GPDSAR</td>
<td>Delete Structures from Archive</td>
<td>Delete Structures from Archive</td>
</tr>
<tr>
<td>GPDSNa</td>
<td>Delete Structure Networks from Archive</td>
<td>Delete Structure Networks from Archive</td>
</tr>
<tr>
<td>GPEAV</td>
<td>Empty All Views</td>
<td>Unpost All Structures</td>
</tr>
<tr>
<td>GPECI</td>
<td>Set Edge Color Index</td>
<td>Set Edge Color Index</td>
</tr>
<tr>
<td>GPEDMO</td>
<td>Set Edit Mode</td>
<td>Set Edit Mode</td>
</tr>
<tr>
<td>GPEF</td>
<td>Set Edge Flag</td>
<td>Set Edge Flag</td>
</tr>
<tr>
<td>GPEHND</td>
<td>Set Error Handling Function</td>
<td>Error Handling</td>
</tr>
<tr>
<td>GPEI</td>
<td>Set Edge Index</td>
<td>Set Edge Index</td>
</tr>
<tr>
<td>GPELOG</td>
<td>Error Logging</td>
<td>Error Logging</td>
</tr>
<tr>
<td>GPELS</td>
<td>Element Search</td>
<td>Element Search</td>
</tr>
<tr>
<td>GPELT</td>
<td>Set Edge Linetype</td>
<td>Set Edgetype</td>
</tr>
<tr>
<td>GPEMO</td>
<td>Set Error Handling Mode</td>
<td>Set Error Handling Mode</td>
</tr>
<tr>
<td>GPEP</td>
<td>Set Element Pointer</td>
<td>Set Element Pointer</td>
</tr>
<tr>
<td>GPEPLB</td>
<td>Set Element Pointer at Label</td>
<td>Set Element Pointer at Label</td>
</tr>
<tr>
<td>GPEPLG</td>
<td>Generalized Set Element Pointer at Label</td>
<td>Set Element Pointer at Label</td>
</tr>
<tr>
<td>GPER</td>
<td>Set Edge Representation</td>
<td>Set Edge Representation</td>
</tr>
<tr>
<td>GRES</td>
<td>Escape</td>
<td>Escape</td>
</tr>
<tr>
<td>GPESC</td>
<td>Set Edge Scale Factor</td>
<td>Set Edgewidth Scale Factor</td>
</tr>
<tr>
<td>GPEST</td>
<td>Empty Structure</td>
<td>Empty Structure</td>
</tr>
<tr>
<td>GPEV</td>
<td>Empty View</td>
<td>Unpost All Structures</td>
</tr>
<tr>
<td>GPEVM2</td>
<td>Evaluate View Mapping Matrix 2</td>
<td>Evaluate View Mapping Matrix 2</td>
</tr>
<tr>
<td>GPEVM3</td>
<td>Evaluate View Mapping Matrix 3</td>
<td>Evaluate View Mapping Matrix 3</td>
</tr>
<tr>
<td>GPEXST</td>
<td>Execute Structure</td>
<td>Execute Structure</td>
</tr>
<tr>
<td>GPFLAV</td>
<td>Flush Device Events</td>
<td>Flush Device Events</td>
</tr>
<tr>
<td>GPGLX2</td>
<td>Set Global Transformation 2</td>
<td>Set Global Transformation 2</td>
</tr>
<tr>
<td>GPGLX3</td>
<td>Set Global Transformation 3</td>
<td>Set Global Transformation 3</td>
</tr>
<tr>
<td>GPGTCH</td>
<td>Get Choice</td>
<td>Get Choice</td>
</tr>
<tr>
<td>GPGTLC</td>
<td>Get Locator</td>
<td>Get Locator</td>
</tr>
<tr>
<td>GPGTPK</td>
<td>Get Pick</td>
<td>Get Pick</td>
</tr>
<tr>
<td>GPGTSH</td>
<td>Get Stroke</td>
<td>Get Stroke</td>
</tr>
<tr>
<td>GPGTST</td>
<td>Get String</td>
<td>Get String</td>
</tr>
<tr>
<td>GPGTVL</td>
<td>Get Valuator</td>
<td>Get Valuator</td>
</tr>
<tr>
<td>GPHID</td>
<td>Set HLHLSR Identifier</td>
<td>Set HLHLSR Identifier</td>
</tr>
<tr>
<td>GPHLF</td>
<td>Set Highlighting Filter</td>
<td>Set Highlighting Filter</td>
</tr>
<tr>
<td>GPICT</td>
<td>Set Interior Color Index</td>
<td>Set Interior Color Index</td>
</tr>
<tr>
<td>GPII</td>
<td>Set Interior Index</td>
<td>Set Interior Index</td>
</tr>
<tr>
<td>GPINADM</td>
<td>Insert Application Data</td>
<td>Application Data</td>
</tr>
<tr>
<td>GPINCH</td>
<td>Initialize Choice</td>
<td>Initialize Choice</td>
</tr>
<tr>
<td>GPINLB</td>
<td>Insert Label</td>
<td>Label</td>
</tr>
<tr>
<td>Subroutine</td>
<td>Subroutine Call Description</td>
<td>Associated ISO PHIGS Subroutine Call(s)</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>GPINLC</td>
<td>Initialize Locator</td>
<td>Initialize Locator 3</td>
</tr>
<tr>
<td>GPINPK</td>
<td>Initialize Pick</td>
<td>Initialize Pick 3</td>
</tr>
<tr>
<td>GPINSK</td>
<td>Initialize Stroke</td>
<td>Initialize Stroke 3</td>
</tr>
<tr>
<td>GPINST</td>
<td>Initialize String</td>
<td>Initialize String 3</td>
</tr>
<tr>
<td>GPINVL</td>
<td>Initialize Valuator</td>
<td>Initialize Valuator 3</td>
</tr>
<tr>
<td>GPIR</td>
<td>Set Interior Representation</td>
<td>Set Interior Representation</td>
</tr>
<tr>
<td>GPIIS</td>
<td>Set Interior Style</td>
<td>Set Interior Style</td>
</tr>
<tr>
<td>GPIISI</td>
<td>Set Interior Style Index</td>
<td>Set Interior Style Index</td>
</tr>
<tr>
<td>GPIVF</td>
<td>Set Invisibility Filter</td>
<td>Set Invisibility Filter</td>
</tr>
<tr>
<td>GPLCMMO</td>
<td>Set Locatoir Mode</td>
<td>Set Locatoir Mode</td>
</tr>
<tr>
<td>GPLT</td>
<td>Set Linetype</td>
<td>Set Linetype</td>
</tr>
<tr>
<td>GPLWSC</td>
<td>Set Linewidth Scale Factor</td>
<td>Set Linewidth Scale Factor</td>
</tr>
<tr>
<td>GPMCI</td>
<td>Set Modeling Clipping Indicator</td>
<td>Set Modeling Clipping Indicator</td>
</tr>
<tr>
<td>GPMCV2</td>
<td>Set Modeling Clipping Volume 2</td>
<td>Set Modeling Clipping Volume</td>
</tr>
<tr>
<td>GPMVC3</td>
<td>Set Modeling Clipping Volume 3</td>
<td>Set Modeling Clipping Volume</td>
</tr>
<tr>
<td>GMLX2</td>
<td>Set Modeling Transformation 2</td>
<td>Set Local Transformation</td>
</tr>
<tr>
<td>GMLX3</td>
<td>Set Modeling Transformation 3</td>
<td>Set Local Transformation 3</td>
</tr>
<tr>
<td>GMSG</td>
<td>Message</td>
<td>Message</td>
</tr>
<tr>
<td>GMSSC</td>
<td>Set Marker Size Scale Factor</td>
<td>Set Marker Size Scale Factor</td>
</tr>
<tr>
<td>GPMT</td>
<td>Set Marker Type</td>
<td>Set Marker Type</td>
</tr>
<tr>
<td>GPOEP</td>
<td>Offset Element Pointer</td>
<td>Offset Element Pointer</td>
</tr>
<tr>
<td>GPOPAR</td>
<td>Open Archive File</td>
<td>Open Archive File</td>
</tr>
<tr>
<td>GPOPH</td>
<td>Open graPHIGS</td>
<td>Open PHIGS</td>
</tr>
<tr>
<td>GPOPST</td>
<td>Open Structure</td>
<td>Open Structure</td>
</tr>
<tr>
<td>GPOPSW</td>
<td>Open Workstation</td>
<td>Open Workstation</td>
</tr>
<tr>
<td>GPPAR</td>
<td>Set Pattern Representation</td>
<td>Set Pattern Representation</td>
</tr>
<tr>
<td>GPPG2</td>
<td>Polygon 2</td>
<td>Fill Area Set</td>
</tr>
<tr>
<td>GPPG3</td>
<td>Polygon 3</td>
<td>Fill Area Set 3</td>
</tr>
<tr>
<td>GPPKF</td>
<td>Set Pick Filter</td>
<td>Set Pick Filter</td>
</tr>
<tr>
<td>GPPKID</td>
<td>Set Pick Identifier</td>
<td>Set Pick Identifier</td>
</tr>
<tr>
<td>GPPKMO</td>
<td>Set Pick Mode</td>
<td>Set Pick Mode</td>
</tr>
<tr>
<td>GPPLCI</td>
<td>Set Polyl ine Color Index</td>
<td>Set Polyl ine Color Index</td>
</tr>
<tr>
<td>GPPLI</td>
<td>Set Polyl ine Index</td>
<td>Set Polyl ine Index</td>
</tr>
<tr>
<td>GPPLR</td>
<td>Set Polyl ine Representation</td>
<td>Set Polyl ine Representation</td>
</tr>
<tr>
<td>GPPL2</td>
<td>Polyl ine 2</td>
<td>Polyl ine</td>
</tr>
<tr>
<td>GPPL3</td>
<td>Polyl ine 3</td>
<td>Polyl ine</td>
</tr>
<tr>
<td>GPMCI</td>
<td>Set Polylarker Color Index</td>
<td>Set Polylarker Color Index</td>
</tr>
<tr>
<td>GPMI</td>
<td>Set Polylarker Index</td>
<td>Set Polylarker Index</td>
</tr>
<tr>
<td>GPMR</td>
<td>Set Polylarker Representation</td>
<td>Set Polylarker Representation</td>
</tr>
<tr>
<td>GPM2</td>
<td>Polylarker 2</td>
<td>Polylarker</td>
</tr>
<tr>
<td>GPM3</td>
<td>Polylarker 3</td>
<td>Polylarker</td>
</tr>
<tr>
<td>GPREC</td>
<td>Pack Data Record</td>
<td>Pack Data Record</td>
</tr>
<tr>
<td>GQACA</td>
<td>Inquire All Conflicting Structures in Archive</td>
<td>Inquire All Conflicting Structures in Archive</td>
</tr>
<tr>
<td>GQANF</td>
<td>Inquire Annotation Facilities</td>
<td>Inquire Annotation Facilities</td>
</tr>
<tr>
<td>GQARF</td>
<td>Inquire Archive Files</td>
<td>Inquire Archive Files</td>
</tr>
<tr>
<td>GQCF</td>
<td>Inquire Color Facilities</td>
<td>Inquire Color Facilities</td>
</tr>
<tr>
<td>GQCCH</td>
<td>Inquire Choice Device State</td>
<td>Inquire Choice Device State 3</td>
</tr>
<tr>
<td>GQCMCL</td>
<td>Inquire Color Model</td>
<td>Inquire Color Model</td>
</tr>
<tr>
<td>GQCNA</td>
<td>Inquire Conflicting Structures in Network in Archive</td>
<td>Inquire Conflicting Structures in Network in Archive</td>
</tr>
<tr>
<td>GQCNR</td>
<td>Inquire Conflict Resolution</td>
<td>Inquire Conflict Resolution</td>
</tr>
<tr>
<td>GQCR</td>
<td>Inquire Color Representation</td>
<td>Inquire Color Representation</td>
</tr>
<tr>
<td>xxxx Subroutine Call</td>
<td>xxxx Subroutine Call Description</td>
<td>Associated ISO PHIGS Subroutine Call(s)</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>GPOCVE</td>
<td>Inquire Current View Table Entries</td>
<td>Inquire List of View Indices</td>
</tr>
<tr>
<td>GPOCVR</td>
<td>Inquire Current View Representation</td>
<td>Inquire View Representation</td>
</tr>
<tr>
<td>GPOCVX</td>
<td>Inquire Current Viewing Transformation</td>
<td>Inquire View Representation</td>
</tr>
<tr>
<td>GPQDCH</td>
<td>Inquire Default Choice Device Data</td>
<td>Inquire Default Choice Device Data 3</td>
</tr>
<tr>
<td>GPQDDV</td>
<td>Inquire Default Deferral State Values</td>
<td>Inquire Default Display Update State</td>
</tr>
<tr>
<td>GPQDLC</td>
<td>Inquire Default Locator Device Data</td>
<td>Inquire Default Locator Device Data 3</td>
</tr>
<tr>
<td>GPQDPK</td>
<td>Inquire Default Pick Device Data</td>
<td>Inquire Default Pick Device Data 3</td>
</tr>
<tr>
<td>GPQDS</td>
<td>Inquire Maximum Display Surface Size</td>
<td>Inquire Display Space Size</td>
</tr>
<tr>
<td>GPQDSK</td>
<td>Inquire Default Stroke Device Data</td>
<td>Inquire Default Stroke Device Data 3</td>
</tr>
<tr>
<td>GPQDST</td>
<td>Inquire Default String Device Data</td>
<td>Inquire Default String Device Data 3</td>
</tr>
<tr>
<td>GPQDV</td>
<td>Inquire Deferral and Update State Value</td>
<td>Inquire Display Update State</td>
</tr>
<tr>
<td>GPQDVL</td>
<td>Inquire Default Valuator Device Data</td>
<td>Inquire Default Valuator Device Data 3</td>
</tr>
<tr>
<td>GPQE</td>
<td>Inquire Element Content</td>
<td>Inquire Current Element Content</td>
</tr>
<tr>
<td>GPQED</td>
<td>Inquire Element Data</td>
<td>Inquire Current Element Content</td>
</tr>
<tr>
<td>GPQEDA</td>
<td>Inquire List of Element Data For Any Structure</td>
<td>Inquire Element Content</td>
</tr>
<tr>
<td>GPQEDM</td>
<td>Inquire Edit Mode</td>
<td>Inquire Edit Mode</td>
</tr>
<tr>
<td>GPQEF</td>
<td>Inquire Edge Facilities</td>
<td>Inquire Edge Facilities</td>
</tr>
<tr>
<td>GPQEHA</td>
<td>Inquire List of Element Headers For Any Structure</td>
<td>Inquire Element Type and Size</td>
</tr>
<tr>
<td>GPQEHBD</td>
<td>Inquire List of Element Headers</td>
<td>Inquire Current Element Type and Size</td>
</tr>
<tr>
<td>GPQEMO</td>
<td>Inquire Error Handling Mode</td>
<td>Inquire Error Handling Mode</td>
</tr>
<tr>
<td>GPQEP</td>
<td>Inquire Element Pointer</td>
<td>Inquire Element Pointer</td>
</tr>
<tr>
<td>GPQER</td>
<td>Inquire Edge Representation</td>
<td>Inquire Edge Representation</td>
</tr>
<tr>
<td>GPQETS</td>
<td>Inquire Element Type and Size</td>
<td>Inquire Current Element Type and Size</td>
</tr>
<tr>
<td>GPQEXS</td>
<td>Inquire Executed Structures</td>
<td>Inquire Paths to Descendants</td>
</tr>
<tr>
<td>GPQGF</td>
<td>Inquire Geometric Font Characteristics</td>
<td>Inquire Text Facilities</td>
</tr>
<tr>
<td>GPQHLF</td>
<td>Inquire Highlighting Filter</td>
<td>Inquire Highlighting Filter</td>
</tr>
<tr>
<td>GPQHMO</td>
<td>Inquire HLHSR Modes</td>
<td>Inquire HLHSR (Mode) Facilities</td>
</tr>
<tr>
<td>GPQIF</td>
<td>Inquire Interior Facilities</td>
<td>Inquire Interior Facilities</td>
</tr>
<tr>
<td>GPQIQO</td>
<td>Inquire Input Queue Overflow</td>
<td>Inquire Input Queue Overflow</td>
</tr>
<tr>
<td>GPQIR</td>
<td>Inquire Interior Representation</td>
<td>Inquire Interior Representation</td>
</tr>
<tr>
<td>GPQIVF</td>
<td>Inquire Invisibility Filter</td>
<td>Inquire Invisibility Filter</td>
</tr>
<tr>
<td>GPQLC</td>
<td>Inquire Locator Device State</td>
<td>Inquire Locator Device State 3</td>
</tr>
<tr>
<td>GPQLCF</td>
<td>Inquire List of Color Facilities</td>
<td>Inquire Color Facilities</td>
</tr>
<tr>
<td>GPQLI</td>
<td>Inquire List of Logical Input Devices</td>
<td>Inquire Number of Available Logical Input Devices</td>
</tr>
<tr>
<td>GPQLR</td>
<td>Inquire Polyl ine Representation</td>
<td>Inquire Polyl ine Representation</td>
</tr>
<tr>
<td>GPQLW</td>
<td>Inquire Length of Workstation State Tables</td>
<td>Inquire Workstation State Table Lengths</td>
</tr>
<tr>
<td>GPQMGR</td>
<td>Inquire Polymarker Representation</td>
<td>Inquire Polymarker Representation</td>
</tr>
<tr>
<td>GPQNCN</td>
<td>Inquire Number of Available Class Names</td>
<td>Inquire PHIGS Facilities</td>
</tr>
<tr>
<td>GPQNSP</td>
<td>Inquire Number of Structure Priorities Supported</td>
<td>Inquire Number of Display Priorities Supported</td>
</tr>
<tr>
<td>GPQNV</td>
<td>Inquire Number of Definable View Table Entries</td>
<td>Inquire Workstation State Table Lengths</td>
</tr>
<tr>
<td>GPQOPS</td>
<td>Inquire Open Structure</td>
<td>Inquire Open Structure</td>
</tr>
<tr>
<td>GPQOPW</td>
<td>Inquire Set of Open Workstations</td>
<td>Inquire Set of Open Workstations</td>
</tr>
<tr>
<td>xxxx Subroutine Call</td>
<td>xxxx Subroutine Call Description</td>
<td>Associated ISO PHIGS Subroutine Call(s)</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>GPOPAR</td>
<td>Inquire Pattern Representation</td>
<td>Inquire Pattern Representation</td>
</tr>
<tr>
<td>GPQPAS</td>
<td>Inquire Ancestors of Structure</td>
<td>Inquire Paths to Ancestors</td>
</tr>
<tr>
<td>GPQPCR</td>
<td>Inquire Predefined Color</td>
<td>Inquire Predefined Color Representation</td>
</tr>
<tr>
<td>GPQPDS</td>
<td>Inquire Descendants of Structure</td>
<td>Inquire Paths to Descendants</td>
</tr>
<tr>
<td>GQPQER</td>
<td>Inquire Predefined Edge</td>
<td>Inquire Predefined Edge Representation</td>
</tr>
<tr>
<td>GPQPIR</td>
<td>Inquire Predefined Interior</td>
<td>Inquire Predefined Interior</td>
</tr>
<tr>
<td>GPQPK</td>
<td>Inquire Pick Device State</td>
<td>Inquire Pick Device State 3</td>
</tr>
<tr>
<td>GPPOLF</td>
<td>Inquire Polylines Facilities</td>
<td>Inquire Polylines Facilities</td>
</tr>
<tr>
<td>GPQPLR</td>
<td>Inquire Predefined Polylines</td>
<td>Inquire Predefined Polylines</td>
</tr>
<tr>
<td>GPQPMF</td>
<td>Inquire Polymarker Facilities</td>
<td>Inquire Polymarker Facilities</td>
</tr>
<tr>
<td>GPQPMR</td>
<td>Inquire Predefined Polymarker</td>
<td>Inquire Predefined Polymarker</td>
</tr>
<tr>
<td>GPQPPR</td>
<td>Inquire Predefined Pattern</td>
<td>Inquire Predefined Pattern</td>
</tr>
<tr>
<td>GPQPTR</td>
<td>Inquire Predefined Text</td>
<td>Inquire Predefined Text</td>
</tr>
<tr>
<td>GPQRCT</td>
<td>Inquire Realized Connection and Type</td>
<td>Inquire Workstation Connection and Type</td>
</tr>
<tr>
<td>GPQORST</td>
<td>Inquire Referencing Structure</td>
<td>Inquire Paths to Ancestors</td>
</tr>
<tr>
<td>GPQRV</td>
<td>Inquire Set of Roots in View</td>
<td>Inquire Posted Structures</td>
</tr>
<tr>
<td>GPQRVE</td>
<td>Inquire Requested View Table Entries</td>
<td>Inquire List of View Indices</td>
</tr>
<tr>
<td>GPQRVR</td>
<td>Inquire Requested View</td>
<td>Inquire View Representation , Inquire HLHSR Mode</td>
</tr>
<tr>
<td>GPQSEV</td>
<td>Inquire More Simultaneous Events</td>
<td>Inquire More Simultaneous Events</td>
</tr>
<tr>
<td>GQSK</td>
<td>Inquire Stroke Device State</td>
<td>Inquire Stroke Device State 3</td>
</tr>
<tr>
<td>GQST</td>
<td>Inquire String Device State</td>
<td>Inquire String Device State 3</td>
</tr>
<tr>
<td>GQSTE</td>
<td>Inquire Structure Existence</td>
<td>Inquire Structure Status</td>
</tr>
<tr>
<td>GQSTI</td>
<td>Inquire Structure Identifiers</td>
<td>Inquire Structure Identifiers</td>
</tr>
<tr>
<td>GQSTS</td>
<td>Inquire Structure Status</td>
<td>Inquire Structure Status</td>
</tr>
<tr>
<td>GQSTSV</td>
<td>Inquire Structure State Value</td>
<td>Inquire Structure State Value</td>
</tr>
<tr>
<td>GQSYV</td>
<td>Inquire System State Value</td>
<td>Inquire System State Value</td>
</tr>
<tr>
<td>GPQTR</td>
<td>Inquire Text Representation</td>
<td>Inquire Text Representation</td>
</tr>
<tr>
<td>GPQTXF</td>
<td>Inquire Text Facilities</td>
<td>Inquire Text Facilities</td>
</tr>
<tr>
<td>GQVL</td>
<td>Inquire Valuator Device State</td>
<td>Inquire Valuator Device State 3</td>
</tr>
<tr>
<td>GPOWC</td>
<td>Inquire Workstation Category</td>
<td>Inquire Workstation Category</td>
</tr>
<tr>
<td>GPOWD</td>
<td>Inquire Workstation Display</td>
<td>Inquire Workstation Classification</td>
</tr>
<tr>
<td>GPOWDT</td>
<td>Inquire Workstation Description</td>
<td>Inquire Modeling Clipping Facilities</td>
</tr>
<tr>
<td>GPQWSA</td>
<td>Inquire Set of Workstations to Which Associated</td>
<td>Inquire Set of Workstations to Which Posted</td>
</tr>
<tr>
<td>GPOWST</td>
<td>Inquire List of Available Workstation Types</td>
<td>Inquire List of Available Workstation Types</td>
</tr>
<tr>
<td>GQWSV</td>
<td>Inquire Workstation State Value</td>
<td>Inquire Workstation State Value</td>
</tr>
<tr>
<td>GQWSX</td>
<td>Inquire Workstation Transformation</td>
<td>Inquire Workstation Transformation 3</td>
</tr>
<tr>
<td>GQXAF</td>
<td>Inquire Extended Annotation Font Characteristics</td>
<td>Inquire Annotation Facilities</td>
</tr>
<tr>
<td>GQXCR</td>
<td>Inquire Extended Color Representation</td>
<td>Inquire Color Representation</td>
</tr>
<tr>
<td>Subroutine Call</td>
<td>Subroutine Call Description</td>
<td>Associated ISO PHIGS Subroutine Call(s)</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>GPOQXER</td>
<td>Inquire Extended Edge</td>
<td>Inquire Edge Representation</td>
</tr>
<tr>
<td>GPOQXIR</td>
<td>Inquire Extended Interior</td>
<td>Inquire Interior Representation</td>
</tr>
<tr>
<td>GPOQXLR</td>
<td>Inquire Extended Polyline</td>
<td>Inquire Polyline Representation</td>
</tr>
<tr>
<td>GPOQXMR</td>
<td>Inquire Extended Polymarker</td>
<td>Inquire Polymarker Representation</td>
</tr>
<tr>
<td>GPOQXTR</td>
<td>Inquire Extended Text</td>
<td>Inquire Text Representation</td>
</tr>
<tr>
<td>GPRAS</td>
<td>Retrieve Ancestors to Structures</td>
<td>Retrieve Ancestors to Structures</td>
</tr>
<tr>
<td>GPRAST</td>
<td>Redraw All Structures</td>
<td>Redraw All Structures</td>
</tr>
<tr>
<td>GPRCN</td>
<td>Remove Class Name from Set</td>
<td>Remove Names from Set</td>
</tr>
<tr>
<td>GPRDS</td>
<td>Retrieve Descendants to Structures</td>
<td>Retrieve Descendants to Structures</td>
</tr>
<tr>
<td>GPROTX</td>
<td>Rotate X</td>
<td>Rotate X</td>
</tr>
<tr>
<td>GPROTY</td>
<td>Rotate Y</td>
<td>Rotate Y</td>
</tr>
<tr>
<td>GPROTZ</td>
<td>Rotate Z</td>
<td>Rotate Z</td>
</tr>
<tr>
<td>GPROT2</td>
<td>Rotate 2</td>
<td>Rotate</td>
</tr>
<tr>
<td>GPRQCH</td>
<td>Request Choice</td>
<td>Request Choice</td>
</tr>
<tr>
<td>GPRQLC</td>
<td>Request Locator</td>
<td>Request Locator 3</td>
</tr>
<tr>
<td>GPRQPK</td>
<td>Request Pick</td>
<td>Request Pick</td>
</tr>
<tr>
<td>GPRQSK</td>
<td>Request Stroke</td>
<td>Request Stroke 3</td>
</tr>
<tr>
<td>GPRQST</td>
<td>Request String</td>
<td>Request String</td>
</tr>
<tr>
<td>GPRQVL</td>
<td>Request Valuator</td>
<td>Request Valuator</td>
</tr>
<tr>
<td>GPRMCV</td>
<td>Restore Modeling Clipping Volume</td>
<td>Restore Modeling Clipping Volume</td>
</tr>
<tr>
<td>GPRSTI</td>
<td>Retrieve Structure Identifiers</td>
<td>Retrieve Structure Identifiers</td>
</tr>
<tr>
<td>GPRVSN</td>
<td>Retrieve Structure Networks</td>
<td>Retrieve Structure Networks</td>
</tr>
<tr>
<td>GPRVST</td>
<td>Retrieve Structures</td>
<td>Retrieve Structures</td>
</tr>
<tr>
<td>GPS2C</td>
<td>Scale 2</td>
<td>Scale</td>
</tr>
<tr>
<td>GPS3C</td>
<td>Scale 3</td>
<td>Scale</td>
</tr>
<tr>
<td>GPSKMO</td>
<td>Set Stroke Mode</td>
<td>Set Stroke Mode</td>
</tr>
<tr>
<td>GPSMCH</td>
<td>Sample Choice</td>
<td>Sample Choice</td>
</tr>
<tr>
<td>GPSMLC</td>
<td>Sample Locator</td>
<td>Sample Locator 3</td>
</tr>
<tr>
<td>GPSMPK</td>
<td>Sample Pick</td>
<td>Sample Pick</td>
</tr>
<tr>
<td>GPSMSK</td>
<td>Sample Stroke</td>
<td>Sample Stroke 3</td>
</tr>
<tr>
<td>GPSMST</td>
<td>Sample String</td>
<td>Sample String</td>
</tr>
<tr>
<td>GPSMLV</td>
<td>Sample Valuator</td>
<td>Sample Valuator</td>
</tr>
<tr>
<td>GPSTM0</td>
<td>Set String Mode</td>
<td>Set String Mode</td>
</tr>
<tr>
<td>GPTRL2</td>
<td>Translate 2</td>
<td>Translate</td>
</tr>
<tr>
<td>GPTRL3</td>
<td>Translate 3</td>
<td>Translate 3</td>
</tr>
<tr>
<td>GPTXAL</td>
<td>Set Text Alignment</td>
<td>Set Text Alignment</td>
</tr>
<tr>
<td>GPTXCI</td>
<td>Set Text Color Index</td>
<td>Set Text Color Index</td>
</tr>
<tr>
<td>GPTXFO</td>
<td>Set Text Font</td>
<td>Set Text Font</td>
</tr>
<tr>
<td>GPTXI</td>
<td>Set Text Index</td>
<td>Set Text Index</td>
</tr>
<tr>
<td>GPTXPFR</td>
<td>Set Text Precision</td>
<td>Set Text Precision</td>
</tr>
<tr>
<td>GPTXPT</td>
<td>Set Text Path</td>
<td>Set Text Path</td>
</tr>
<tr>
<td>GPTXR</td>
<td>Set Text Representation</td>
<td>Set Text Representation</td>
</tr>
<tr>
<td>GPTX2</td>
<td>Text 2</td>
<td>Text</td>
</tr>
<tr>
<td>GPTX3</td>
<td>Text 3</td>
<td>Text 3</td>
</tr>
<tr>
<td>GPUPWS</td>
<td>Update Workstation</td>
<td>Update Workstation</td>
</tr>
<tr>
<td>GPVCH</td>
<td>Set View Characteristics</td>
<td>Set View Representation, Set View</td>
</tr>
<tr>
<td>GPVIP</td>
<td>Set View Input Priority</td>
<td>Set View Transformation Input Priority</td>
</tr>
<tr>
<td>GPVLMO</td>
<td>Set Valuator Mode</td>
<td>Set Valuator Mode</td>
</tr>
</tbody>
</table>
Table 3. GPxxxx Subroutine Calls and Their Associated ISO PHIGS Subroutines (continued)

<table>
<thead>
<tr>
<th>xxxx Subroutine Call</th>
<th>xxxx Subroutine Call Description</th>
<th>Associated ISO PHIGS Subroutine Call(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPVMP2</td>
<td>Set View Mapping 2</td>
<td>Set View Representation, Evaluate View Mapping Matrix</td>
</tr>
<tr>
<td>GPVMP3</td>
<td>Set View Mapping 3</td>
<td>Set View Representation 3, Evaluate View Mapping Matrix 3</td>
</tr>
<tr>
<td>GPXMT2</td>
<td>Set View Matrix 2</td>
<td>Set View Representation, Evaluate View Orientation Matrix</td>
</tr>
<tr>
<td>GPXMT3</td>
<td>Set View Matrix 3</td>
<td>Set View Representation 3, Evaluate View Orientation Matrix 3</td>
</tr>
<tr>
<td>GPVP</td>
<td>Set View Priority</td>
<td>Set View Transformation Input Priority</td>
</tr>
<tr>
<td>GPVPLN</td>
<td>Set View Plane Normal</td>
<td>Evaluate View Orientation Matrix 3</td>
</tr>
<tr>
<td>GPVR</td>
<td>Set View Reference Point</td>
<td>Evaluate View Orientation Matrix 3</td>
</tr>
<tr>
<td>GPVUP</td>
<td>Set View Up</td>
<td>Evaluate View Orientation Matrix 3</td>
</tr>
<tr>
<td>GPVWI</td>
<td>Set View Index</td>
<td>Set View Index</td>
</tr>
<tr>
<td>GPWSX2</td>
<td>Set Workstation Transformation 2</td>
<td>Set Workstation Viewport</td>
</tr>
<tr>
<td>GPWSX3</td>
<td>Set Workstation Transformation 3</td>
<td>Set Workstation Viewport 3</td>
</tr>
<tr>
<td>GPXCR</td>
<td>Set Extended Color Representation</td>
<td>Set Color Representation</td>
</tr>
<tr>
<td>GPXER</td>
<td>Set Extended Edge Representation</td>
<td>Set Edge Representation</td>
</tr>
<tr>
<td>GPXF2</td>
<td>Transform Point 2</td>
<td>Transform Point</td>
</tr>
<tr>
<td>GPXF3</td>
<td>Transform Point 3</td>
<td>Transform Point 3</td>
</tr>
<tr>
<td>GPXIR</td>
<td>Set Extended Interior Representation</td>
<td>Set Interior Representation</td>
</tr>
<tr>
<td>GPXPLR</td>
<td>Set Extended Polyline Representation</td>
<td>Set Polyline Representation</td>
</tr>
<tr>
<td>GPXPMR</td>
<td>Set Extended Polymarker Representation</td>
<td>Set Polymarker Representation</td>
</tr>
<tr>
<td>GPXTXR</td>
<td>Set Extended Text Representation</td>
<td>Set Text Representation</td>
</tr>
<tr>
<td>GPXVCH</td>
<td>Set Extended View Characteristics</td>
<td>Set View Representation, Set View Representation 3</td>
</tr>
<tr>
<td>GPXVR</td>
<td>Set Extended View Representation</td>
<td>Set View Representation, Set View Representation 3, Set HLHSR Mode</td>
</tr>
</tbody>
</table>
Chapter 22. Implementation Errors and graPHIGS API Messages for ISO PHIGS-Defined Errors

ISO PHIGS error numbers are divided into the following ranges:

**Errors <0**  
Implementation dependent

**Errors 1-1999**  
ISO PHIGS standard errors

**Errors 2000-3999**  
Language binding errors

The absolute value of a negative error number refers to a GPxxxx message number listed in *The graPHIGS Programming Interface: Messages and Codes*.

The following tables list the graPHIGS API messages for ISO PHIGS standard errors, ISO PHIGS C binding errors, and ISO PHIGS FORTRAN binding errors. The graPHIGS API ignores functions which cause errors, unless the error message specifically says that it is a WARNING. See the ISO PHIGS standard and ISO PHIGS FORTRAN and C bindings for more information on the errors.

**Table 4. graPHIGS API Messages for ISO PHIGS Standard Errors**

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FUNCTION REQUIRES STATE (PHCL,WSCL,STCL,ARCL)</td>
</tr>
<tr>
<td>2</td>
<td>FUNCTION REQUIRES STATE (PHOP,<em>,</em>,*)</td>
</tr>
<tr>
<td>3</td>
<td>FUNCTION REQUIRES STATE (PHOP,WSOP,<em>,</em>)</td>
</tr>
<tr>
<td>4</td>
<td>FUNCTION REQUIRES STATE (PHOP,WSCL,STCL,ARCL)</td>
</tr>
<tr>
<td>5</td>
<td>FUNCTION REQUIRES STATE (PHOP,<em>,STOP,</em>)</td>
</tr>
<tr>
<td>6</td>
<td>FUNCTION REQUIRES STATE (PHOP,<em>,STCL,</em>)</td>
</tr>
<tr>
<td>7</td>
<td>FUNCTION REQUIRES STATE (PHOP,<em>,</em>,AROP)</td>
</tr>
<tr>
<td>50</td>
<td>CONNECTION IDENTIFIER NOT RECOGNIZED BY IMPLEMENTATION</td>
</tr>
<tr>
<td>51</td>
<td>INFORMATION NOT AVAILABLE FOR GENERIC WORKSTATION TYPE</td>
</tr>
<tr>
<td>52</td>
<td>WORKSTATION TYPE NOT RECOGNIZED BY IMPLEMENTATION</td>
</tr>
<tr>
<td>53</td>
<td>WORKSTATION IDENTIFIER ALREADY IN USE</td>
</tr>
<tr>
<td>54</td>
<td>SPECIFIED WORKSTATION IS NOT OPEN</td>
</tr>
<tr>
<td>55</td>
<td>WORKSTATION NOT OPENED FOR IMPLEMENTATION DEPENDENT REASON</td>
</tr>
<tr>
<td>56</td>
<td>SPECIFIED WORKSTATION IS NOT OF CATEGORY MO</td>
</tr>
<tr>
<td>57</td>
<td>SPECIFIED WORKSTATION IS OF CATEGORY MI</td>
</tr>
<tr>
<td>58</td>
<td>SPECIFIED WORKSTATION IS NOT OF CATEGORY MI</td>
</tr>
<tr>
<td>59</td>
<td>SPECIFIED WORKSTATION DOES NOT HAVE OUTPUT CAPABILITY</td>
</tr>
<tr>
<td>60</td>
<td>SPECIFIED WORKSTATION IS NOT OF CATEGORY OUTIN</td>
</tr>
<tr>
<td>61</td>
<td>SPECIFIED WORKSTATION IS NOT OF CATEGORY INPUT OR OUTIN</td>
</tr>
<tr>
<td>62</td>
<td>THIS INFORMATION NOT AVAILABLE FOR MO WORKSTATION TYPE</td>
</tr>
<tr>
<td>63</td>
<td>EXCEEDED MAXIMUM NUMBER OF SIMULTANEOUSLY OPEN WORKSTATIONS</td>
</tr>
<tr>
<td>64</td>
<td>SPECIFIED WORKSTATION TYPE CANNOT GENERATE SPECIFIED GDP</td>
</tr>
<tr>
<td>100</td>
<td>BUNDLE INDEX VALUE IS LESS THAN ONE</td>
</tr>
<tr>
<td>101</td>
<td>SPECIFIED REPRESENTATION HAS NOT BEEN DEFINED</td>
</tr>
<tr>
<td>102</td>
<td>REPRESENTATION HAS NOT BEEN PREDEFINED ON THIS WORKSTATION</td>
</tr>
<tr>
<td>103</td>
<td>EXCEEDED MAXIMUM NUMBER OF WORKSTATION BUNDLE TABLE ENTRIES</td>
</tr>
<tr>
<td>104</td>
<td>SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>105</td>
<td>SPECIFIED MARKER TYPE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>106</td>
<td>SPECIFIED FONT NOT AVAILABLE FOR REQUESTED TEXT PRECISION</td>
</tr>
<tr>
<td>107</td>
<td>SPECIFIED EDGETYPE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>108</td>
<td>SPECIFIED INTERIOR STYLE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>109</td>
<td>INTERIOR STYLE PATTERN NOT SUPPORTED ON WORKSTATION</td>
</tr>
<tr>
<td>Error Number</td>
<td>Message</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
</tr>
<tr>
<td>110</td>
<td>SPECIFIED COLOUR MODEL NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>111</td>
<td>SPECIFIED HLHSR MODE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>112</td>
<td>PATTERN INDEX VALUE &lt; ONE</td>
</tr>
<tr>
<td>113</td>
<td>COLOUR INDEX VALUE &lt; ZERO</td>
</tr>
<tr>
<td>114</td>
<td>VIEW INDEX VALUE &lt; ZERO</td>
</tr>
<tr>
<td>115</td>
<td>VIEW INDEX VALUE &lt; ONE</td>
</tr>
<tr>
<td>116</td>
<td>ONE DIMENSION OF PATTERN COLOUR INDEX ARRAY &lt; ONE</td>
</tr>
<tr>
<td>117</td>
<td>ONE DIMENSION OF COLOUR INDEX ARRAY &lt; ZERO</td>
</tr>
<tr>
<td>118</td>
<td>COLOUR COMPONENT IS OUT OF RANGE</td>
</tr>
<tr>
<td>150</td>
<td>EXCEEDED MAXIMUM NUMBER OF VIEW TABLE ENTRIES</td>
</tr>
<tr>
<td>151</td>
<td>INVALID WINDOW: MINIMUM VALUE =&gt; TO CORRESPONDING MAXIMUM VALUE</td>
</tr>
<tr>
<td>152</td>
<td>INVALID VIEWPORT: XMIN &gt;= XMAX, YMIN &gt;= YMAX OR ZMIN &gt; ZMAX</td>
</tr>
<tr>
<td>153</td>
<td>INVALID VIEW CLIPPING LIMITS: XMIN&gt;=XMAX, YMIN&gt;=YMAX or ZMIN&gt;ZMAX</td>
</tr>
<tr>
<td>154</td>
<td>VIEW CLIPPING LIMITS ARE NOT WITHIN NPC RANGE</td>
</tr>
<tr>
<td>155</td>
<td>PROJECTION VIEWPORT LIMITS ARE NOT WITHIN NPC RANGE</td>
</tr>
<tr>
<td>156</td>
<td>WORKSTATION WINDOW LIMITS ARE NOT WITHIN NPC RANGE</td>
</tr>
<tr>
<td>157</td>
<td>WORKSTATION VIEWPORT IS NOT WITHIN DISPLAY SPACE</td>
</tr>
<tr>
<td>158</td>
<td>FRONT PLANE DISTANCE = BACK PLANE DISTANCE WHEN Z EXTENT NON ZERO</td>
</tr>
<tr>
<td>159</td>
<td>VIEW PLANE NORMAL VECTOR HAS LENGTH ZERO</td>
</tr>
<tr>
<td>160</td>
<td>VIEW UP VECTOR HAS LENGTH ZERO</td>
</tr>
<tr>
<td>161</td>
<td>VIEW UP AND VIEW PLANE NORMAL VECTORS ARE PARALLEL</td>
</tr>
<tr>
<td>162</td>
<td>PROJECTION REFERENCE POINT BETWEEN FRONT AND BACK PLANES</td>
</tr>
<tr>
<td>163</td>
<td>PROJECTION REFERENCE POINT CANNOT BE POSITIONED ON VIEW PLANE</td>
</tr>
<tr>
<td>164</td>
<td>BACK PLANE IS IN FRONT OF THE FRONT PLANE</td>
</tr>
<tr>
<td>200</td>
<td>WARNING, IGNORING STRUCTURES THAT DO NOT EXIST</td>
</tr>
<tr>
<td>201</td>
<td>SPECIFIED STRUCTURE DOES NOT EXIST</td>
</tr>
<tr>
<td>202</td>
<td>SPECIFIED ELEMENT DOES NOT EXIST</td>
</tr>
<tr>
<td>203</td>
<td>SPECIFIED STARTING PATH NOT FOUND IN CSS</td>
</tr>
<tr>
<td>204</td>
<td>SPECIFIED SEARCH CEILING INDEX OUT OF RANGE</td>
</tr>
<tr>
<td>205</td>
<td>LABEL NOT BETWEEN ELEMENT POINTER AND END OF STRUCTURE</td>
</tr>
<tr>
<td>206</td>
<td>LABEL(S) NOT BETWEEN ELEMENT POINTER AND END OF STRUCTURE</td>
</tr>
<tr>
<td>207</td>
<td>SPECIFIED PATH DEPTH &lt; ZERO</td>
</tr>
<tr>
<td>208</td>
<td>DISPLAY PRIORITY IS OUT OF RANGE</td>
</tr>
<tr>
<td>250</td>
<td>SPECIFIED DEVICE NOT AVAILABLE ON WORKSTATION</td>
</tr>
<tr>
<td>251</td>
<td>FUNCTION REQUIRES INPUT DEVICE TO BE IN REQUEST MODE</td>
</tr>
<tr>
<td>252</td>
<td>FUNCTION REQUIRES INPUT DEVICE TO BE IN SAMPLE MODE</td>
</tr>
<tr>
<td>253</td>
<td>PROMPT/ECHO TYPE NOT AVAILABLE ON SPECIFIED WORKSTATION</td>
</tr>
<tr>
<td>254</td>
<td>INVALID ECHO AREA/VOLUME: XMIN&gt;=XMAX, YMIN&gt;=YMAX OR ZMIN&gt;ZMAX</td>
</tr>
<tr>
<td>255</td>
<td>ECHO AREA/VOLUME BOUNDARY POINT(S) OUTSIDE DEVICE RANGE</td>
</tr>
<tr>
<td>256</td>
<td>WARNING, INPUT QUEUE HAS OVERFLOWED</td>
</tr>
<tr>
<td>257</td>
<td>INPUT QUEUE HAS NOT OVERFLOWED</td>
</tr>
<tr>
<td>258</td>
<td>INPUT QUEUE HAS OVERFLOWED, BUT WORKSTATION IS CLOSED</td>
</tr>
<tr>
<td>259</td>
<td>REQUESTED DEVICE CLASS NOT CURRENT INPUT REPORT CLASS</td>
</tr>
<tr>
<td>260</td>
<td>INPUT DEVICE DATA RECORD FIELD IS IN ERROR</td>
</tr>
<tr>
<td>261</td>
<td>INITIAL VALUE IS INVALID</td>
</tr>
<tr>
<td>262</td>
<td>NUMBER OF POINTS IN INITIAL STROKE &gt; BUFFER SIZE</td>
</tr>
<tr>
<td>263</td>
<td>LENGTH OF INITIAL STRING &gt; BUFFER SIZE</td>
</tr>
<tr>
<td>300</td>
<td>ITEM TYPE IS NOT ALLOWED FOR USER ITEMS</td>
</tr>
<tr>
<td>301</td>
<td>ITEM LENGTH IS INVALID</td>
</tr>
<tr>
<td>302</td>
<td>NO ITEM IS LEFT IN METAFILE INPUT</td>
</tr>
<tr>
<td>303</td>
<td>METAFILE ITEM IS INVALID</td>
</tr>
<tr>
<td>304</td>
<td>ITEM TYPE IS UNKNOWN</td>
</tr>
<tr>
<td>305</td>
<td>CONTENT OF ITEM DATA RECORD INVALID FOR SPECIFIED ITEM TYPE</td>
</tr>
<tr>
<td>306</td>
<td>MAXIMUM ITEM DATA RECORD LENGTH IS INVALID</td>
</tr>
</tbody>
</table>
Table 4. graPHIGS API Messages for ISO PHIGS Standard Errors (continued)

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>307</td>
<td>USER ITEM CANNOT BE INTERPRETED</td>
</tr>
<tr>
<td>350</td>
<td>WARNING, SPECIFIED ESCAPE UNAVAILABLE ON ONE OR MORE WORKSTATIONS</td>
</tr>
<tr>
<td>351</td>
<td>ONE OF THE FIELDS WITHIN THE ESCAPE DATA RECORD IS IN ERROR</td>
</tr>
<tr>
<td>400</td>
<td>ARCHIVE FILE CANNOT BE OPENED</td>
</tr>
<tr>
<td>401</td>
<td>EXCEEDED MAXIMUM NUMBER OF SIMULTANEOUSLY OPEN ARCHIVE FILES</td>
</tr>
<tr>
<td>402</td>
<td>ARCHIVE FILE IDENTIFIER ALREADY IN USE</td>
</tr>
<tr>
<td>403</td>
<td>ARCHIVE FILE IS NOT A PHIGS ARCHIVE FILE</td>
</tr>
<tr>
<td>404</td>
<td>SPECIFIED ARCHIVE FILE IS NOT OPEN</td>
</tr>
<tr>
<td>405</td>
<td>NAME CONFLICT OCCURRED, CONFLICT RESOLUTION FLAG = ABANDON</td>
</tr>
<tr>
<td>406</td>
<td>WARNING, ARCHIVE FILE IS FULL</td>
</tr>
<tr>
<td>407</td>
<td>WARNING, SOME SPECIFIED STRUCTURES DO NOT EXIST ON ARCHIVE FILE</td>
</tr>
<tr>
<td>408</td>
<td>WARNING, STRUCTURE(S) NOT IN ARCHIVE, EMPTY ONE(S) TO BE CREATED</td>
</tr>
<tr>
<td>450</td>
<td>SPECIFIED ERROR FILE IS INVALID</td>
</tr>
<tr>
<td>900</td>
<td>STORAGE OVERFLOW HAS OCCURRED IN PHIGS</td>
</tr>
<tr>
<td>901</td>
<td>STORAGE OVERFLOW HAS OCCURRED IN CSS</td>
</tr>
<tr>
<td>902</td>
<td>INPUT/OUTPUT ERROR OCCURRED WHILE READING</td>
</tr>
<tr>
<td>903</td>
<td>INPUT/OUTPUT ERROR OCCURRED WHILE WRITING</td>
</tr>
<tr>
<td>904</td>
<td>INPUT/OUTPUT ERROR OCCURRED WHILE SENDING DATA TO A WORKSTATION</td>
</tr>
<tr>
<td>905</td>
<td>INPUT/OUTPUT ERROR OCCURRED WHILE RECEIVING DATA FROM A WORKSTATION</td>
</tr>
<tr>
<td>906</td>
<td>INPUT/OUTPUT ERROR OCCURRED DURING PROGRAM LIBRARY MANAGEMENT</td>
</tr>
<tr>
<td>907</td>
<td>INPUT/OUTPUT ERROR OCCURRED WHILE READING THE WDT</td>
</tr>
<tr>
<td>908</td>
<td>ARITHMETIC ERROR HAS OCCURRED</td>
</tr>
</tbody>
</table>

The following table lists the graPHIGS API messages for ISO PHIGS FORTRAN binding errors. See the ISO/IEC FORTRAN binding for more information on these errors.

Table 5. graPHIGS API Messages for ISO PHIGS FORTRAN Binding Errors

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>ENUMERATION TYPE OUT OF RANGE</td>
</tr>
<tr>
<td>2001</td>
<td>OUTPUT PARAMETER SIZE INSUFFICIENT</td>
</tr>
<tr>
<td>2002†</td>
<td>LIST OR SET ELEMENT NOT AVAILABLE</td>
</tr>
<tr>
<td>2003</td>
<td>INVALID DATA RECORD</td>
</tr>
<tr>
<td>2004</td>
<td>INPUT PARAMETER SIZE OUT OF RANGE</td>
</tr>
<tr>
<td>2005</td>
<td>INVALID LIST OF POINT LISTS</td>
</tr>
<tr>
<td>2006</td>
<td>INVALID LIST OF FILTERS</td>
</tr>
</tbody>
</table>

Note: †If this error occurs, the total number in the list is still returned, but the requested element is not returned.

The following table lists the graPHIGS API messages for ISO PHIGS C binding errors.

Table 6. graPHIGS API Messages for ISO PHIGS C Binding Errors

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>2200</td>
<td>START INDEX IS OUT OF RANGE</td>
</tr>
<tr>
<td>2201</td>
<td>LENGTH OF THE APPLICATION’S LIST IS NEGATIVE</td>
</tr>
<tr>
<td>2202</td>
<td>ENUMERATION TYPE OUT OF RANGE</td>
</tr>
<tr>
<td>2203</td>
<td>ERROR WHILE ALLOCATING STORE</td>
</tr>
<tr>
<td>2204</td>
<td>ERROR WHILE ALLOCATING MEMORY FOR STORE</td>
</tr>
</tbody>
</table>
You can write applications that combine GPxxx subroutine calls and ISO PHIGS subroutine calls. All of the GPxxx subroutines (documented in The graPHIGS Programming Interface: Subroutine Reference) are available as extensions to the ISO PHIGS library of subroutine calls. This chapter identifies some graPHIGS API extensions and discusses considerations for combining GPxxx subroutine calls and ISO PHIGS subroutine calls.

Helpful Hints
Enumerations in ISO PHIGS subroutine calls are not necessarily the same as in GPxxx subroutine calls.

Matrixes in ISO PHIGS apply to a column vector, while GPxxx matrixes apply to a row vector. In addition, the order in which the matrix is stored is determined by the binding. See Chapter 15, "ISO PHIGS Transformations" for a description of the storage of a transformation matrix.

graPHIGS API Extensions to the ISO PHIGS Standard

Many GPxxx subroutine calls offer features beyond the ISO PHIGS standard. These include:

ISO PHIGS PLUS functions
• Advanced primitives (NURBS curves/surfaces, triangle strip, . . .)
• Lighting and shading
• Hidden-line/hidden-surface removal (HLHSR)
• Depth cueing
• Direct color.

Other features
• More primitives (spheres, grids, composite fill, polyhedron edge, . . .)
• Proportional/filled fonts
• Conditional traversal
• Image display
• Integration with X-Windows
• Distributed architecture for networked processing
• Sharing of graphical resources between processes
• Enhanced structure editing
• View-based traversal.

Compatibility between graPHIGS API Extensions and the ISO PHIGS Standard

When writing an application that uses both GPxxx subroutine calls and ISO PHIGS subroutine calls, there are several considerations you should be aware of:

The architecture
To provide for distributed processing, the graPHIGS API separates its code into the graPHIGS API shell and the graPHIGS API nucleus. For more information about the graPHIGS API shell and the graPHIGS API nucleus, see The graPHIGS Programming Interface: Understanding Concepts. Many of the considerations which are listed below are directly related to the implementation of this distributed architecture.
View specification

Two approaches are available in the area of view specification. ISO PHIGS and GPxxxx subroutines provide an approach whereby structures are posted to workstations and then traversed for display. Display traversal begins with a default view index which may change during traversal due to Set View Index structure elements. For a discussion of Set View Index processing within the graPHIGS API environment, see Appendix C of The graPHIGS Programming Interface: Understanding Concepts.

In addition, through GPxxxx subroutines, the view specification may be handled in another way. Root structures may be posted to views within workstations by using the Associate Root to View (GPARV) subroutine function. Views are then traversed in output priority order. View-based traversal allows the graPHIGS API to optimize updates by redrawing only views with changed data.

The following sections describe:

• Considerations for combining the two types of subroutines, listed by type of function.
• Rules used by the graPHIGS API when reporting errors to an application that combines subroutine calls.

See Chapter 19, "ISO PHIGS Subroutines to GPxxxx Subroutines" and Chapter 20, "GPxxxx Subroutines to ISO PHIGS Subroutines" for listings of ISO PHIGS subroutines and their GPxxxx equivalents or approximate equivalents, and vice versa.

Control Subroutines

Issuing Open Workstation

Opening a workstation with the ISO PHIGS Open Workstation subroutine results in the following differences in default values used at the workstation as compared to the defaults assumed when the workstation is opened using either GPOPWS or GPCRWS.

Table 12. Default values when opening (creating) the workstation.

<table>
<thead>
<tr>
<th></th>
<th>ISO PHIGS Open Workstation</th>
<th>graPHIGS API GPOPWS or GPCRWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of line types available</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Number of edge types available</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Default Mapping Matrixes</td>
<td>Maps WC cube [0, 1] x [0, 1] x [0, 1] to the cube in NPC space of [0, 1] x [0, 1] x [0, 1].</td>
<td>Maps WC cube [-1, 1] x [-1, 1] x [-1, 1] to the cube in NPC space of [0, 1] x [0, 1] x [0, 1].</td>
</tr>
<tr>
<td>Workstation viewport</td>
<td>All of DC space.¹</td>
<td>Largest square in DC space.¹</td>
</tr>
<tr>
<td>View input</td>
<td>Active for each view.</td>
<td>Active only for view 0.</td>
</tr>
<tr>
<td>Annotation text character height default</td>
<td>0.01</td>
<td>The nominal annotation text character height for the open workstation.</td>
</tr>
<tr>
<td>Hatch table defaults</td>
<td>Hatch table indexes 1-6 are as described by the Set Interior Style Index (GPISI) subroutine (SET INTERIOR STYLE INDEX (PHOP, STOP)); and the remaining indexes of the hatch table are defaulted to a hatch index value of 1.</td>
<td>See &quot;Interior Facilities&quot; in The graPHIGS Programming Interface: Technical Reference.</td>
</tr>
</tbody>
</table>

¹Although visual presentation on the screen is identical, the initial values in the workstation state list are different.

Issuing Open graPHIGS
If you issue the Open graPHIGS (GPOPPH) subroutine with suppression of the nucleus and structure store creation via defaults, you must create a structure store with an id value of 1 before issuing any ISO PHIGS subroutine call that attempts to access the structure store. Otherwise the graPHIGS API generates implementation errors (-11 or -12).

Since connecting to a nucleus is not an ISO PHIGS concept, when you issue an ISO PHIGS Open PHIGS subroutine call, the graPHIGS API connects to a nucleus with an id value of 1 by default. If a program issues the Open graPHIGS (GPOPPH) subroutine with suppression of the nucleus and structure store creation via defaults, and then fails to create a nucleus with an id value of 1, the graPHIGS API generates implementation error (-202) on every ISO PHIGS subroutine call that assumes a nucleus with an id value of 1. Inquire PHIGS Facilities is an ISO PHIGS subroutine call that could generate such an error.

Asynchronous Errors
Some errors from the nucleus may not be reported during the subroutine call which forced the error. The following rules are followed when reporting such an error to an application that combines GPxxxx subroutine calls and ISO PHIGS subroutine calls. (For details on the timing of error reporting, see Appendix B of The graPHIGS Programming Interface: Understanding Concepts).

- If the application has opened PHIGS with ISO PHIGS Open PHIGS, the error is returned as an ISO PHIGS error, and the subroutine name is an ISO PHIGS subroutine call.
- If the application has opened PHIGS with GPOPPH, the error is returned as a GPxxxx error, and the subroutine name is a GPxxxx subroutine call.
- These error logs have an asterisk after the subroutine name to alert the user that this is an error being returned asynchronously from the graPHIGS API nucleus, and as such the error reported and the subroutine name follow the above rules.

The following examples illustrate these rules:

Delete Elements Between Labels and Set Element Pointer at Label could cause asynchronous GPxxxx error 130 (LABEL IDENTIFIER CANNOT BE FOUND IN THE OPEN STRUCTURE). ISO PHIGS has two distinct errors (205 and 206) defined instead. The graPHIGS API issues error 130 if you used GPOPPH, otherwise the graPHIGS API issues the mapped ISO PHIGS error.

Initialize input device subroutine calls could cause asynchronous GPxxxx error 141 (INPUT DEVICE NOT IN CORRECT MODE). ISO PHIGS defines error 251. If you used GPOPPH, then the graPHIGS API issues GPxxxx error 141, otherwise the graPHIGS API issues ISO PHIGS error 251.

Workstation Settings
The graPHIGS API supports workstations that have hatch, polymarker, and linetype tables that you can set. The following GPxxxx subroutine calls can change default entries in these tables:

GPLTR  
(Set Linetype Representation)

GPMTR  
(Set Marker Type Representation)

GPHR  
(Set Hatch Representation)

If an application uses the GPxxxx interface to alter these tables, then the representation that ISO PHIGS defines for a line type, marker type, or hatch value has been altered.

Note: For workstations opened by an ISO PHIGS Open Workstation subroutine call, the default representations for the line type, marker type, and hatch tables are as defined by the ISO PHIGS standard.

HLHSR
ISO PHIGS has only one HLHSR mode for a workstation, whereas the GPxxxx interface has an HLHSR mode per view. If your application uses an ISO PHIGS subroutine call to set the HLHSR mode, the graPHIGS API automatically assigns this mode to view 0. The ISO PHIGS inquiry for the HLHSR mode returns the mode for view 0. The only way to set or inquire the mode of a view other than view 0 is through GPxxxx subroutine calls.

**Issuing Inquires**

You must be careful when your application combines GPxxxx subroutines with ISO PHIGS inquiries. Any element placed into a structure by an ISO PHIGS subroutine may always be inquired through the GPxxxx interface as well as through ISO PHIGS. However, using ISO PHIGS inquiries to obtain data that only could have been placed in a structure or into a workstation table via a GPxxxx subroutine call, generates implementation error -606.

Implementation error -606 implies that the graPHIGS API is ignoring this function, and you should use the appropriate GPOxxx inquiry to obtain the data.

Examples of error -606 on workstation inquiries:

- **Inquire Polyline Representation**: GPxxxx subroutine calls can set polyline representations to values unknown to ISO PHIGS. Using ISO PHIGS Inquire Polyline Representation to inquire such a polyline representation generates error -606.

- **Inquire Polymarker Representation**: GPxxxx subroutine calls can set polymarker representations to values unknown to ISO PHIGS. Using ISO PHIGS Inquire Polymarker Representation to inquire such a polymarker representation generates error -606.

- **Inquire Text Representation**: GPxxxx subroutine calls can set text representations to values unknown to ISO PHIGS. Using ISO PHIGS Inquire Text Representation to inquire such a text representation generates error -606.

- **Inquire Interior Representation**: GPxxxx subroutine calls can set interior representations to values unknown to ISO PHIGS. Using ISO PHIGS Inquire Interior Representation to inquire such an interior representation generates error -606.

- **Inquire Edge Representation**: GPxxxx subroutine calls can set edge representations to values unknown to ISO PHIGS. Using ISO PHIGS Inquire Edge Representation to inquire such an edge representation produces error -606.

- **Inquire Color Model set to CMY**: GPxxxx subroutine calls support the CMY color model, which is unknown to ISO PHIGS. Using ISO PHIGS Inquire Color Model to inquire the color model which has been set to CMY generates error -606.

- **Inquire state of locator device**: You may issue the Initialize Locator (GPINLC) subroutine to initialize a locator device with prompt/echo types or data record variations which are not defined in ISO PHIGS. Using ISO PHIGS subroutines to inquire the state of such a locator device generates error -606.

- **Inquire state of stroke device**: You may issue the Initialize Stroke (GPINSK) subroutine to initialize a stroke device with prompt/echo types or data record variations which are not defined in ISO PHIGS. Using ISO PHIGS subroutines to inquire the state of such a stroke device generates error -606.

- **Inquire state of valuator device**: You may issue the Initialize Valuator (GPINVL) subroutine to initialize a valuator device with prompt/echo types or data record variations which are not defined in ISO PHIGS. Using ISO PHIGS subroutines to inquire the state of such a valuator device generates error -606.

- **Inquire state of string device**: You may issue the Initialize String (GPINST) subroutine to initialize a string device with prompt/echo types or data record variations which are not defined in ISO PHIGS. Using ISO PHIGS subroutines to inquire the state of such a string device generates error -606.

Examples of error -606 on structure store inquiries:

- **Inquire element type and size or element content generated by Attribute Source Flag Setting**: You may issue the Attribute Source Flag Setting (GPASF) subroutine to generate a structure element containing 0, 1, 2, or more id/flag pairs. ISO PHIGS Set Individual ASF requires exactly one id/flag pair.
Thus, if the structure element placed into the structure via GPASF contains zero or more than one
id/flag pair, inquiring the type and size, or the contents of this element through ISO PHIGS generates
error -606.

- **Inquire contents of element generated by Set Edge Flag**: You may issue the Set Edge Flag (GPEF)
subroutine to set the edge flag to GEOMETRY ONLY in addition to the values known to ISO PHIGS Set
Edge Flag. If the structure element placed into the structure via GPEF has the edge flag indicated as
GEOMETRY ONLY, then using ISO PHIGS subroutines to inquire the contents of this element
generates error -606.

- **Inquire type and size or contents of Polygon 2/3**: You may issue the Polygon 2 (GPPG2) and
Polygon 3 (GPPG3) subroutines to generate structure elements with zero contours and contours with 0,
1 or 2 points. All of these are unavailable through ISO PHIGS. If the contents or the type and size of
such elements are inquired via ISO PHIGS, then the graPHIGS API generates error -606.

**Note**: An element type may be known to both ISO PHIGS and the GPxxxx interface, but the GPxxxx
interface may only know the data within the type. If such an element is inquired through ISO
PHIGS, then the graPHIGS API generates error -606.

Additional information, for FORTRAN Binding Only:

- **Inquire element type and size or content set by Insert Application Data**: Application data generated
by the Insert Application Data (GPINAD) subroutine may appear to have more data when inquired by
the Inquire Element Content (PQECO), the Inquire Current Element Content (PQCECO), the Inquire
Element Type and Size (POETS), or the Inquire Current Element Type and Size (POCETS) subroutine.
The length of ISO PHIGS application data is in multiples of 80 bytes. The equivalent GPxxxx subroutine
call, Insert Application Data (GPINAD) specifies the exact length of application data. The GPxxxx data
length is rounded up to the nearest multiple of 80 when the element is set by GPINAD and then
inquired by an ISO PHIGS FORTRAN subroutine.

- **Inquire workstation connection and type**: The connection identifier (integer) returned by the ISO
PHIGS FORTRAN Inquire Workstation Connection and Type (PQWKC) subroutine is unique to ISO
PHIGS. If you opened your workstation using Open Workstation (GPOPWS) or Create Workstation
(GPCRWS), then the connection identifier returned by PQWKC is not usable as the connection identifier
for the ISO PHIGS FORTRAN Open Workstation (POPWK) (). If you opened your workstation using
the ISO PHIGS Open Workstation (POPWK) subroutine, then PQWKC returns the connection identifier
used on that subroutine call.

### Display Subroutines

The following table lists display-related ISO PHIGS calls and their equivalent GPxxxx subroutine calls:

<table>
<thead>
<tr>
<th>ISO PHIGS Subroutine</th>
<th>Equivalent GPxxxx Subroutine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post Structure</td>
<td>GPARV (Associate Root with View) with a view parameter of 0</td>
</tr>
<tr>
<td>Unpost Structure</td>
<td>GPRDRW (Disassociate Root from Workstation)</td>
</tr>
<tr>
<td>Unpost All Structures</td>
<td>GPDarw (Disassociate All Roots from Workstation)</td>
</tr>
<tr>
<td>Inquire Posted Structures</td>
<td>GPQRV (Inquire Set of Roots in View) with a requested view of 0</td>
</tr>
</tbody>
</table>

### Input Events

For programs that combine GPxxxx subroutines and ISO PHIGS subroutines, the ISO PHIGS Await Event
subroutine could return such events as Link Switch In, Link Switch Out, Update Completion, Input
Overflow Events, Broadcast Message, Private Message, and Threshold Exceeded. For more information
on events supported by the graPHIGS API, see *The graPHIGS Programming Interface: Technical
Reference.*
Chapter 24. graPHIGS API Deviations from the ISO PHIGS Standard

This chapter describes the following graPHIGS API deviations from the ISO PHIGS standard:

- Unsupported subroutines
- Structure building
- Color components
- Input
- Traversal defaults
- Errors

Unsupported Subroutines

The graPHIGS API implementation of ISO PHIGS does not support the functionality required for:

- Metafile subroutine functions
- Incremental spatial search functions.

The C and FORTRAN bindings are supported, but the functionality is not implemented. A complete list of unsupported functions follows:

Metafile Functions

- Get Item Type From Metafile
- Interpret Item
- Read Item From Metafile
- Write Item To Metafile

Incremental Spatial Search Subroutines

- Incremental Spatial Search
- Incremental Spatial Search 3

Structure Building

For Set Annotation Text Character Up Vector and Set Character Up Vector, the graPHIGS API stores only the normalized vectors. Therefore, on the inquiry for these elements, the normalized vectors are returned. In addition, if the application passes a vector of length 0, the default vector is stored in the element. On a subsequent inquiry for the element, the graPHIGS API returns the default vector.

For Text 3, only the normalized text reference vectors are stored into the element. These normalized vectors are returned on an inquiry for the element. If the text reference vectors are linearly dependent (e.g., one vector is 0, or they are parallel) the default vectors of [1,0,0] and [0,1,0] are stored. An inquiry for this element returns the default vectors. In either case, the display is identical.

Color Components

Inquired color components of type SET may differ from the parameters passed to the setting routine due to conversions between the color model in which color components are stored and the model at the time the set and inquiry routines are called. Either subroutine call. In most cases these differences are due to numerical roundoff and truncation, and are slight. Differences can be large when CIELUV coordinates are
outside the gamut of the monitor primaries. The \( u' \) and \( v' \) coordinates are desaturated toward white to obtain a color within the gamut without changing the luminosity \( y \). The resulting RGB coordinates are saved and inquired.

**Input**

The ISO PHIGS Await Event subroutine call could return a class of Input Overflow Events in the graPHIGS API. See *The graPHIGS Programming Interface: Technical Reference* for additional information.

Initial pick path is ignored by the graPHIGS API. Therefore, on an inquiry for the pick device data, the pick path set by the application on the Initialize Pick is not returned.

A sample input device subroutine call forces an invalid mode error if the device is in request mode, but no error is issued if the device is in event or sample mode. The ISO PHIGS standard defines a 252 error if not in sample mode, but the graPHIGS API does not issue the error if the device is sampled while in event or sample mode.

A request input device subroutine call never forces an invalid mode error. ISO PHIGS defines a 251 error if you are not in request mode.

**Traversal Defaults**

During traversal, if a text precision is not available in the current font, then the graPHIGS API uses the highest precision available for that font. ISO PHIGS states that *STRING* precision of font 1 should be the default.

During traversal, if a text font is not available, the graPHIGS API substitutes font 1 at the same precision. ISO PHIGS states that *STRING* precision of font 1 should be used.

**Errors**

Copy All Elements from Structure and Execute Structure have no error defined in ISO PHIGS for “an attempt to execute the open structure”. The graPHIGS API rejects such an attempt and issues implementation error -125.

The following is a list of functions and related errors which can be detected only by a graPHIGS API nucleus. Therefore, it is possible that the error may not be generated during the subroutine call which caused it.

<table>
<thead>
<tr>
<th>Subroutine Function</th>
<th>Error Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize Input Device Subroutine functions</td>
<td>251</td>
</tr>
<tr>
<td>Delete Element Between Labels</td>
<td>206</td>
</tr>
<tr>
<td>Set Element Pointer at Label</td>
<td>205</td>
</tr>
<tr>
<td>Copy All Elements from Structure</td>
<td>-125</td>
</tr>
<tr>
<td>Change Structure Identifier</td>
<td>-129</td>
</tr>
<tr>
<td>Change Structure Identifier and References</td>
<td>-129</td>
</tr>
<tr>
<td>Change Structure References</td>
<td>-129</td>
</tr>
</tbody>
</table>
Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 003
11400 Burnet Road
Austin, TX 78758-3498
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:
(c) (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:
   AIX
   GDDM
   IBM
   RS/6000

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be the trademarks or service marks of others.
Readers’ Comments — We’d Like to Hear from You

The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference

Publication No. SC33-8140-03

Overall, how satisfied are you with the information in this book?

<table>
<thead>
<tr>
<th>Overall satisfaction</th>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
</table>

How satisfied are you that the information in this book is:

<table>
<thead>
<tr>
<th></th>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accurate</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy to find</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy to understand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Well organized</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applicable to your tasks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes  □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.