SunPHIGS 3.0
Sun-specific Reference Manual

sun
microsystems

Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View, CA 94043

Part No: 801-4367-10
Revision A, April 1993
SunPHIGS Release 3.0

NAME

INTRODUCTION TO SUN-SPECIFIC FUNCTIONS – the introduction to the Sun-specific extensions in the SunPHIGS graphics library

DESCRIPTION

The SunPHIGS graphics library offers Sun-specific extensions to PHIGS.

Sun-specific Extensions to the PHIGS Standard

The SunPHIGS library contains the following optional functions that are not in the PHIGS specification, but are extremely useful in the Sun environment.

WORKSTATION TYPE CREATE
WORKSTATION TYPE DESTROY
WORKSTATION TYPE GET
WORKSTATION TYPE SET

This manual describes the following Sun-specific functions:

- Escapes
- 2D and 3D generalized drawing primitives
- generalized structure elements

Sun-specific Functions

The following functions are implemented as Sun-specific SunPHIGS Extensions:

ESCAPE
escape (3P)

ESCAPE -1
escape_-1 (3P)

ESCAPE -3
escape_-3 (3P)

ESCAPE -4
escape_-4 (3P)

ESCAPE -5
escape_-5 (3P)

ESCAPE -6
escape_-6 (3P)

ESCAPE -7
escape_-7 (3P)

ESCAPE -8
escape_-8 (3P)

ESCAPE -9
escape_-9 (3P)

ESCAPE -11
escape_-11 (3P)

ESCAPE -12
escape_-12 (3P)

ESCAPE -13
　escape_-13 (3P)

ESCAPE -14
　escape_-14 (3P)

ESCAPE -15
　escape_-15 (3P)

ESCAPE -16
　escape_-16 (3P)

ESCAPE -17
　escape_-17 (3P)

ESCAPE -18
　escape_-18 (3P)

ESCAPE -19
　escape_-19 (3P)

ESCAPE -20
　escape_-20 (3P)

ESCAPE -21
　escape_-21 (3P)

ESCAPE -22
　escape_-22 (3P)

ESCAPE -23
　escape_-23 (3P)

ESCAPE -25
　escape_-25 (3P)

ESCAPE -26
　escape_-26 (3P)

ESCAPE -27
　escape_-27 (3P)

ESCAPE -28
　escape_-28 (3P)

ESCAPE -32
　escape_-32 (3P)

ESCAPE -33
　escape_-33 (3P)

ESCAPE -34
　escape_-34 (3P)

ESCAPE -35
　escape_-35 (3P)

ESCAPE -40
escape_-40 (3P)

ESCAPE -41
escape_-41 (3P)

ESCAPE -42
escape_-42 (3P)

ESCAPE -43
escape_-43 (3P)

ESCAPE -44
escape_-44 (3P)

ESCAPE -45
escape_-45 (3P)

ESCAPE -46
escape_-46 (3P)

GENERALIZED DRAWING PRIMITIVE
generalized_drawing_primitive (3P)

GENERALIZED DRAWING PRIMITIVE -2
generalized_drawing_primitive_-2 (3P)

GENERALIZED DRAWING PRIMITIVE -3
generalized_drawing_primitive_-3 (3P)

GENERALIZED DRAWING PRIMITIVE -4
generalized_drawing_primitive_-4 (3P)

GENERALIZED DRAWING PRIMITIVE -5
generalized_drawing_primitive_-5 (3P)

GENERALIZED DRAWING PRIMITIVE -6
generalized_drawing_primitive_-6 (3P)

GENERALIZED DRAWING PRIMITIVE -7
generalized_drawing_primitive_-7 (3P)

GENERALIZED DRAWING PRIMITIVE -8
generalized_drawing_primitive_-8 (3P)

GENERALIZED DRAWING PRIMITIVE -9
generalized_drawing_primitive_-9 (3P)

GENERALIZED DRAWING PRIMITIVE -10
generalized_drawing_primitive_-10 (3P)

GENERALIZED DRAWING PRIMITIVE -11
generalized_drawing_primitive_-11 (3P)

GENERALIZED DRAWING PRIMITIVE -12
generalized_drawing_primitive_-12 (3P)

GENERALIZED DRAWING PRIMITIVE -13
    generalized_drawing_primitive_-13 (3P)

GENERALIZED DRAWING PRIMITIVE -14
    generalized_drawing_primitive_-14 (3P)

GENERALIZED DRAWING PRIMITIVE -15
    generalized_drawing_primitive_-15 (3P)

GENERALIZED DRAWING PRIMITIVE -16
    generalized_drawing_primitive_-16 (3P)

GENERALIZED DRAWING PRIMITIVE -17
    generalized_drawing_primitive_-17 (3P)

GENERALIZED DRAWING PRIMITIVE -18
    generalized_drawing_primitive_-18 (3P)

GENERALIZED DRAWING PRIMITIVE 3
    generalized_drawing_primitive_3 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -2
    generalized_drawing_primitive_3_-2 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -3
    generalized_drawing_primitive_3_-3 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -4
    generalized_drawing_primitive_3_-4 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -5
    generalized_drawing_primitive_3_-5 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -6
    generalized_drawing_primitive_3_-6 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -7
    generalized_drawing_primitive_3_-7 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -8
    generalized_drawing_primitive_3_-8 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -9
    generalized_drawing_primitive_3_-9 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -10
    generalized_drawing_primitive_3_-10 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -11
    generalized_drawing_primitive_3_-11 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -12
    generalized_drawing_primitive_3_-12 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -13
    generalized_drawing_primitive_3_-13 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -14
generalized_drawing_primitive_3_-14 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -15
generalized_drawing_primitive_3_-15 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -16
generalized_drawing_primitive_3_-16 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -17
generalized_drawing_primitive_3_-17 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -18
generalized_drawing_primitive_3_-18 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -19
generalized_drawing_primitive_3_-19 (3P)

GENERALIZED STRUCTURE ELEMENT
generalized_structure_element (3P)

GENERALIZED STRUCTURE ELEMENT -1
generalized_structure_element_-1 (3P)

GENERALIZED STRUCTURE ELEMENT -4
generalized_structure_element_-4 (3P)

GENERALIZED STRUCTURE ELEMENT -5
generalized_structure_element_-5 (3P)

GENERALIZED STRUCTURE ELEMENT -6
generalized_structure_element_-6 (3P)

GENERALIZED STRUCTURE ELEMENT -7
generalized_structure_element_-7 (3P)

GENERALIZED STRUCTURE ELEMENT -8
generalized_structure_element_-8 (3P)

GENERALIZED STRUCTURE ELEMENT -9
generalized_structure_element_-9 (3P)

GENERALIZED STRUCTURE ELEMENT -10
generalized_structure_element_-10 (3P)

GENERALIZED STRUCTURE ELEMENT -11
generalized_structure_element_-11 (3P)

ESCAPE	— invoke an implementation-dependent function
ESCAPE -1	— enforce synchronized error reporting
ESCAPE -3	— disable or enable PHIGS DC bounds checking
ESCAPE -4	— map an X-drawable point to a PHIGS pick path
ESCAPE -5	— map a list of X-drawable points to PHIGS World Coordinate (WC) points
ESCAPE -6	— redraw a list of PHIGS a workstation window regions

ESCAPE -7	— synchronize with a workstation
ESCAPE -8	— raster resize
ESCAPE -9	— inquire character set facilities
ESCAPE -11	— inquire for fonts available for character set
ESCAPE -12	— inquire for extended text extent
ESCAPE -1	— inquire for whole physical window size
ESCAPE -14	— inquire for XGL window raster
ESCAPE -15	— specify type of archive
ESCAPE -16	— inquire for workstation type support
ESCAPE -17	— set visual representation state
ESCAPE -18	— update a set of region workstations on a canvas
ESCAPE -19	— add an input device association
ESCAPE -20	— remove input device association
ESCAPE -21	— set workstation's cursor type and colour
ESCAPE -22	— set double buffer mode for quick update method (QUM)
ESCAPE -23	— query double buffer mode for quick update method (QUM)
ESCAPE -25	— define user-definable polymarker in PHIGS description table
ESCAPE -26	— inquire for user-defined marker definition from PHIGS description table
ESCAPE -27	— inquire for list of available user-definable marker types from PHIGS description table
ESCAPE -28	— simulate pick
ESCAPE -32	— set workstation's NURB cache mode
ESCAPE -33	— inquire for workstation's NURB cache mode
ESCAPE -34	— set workstation's silhouette mode
ESCAPE -35	— inquire for workstation's silhouette mode
ESCAPE -40	— set workstation anti-aliasing mode
ESCAPE -41	— inquire for workstation anti-aliasing mode
ESCAPE -42	— inquire for workstation facilities for wide-stroke control
ESCAPE -43	— set workstation transparency quality / performance trade off
ESCAPE -44	— inquire for workstation transparency quality / performance trade off
ESCAPE -45	— set view representation for left eye
ESCAPE -46	— inquire for view representation for left eye
GENERALIZED DRAWING PRIMITIVE	— create 2D GDP elements such as polyline set, circle, or ellipse
GENERALIZED DRAWING PRIMITIVE -2	

— create 2D GDP circle

GENERALIZED DRAWING PRIMITIVE -3
— create 2D GDP circular arc

GENERALIZED DRAWING PRIMITIVE -4
— create 2D GDP circular arc close

GENERALIZED DRAWING PRIMITIVE -5
— create 2D GDP annotation circle

GENERALIZED DRAWING PRIMITIVE -6
— create 2D GDP annotation circular arc

GENERALIZED DRAWING PRIMITIVE -7
— create 2D GDP annotation circular arc close

GENERALIZED DRAWING PRIMITIVE -8
— create a 2D GDP ellipse

GENERALIZED DRAWING PRIMITIVE -9
— create a 2D GDP elliptical arc

GENERALIZED DRAWING PRIMITIVE -10
— create a 2D GDP elliptical arc close

GENERALIZED DRAWING PRIMITIVE -11
— create a 2D GDP annotation ellipse

GENERALIZED DRAWING PRIMITIVE -12
— create a 2D GDP annotation elliptical arc

GENERALIZED DRAWING PRIMITIVE -13
— create a 2D GDP annotation elliptical arc close

GENERALIZED DRAWING PRIMITIVE -14
— create a 2D image

GENERALIZED DRAWING PRIMITIVE -15
— create a 2D GDP rectangular grid

GENERALIZED DRAWING PRIMITIVE -16
— a 2D GDP radial grid

GENERALIZED DRAWING PRIMITIVE -17
— create 2D mono-encoded text

GENERALIZED DRAWING PRIMITIVE -18
— create 2D mono-encoded annotation text relative

GENERALIZED DRAWING PRIMITIVE 3
— create 3D GDP elements such as polyline set 3, circle 3, or ellipse 3

GENERALIZED DRAWING PRIMITIVE 3 -2
— create 3D GDP circle 3

GENERALIZED DRAWING PRIMITIVE 3 -3
— create 3D GDP circular arc 3

GENERALIZED DRAWING PRIMITIVE 3 -4
— create 3D GDP circular arc close 3

GENERALIZED DRAWING PRIMITIVE 3 -5
— create 3D GDP annotation circle 3

GENERALIZED DRAWING PRIMITIVE 3 -6
— create 3D GDP annotation circular arc 3

GENERALIZED DRAWING PRIMITIVE 3 -7
— create 3D GDP annotation circular arc close 3

GENERALIZED DRAWING PRIMITIVE 3 -8
— create 3D GDP ellipse 3

GENERALIZED DRAWING PRIMITIVE 3 -9
— create 3D GDP elliptical arc 3

GENERALIZED DRAWING PRIMITIVE 3 -10
— create 3D GDP elliptical arc close 3

GENERALIZED DRAWING PRIMITIVE 3 -11
— create 3D GDP annotation ellipse 3

GENERALIZED DRAWING PRIMITIVE 3 -12
— create 3D GDP annotation elliptical arc 3

GENERALIZED DRAWING PRIMITIVE 3 -13
— create 3D GDP annotation elliptical arc close 3

GENERALIZED DRAWING PRIMITIVE 3 -14
— create a 3D image

GENERALIZED DRAWING PRIMITIVE 3 -15
— create a 3D GDP rectangular grid

GENERALIZED DRAWING PRIMITIVE 3 -16
— 3D GDP radial grid

GENERALIZED DRAWING PRIMITIVE 3 -17
— create 3D mono-encoded text

GENERALIZED DRAWING PRIMITIVE 3 -18
— create 3D mono-encoded annotation text relative

GENERALIZED DRAWING PRIMITIVE 3 -19
— create 3D hinted NURBS surface

GENERALIZED STRUCTURE ELEMENT
— create generalized structure element

GENERALIZED STRUCTURE ELEMENT -1
— create structure element to set highlight colour index

GENERALIZED STRUCTURE ELEMENT -4
— create structure element to set slant angle for text

GENERALIZED STRUCTURE ELEMENT -5

— create structure element to set slant angle for annotation text

GENERALIZED STRUCTURE ELEMENT -6
— create structure element to set current solid polygon type

GENERALIZED STRUCTURE ELEMENT -7
— create structure element to control width-1 stroke anti-aliasing

GENERALIZED STRUCTURE ELEMENT -8
— create structure element to control width of vector strokes used to draw vector text

GENERALIZED STRUCTURE ELEMENT -9
— create structure element to control appearance of wide stroke end-caps

GENERALIZED STRUCTURE ELEMENT -10
— create structure element to control appearance of wide-stroke joints

GENERALIZED STRUCTURE ELEMENT -11
— create structure element to set the current silhouette edge flag

SEE ALSO

INTRO (3PP)
INTRO PHIGS (3P)
INTRO (7P)

NAME

ESCAPE – invoke implementation-dependent function

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE provides a standard way of performing a nonstandard function. A single function can access any of the implementation-dependent escape functions, which can perform a nonstandard action or return nonstandard information.

Each escape function has its own page in the this manual. The name of the function file is escape_–*xx*, where –*xx* is replaced by the function identifier (for example, escape_-2). Thus, when the page header is ESCAPE -2, the function identifier for the escape is -2.

The SunPHIGS ESCAPE can be used to perform the following tasks:

- Enforce synchronized error reporting
- Disable or enable PHIGS DC bounds checking
- Map an X Drawable point to a PHIGS pick path
- Map a list of X Drawable points to PHIGS World Coordinate (WC) points
- Redraw a list of a PHIGS workstation’s window regions
- Synchronize with a workstation
- Notify PHIGS of resize event
- Set an X Drawable Region’s WS rect
- Inquire character set facilities
- Inquire fonts available for a character set
- Inquire extended text extent

- Inquire the physical window size
- Specify type of archive
- Inquire the XGL raster
- Inquire workstation type support
- Set visual representation state
- Update a set of region workstations on a drawable
- Add an input device association
- Remove an input device association
- Set a workstation's cursor type and colour
- Set / inquire the double buffer mode for quick updates
- Define a user-definable marker type
- Inquire user defined marker type
- Inquire list of user-definable marker types
- Simulate a pick
- Set / inquire the workstation NURB cache mode
- Set / inquire the silhouette edge mode
- Set / inquire workstation mode for image anti-aliasing
- Inquire workstation facilities for wide-stroke control
- Set / inquire transparency-quality / performance trade-off
- Set / inquire a view representation for the left eye

C Input Parameters

func_id The function identifier defines the nonstandard function to perform. Valid identifiers are:

−1	PUESC_ERRSYNC	*Error synchronization*
−3	PUESC_IGNORE_DC_ERRORS	*Disable or enable PHIGS DC bounds checking*
−4	PUESC_DRAWABLE_POINT_TO_PICK	*Map an X-drawable point to a PHIGS DC pick path*
−5	PUESC_DRAWABLE_POINTS_TO_WC	*Map a list of X-drawable points to PHIGS WC points*
−6	PUESC_REDRAW_REGIONS	*Redraw a list of a PHIGS workstation's window regions*
−7	PUESC_WS_SYNCH	*Synchronize with a workstation*
−8	PUESC_RASTER_RESIZE	*Raster resize*
−9	PUESC_SET_REGION_WS_RECT	*Set an X Drawable Region WS rect*
−10	PUESC_INQ_CHAR_SET_FACILITIES	*Obtain character set facilities*
−11	PUESC_INQ_FONTS_FOR_CHAR_SET	*Obtain fonts available for a character set*
−12	PUESC_INQ_EXT_TEXT_EXTENT	*Obtain extended text extent*

−13	PUESC_INQ_WS_PHYS_SIZE	*Obtain the physical window size*
−14	PUESC_INQ_XGL_RASTER	*Obtain XGL raster handle*
−15	PUESC_SET_ARCHIVE_TYPE	*Specify type of archive*
−16	PUESC_INQ_WS_TYPE_SUPPORT	*Obtain workstation type support*
−17	PUESC_SET_VIS_REP	*Set visual representation state*
−18	PUESC_UPDATE_WS_SET	*Update a set of region workstations on a drawable*
−19	PUESC_ADD_DEVICE_ASSOCIATION	*Add input device association*
−20	PUESC_REMOVE_DEVICE_ASSOCIATION	*Remove input device association*
−21	PUESC_SET_CURSOR	*Set cursor type*
−22	PUESC_SET_DBL_BUF_QUM	*Set the double buffer mode for quick updates*
−23	PUESC_INQ_DBL_BUF_QUM	*Obtain the double buffer mode for quick updates*
−25	PUESC_DEFINE_MARKER	*Define a marker type*
−26	PUESC_INQ_UDEF_MARKER	*Obtain the user-defined marker*
−27	PUESC_INQ_UDEF_MARKER_TYPES	*Obtain list of user-defined marker types*
−28	PUESC_SIMULATE_PICK	*Simulate a pick*
−32	PUESC_SET_NURB_CACHE_MODE	*Set the NURB cache mode*
−33	PUESC_INQ_NURB_CACHE_MODE	*Inquire the NURB cache mode*
−34	PUESC_SET_SIL_EDGE_MODE	*Set the silhouette edge mode*
−35	PUESC_INQ_SIL_EDGE_MODE	*Inquire the silhouette edge mode*
−40	PUESC_SET_AA_MODE	*Set workstation mode for image anti-aliasing*
−41	PUESC_INQ_AA_MODE	*Obtain workstation mode for image anti-aliasing*
−42	PUESC_INQ_WIDE_STROKE_FACILITIES	*Obtain workstation facilities for wide-stroke control*
−43	PUESC_SET_TRANSP_QUALITY	*Set transparency-quality/performance trade-off*
−44	PUESC_INQ_TRANSP_QUALITY	*Obtain transparency-quality*
−45	PUESC_SET_LEFT_VIEW_REP3	*Set a view representation for the left eye*
−46	PUESC_INQ_LEFT_VIEW_REP3	*Obtain view representation for the left eye*

in A pointer to a Pescape_in_data union containing the input data record required by the escape function specified in the *func_id* parameter. Pescape_in_data is defined in phigs.h as follows:

```
typedef union {
        /* implementation-dependent */
    struct Pescape_in_u1 {
        Perrsync                sync_on;
```

```
} escape_in_u1;
struct Pescape_in_u3 {
        Pint                    ignore_DC_errors;   /∗ ignore errors if
                                                       non-zero ∗/
} escape_in_u3;
struct Pescape_in_u4 {
        Pint                    ws_id;
        Ppoint                  point;              /∗ a drawable point
                                                       — not DC ∗/
        Pfloat                  ap_size;            /∗ aperture size, half-
                                                       width in DC units ∗/
        Ppath_order             order;
        Pint                    depth;              /∗ depth of path to
                                                       return ∗/
        Pint                    pet;
        Pecho_switch            echo_switch;
        Plimit3                 echo_volume;
        Pfilter                 filter;             /∗ detectability
                                                       filter ∗/
} escape_in_u4;
struct Pescape_in_u5 {
        Pint                    ws_id;              /∗ workstation
                                                       identifier ∗/
        Ppoint_list3            points;             /∗ x,y are drawable
                                                       coords, z is DC ∗/
} escape_in_u5;
struct Pescape_in_u6 {
        Pint                    ws_id;              /∗ workstation
                                                       identifier ∗/
        Pint                    num_regions;        /∗ number of
                                                       regions ∗/
        XRectangle              *regions;           /∗ regions to
                                                       redraw ∗/
} escape_in_u6;
struct Pescape_in_u7 {
        Pint                    ws_id;              /∗ workstation
                                                       identifier ∗/
} escape_in_u7;
struct Pescape_in_u8 {
        Pint                    ws_id;              /∗ workstation
                                                       identifier ∗/
} escape_in_u8;
struct Pescape_in_u9 {
        Pint                 ws_id;          /* workstation
                                                       identifier */
        Prect                rect;           /* rectangle on
                                                       drawable to
                                                       place the
                                                       region WS
                                                       in — origin is
                                                       at lower left,
                                                       coords are
                                                       in pixels */
} escape_in_u9;
struct Pescape_in_u10 {
        Pint                 ws_type;        /* workstation
                                                       type */
} escape_in_u10;
struct Pescape_in_u11 {
        Pint                 ws_type;        /* workstation
                                                       type */
        Pint                 char_set;       /* charset to get
                                                       font informa-
                                                       tion for */
} escape_in_u11;
struct Pescape_in_u12 {
        Pint                 ws_type;        /* workstation
                                                       type */
        Pint                 font;           /* text font */
        Pfloat               expan;          /* char expansion
                                                       factor */
        Pfloat               space;          /* char spacing */
        Pfloat               ht;             /* char height */
        Ptext_path           path;           /* text path */
        Phor_text_align      hor;            /* horizontal
                                                       alignment */
        Pvert_text_align     vert;           /* vertical
                                                       alignment */
        Phigs_i_string       i_string;       /* text I-String */
} escape_in_u12;
struct Pescape_in_u13 {
        Pint                 ws_id;          /* workstation
                                                       identifier */
} escape_in_u13;
struct Pescape_in_u14 {
        Pint                 ws_id;          /* workstation
```

```
                                                          identifier */
} escape_in_u14;
struct Pescape_in_u15 {
        Phigs_ar_mode           ar_mode;
} escape_in_u15;
struct Pescape_in_u16 {
        Pint                    ws_type;
} escape_in_u16;
struct Pescape_in_u17 {
        Pint                    ws_id;              /* workstation
                                                       identifier */
        Pvisual_st              visual_st;          /* update state */
} escape_in_u17;
struct Pescape_in_u18 {
        XID                     drawable_id;
        Pint_list               ws_list;
} escape_in_u18;
struct Pescape_in_u19 {
        Pint                    ws_id;              /* workstation
                                                       identifier */
        Phigs_in_assoc_master   master;
        Phigs_in_assoc_slave    slave;
} escape_in_u19;
struct Pescape_in_u20 {
        Pint                    ws_id;              /* workstation
                                                       identifier */
        Phigs_in_assoc_master   master;
        Phigs_in_assoc_slave    slave;
} escape_in_u20;
struct Pescape_in_u21 {
        Pint                    ws_id;              /* workstation
                                                       identifier */
        Phigs_cursor_which      cursor_which;       /* cursor to change */
        Phigs_cursor_type       cursor_type;        /* cursor type */
        Prgb                    cursor_colr;        /* cursor colour */
} escape_in_u21;
struct Pescape_in_u22 {
        Pint                      ws_id;            /* workstation
                                                       identifier */
        Phigs_db_qum_mode         db_qum_mode;      /* double buffer
                                                       QUM mode */
} escape_in_u22;
```

```
struct Pescape_in_u23 {
        Pint                        ws_id;              /* workstation
                                                           identifier */
} escape_in_u23;
struct Pescape_in_u25 {
        Pint                        type;               /* marker type */
        Phigs_marker_data           marker;             /* polyline marker
                                                           data */
} escape_in_u25;
struct Pescape_in_u26 {
        Pint                        type;               /* marker type */
} escape_in_u26;
struct Pescape_in_u28 {
        Pint                        ws_id;               /* workstation
                                                           identifier */
        Pint                        max_picks;           /* maximum num-
                                                           ber of picks
                                                           to return */
        Pint                        max_depth;           /* maximum depth
                                                           of each pick */
        Pint                        options;             /* option flags */
        Plimit                      pick_ap;             /* pick aperture */
        Pfilter                     filter;              /* pick inclusion / ex-
                                                           clusion filter */
} escape_in_u28;
struct Pescape_in_u32 {
        Pint                        ws_id;                /* workstation
                                                           identifier */
        Phigs_nurb_cache_mode       cache_mode;           /* NURB cache
                                         mode */
} escape_in_u32;
struct Pescape_in_u33 {
        Pint                        ws_id;              /* workstation
                                                           identifier */
} escape_in_u33;
struct Pescape_in_u34 {
        Pint                        ws_id;              /* workstation
                                                           identifier */
        Phigs_sil_edge_mode         sil_edge_mode;      /* silhouette edge
                                                           mode */
} escape_in_u34;
struct Pescape_in_u35 {
        Pint                        ws_id;              /* workstation
```

```
                                                    identifier */
} escape_in_u35;
struct Pescape_in_u40 {
        Pint                    ws_id;          /* workstation
                                                   identifier */
        Phigs_aalias_mode       aa_mode;        /* none, stroke, or
                                                   image */
        Phigs_aalias_rec        aa_record;      /* anti-aliasing
                                                   record */
} escape_in_u40;
struct Pescape_in_u41 {
        Pint                    ws_id;          /* workstation
                                                   identifier */
        Pinq_type               type;           /* PSET or
                                                   PREALIZED */
} escape_in_u41;
struct Pescape_in_u42 {
        Pint                    ws_type;        /* workstation
                                                   type */
        Pattrs                  primitive;      /* attribute set of
                                                   interest */
} escape_in_u42;
struct Pescape_in_u43 {
        Pint                    ws_id;          /* workstation
                                                   identifier */
        Phigs_transp            quality;        /* transparency qual-
                                                   ity tradeoff */
} escape_in_u43;
struct Pescape_in_u44 {
        Pint                    ws_id;          /* workstation
                                                   identifier */
        Pinq_type               type;           /* PSET or
                                                   PREALIZED */
} escape_in_u44;
struct Pescape_in_u45 {
        Pint                    ws_id;          /* workstation
                                                   identifier */
        Pint                    ind;            /* index of view
                                                   representation
                                                   to set */
        Pview_rep3              *view_rep;      /* view represen-
                                                   tation, left eye */
} escape_in_u45;
```

```
        struct Pescape_in_u46 {
                Pint                    ws_id;          /* workstation
                                                           identifier */
                Pint                    ind;            /* index of view
                                                           representation
                                                           to set*/
        } escape_in_u46;
} Pescape_in_data;
```

store The memory buffer PHIGS is to use for storing the information returned. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to the type Pescape_out_data. If the escape function specified in the *func_id* parameter returns any data, an output data record of type Pescape_out_data will be allocated from the memory buffer specified by the parameter *store*, and its address will be stored in **out*. Pescape_out_data is defined in phigs.h as follows:

```
typedef union {
                                        /* implementation-dependent */
        struct Pescape_out_u4 {
                Pin_status              status;
                Ppick_path              pick;
        } escape_out_u4;
        struct Pescape_out_u5 {
                Pint                    view_index;
                Ppoint_list3            points;
        } escape_out_u5;
        struct Pescape_out_u10 {
                Pint                    err_ind;            /* error
                                                               indicator */
                Phigs_char_set_facs     facs;               /*list of available
                                                              character sets*/
        } escape_out_u10;
        struct Pescape_out_u11 {
                Pint                    err_ind;            /* error
                                                               indicator */
                Phigs_font_facs         facs;               /* text facilities */
        } escape_out_u11;
        struct Pescape_out_u12 {
                Pint                    err_ind;            /* error
                                                               indicator */
                Prect                   rect;               /* extent
                                                               rectangle */
                Ppoint                  offset;          /* concatenation
                                                               offset */
        } escape_out_u12;
        struct Pescape_out_u13 {
                Pint                    err_ind;         /* error
                                                               indicator */
                Pint                    width;           /* window phys-
                                                               ical width */
                Pint                    ht;              /* window phys-
                                                               ical height */
        } escape_out_u13;
        struct Pescape_out_u14 {
                Pint                    err_ind;         /* error
                                                               indicator */
                void                    *win_ras;        /* XGL window
                                                               raster object */
        } escape_out_u14;
        struct Pescape_out_u16 {
                Pint                    err_ind;         /* error
                                                               indicator */
                Phigs_ws_type_support   ws_type_sup;     /* workstation
                                                               type
                                                               support */
        } escape_out_u16;
        struct Pescape_out_u23 {
                Pint                      err_ind;       /* error
                                                               indicator */
                Phigs_db_qum_mode         db_qum_mode;   /* double buffer
                                                               QUM mode */
        } escape_out_u23;
        struct Pescape_out_u26 {
                Pint                    err_ind;         /* error
                                                              indicator */
                Ppoint_list_list        marker;          /* output pointer
                                                              to buffer */
        } escape_out_u26;
        struct Pescape_out_u27 {
                Pint                    err_ind;         /* error
                                                              indicator */
                Pint_list               type_list;       /* list of user-
                                                              defined
                                                               marker types */
        } escape_out_u27;
```

```
struct Pescape_out_u28 {
        Pint                   err_ind;                /∗ error
                                                          indicator ∗/
        Pint                   num_picks;              /∗ number of
                                                          picks
                                                          returned ∗/
        Ppick_path             ∗picks;                 /∗ returned pick
                                                           paths ∗/
        Pint                   total_picks;            /∗ total number
                                                           of picks ∗/
        Pint                   total_max_depth;        /∗ max. depth of
                                                           any pick ∗/
        Pint                   total_pick_path_elems; /∗ total number
                                                          of pick path
                                                          elements
                                                          needed in the
                                                          input pick
                                                          buffer ∗/
} escape_out_u28;
struct Pescape_out_u33 {
        Pin                    err_ind;                /∗ error
                                                           indicator  ∗/
        Phigs_nurb_cache_mode  cache_mode;             /∗ NURB cache
                                                           mode ∗/
} escape_out_u33;
struct Pescape_out_u35 {
        Pint                   error_ind;              /∗ error
                                                           indicator ∗/
        Pupd_st                state;                  /∗ silhouette edge
                                                           update state ∗/
        Phigs_sil_edge_mode    req_mode;               /∗ requested sil.
                                                           edge mode ∗/
        Phigs_sil_edge_mode    cur_mode;               /∗ current sil.
                                                           edge mode ∗/
} escape_out_u35;
struct Pescape_out_u41 {
        Pint                   err_ind;                /∗ error
                                                           indicator ∗/
        Phigs_aalias_mode      aa_mode;                /∗ aa mode ∗/
        Phigs_aalias_rec       aa_record;              /∗ num of aa
                                                           passes ∗/
} escape_out_u41;
struct Pescape_out_u42 {
        Pint            err_ind;        /* error
                                           indicator */
        Pint_list       end_caps;       /* list of end-caps
                                           supported */
        Pint_list       join_styles;    /* list of join-
                                           styles
                                           supported */
    } escape_out_u42;
    struct Pescape_out_u44 {
        Pint            err_ind;        /* error
                                           indicator */
        Phigs_transp    quality;        /* transparency
                                           quality
                                           tradeoff */
    } escape_out_u44;
    struct Pescape_out_u46 {
        Pint            err_ind;        /* error
                                           indicator */
        Pupd_st         upd_st;         /* view
                                           transformation
                                           update state */
        Pview_rep3      cur_rep;        /* current left
                                           eye view
                                           rep */
        Pview_rep3      req_rep;        /* requested
                                           left eye
                                           view rep */
    } escape_out_u46;
} Pescape_out_data;
```

FORTRAN Input Parameters

FCTID The function identifier defines the nonstandard function to perform. Valid identifiers are:

−1	PUESCERRSYNC	*Error synchronization*
−3	PUESCIGNOREDC	*Disable or enable PHIGS DC bounds checking*
−4	PUESCDRPTTOPICK	*Map an X-drawable point to a PHIGS DC pick path*
−5	PUESCDRPTTOWC	*Map a list of X-drawable points to PHIGS WC points*
−6	PUESCREDRAWREG	*Redraw a list of a PHIGS workstation's window regions*
−7	PUESCWSSYNC	*Synchronize with a workstation*
−8	PUESCRASRESIZE	*Raster resize*

−9	PUESCSETREGRECT	*Set an X Drawable Region WS rect*
−10	PUESCINQCSFAC	*Obtain character set facilities*
−11	PUESCINQFTCS	*Obtain fonts available for a character set*
−12	PUESCINQXTXEXT	*Obtain extended text extent*
−13	PUESCPHYSSIZE	*Obtain the physical window size*
−14	PUESCINQXGLRAS	*Obtain XGL Raster handle*
−15	PUESCSETARMODE	*Specify type of archive*
−16	PUESCINQWSTSUP	*Obtain workstation type support*
−17	PUESCSETVISREP	*Set visual representation state*
−18	PUESCUPDATESET	*Update a set of region workstations on a drawable*
−19	PUESCADDASSOC	*Add Input device association*
−20	PUESCREMOVEASSOC	*Remove input device association*
−21	PUESCSETCURSOR	*Set cursor type*
−22	PUESCSETDBQUM	*Set the double buffer mode for quick updates*
−23	PUESCINQDBQUM	*Obtain the double buffer mode for quick updates*
−25	PUESCDEFINEMKR	*Define a marker type*
−26	PUESCINQUDEFMKR	*Obtain the user defined marker*
−27	PUESCINQUDMKTPS	*Obtain list of user defined marker types*
−28	PUESCSIMPICK	*Simulate a pick*
−32	PUESCSETNURBCACHEMODE	*Set the NURB cache mode*
−33	PUESCINQNURBCACHEMODE	*Inquire the NURB cache mode*
−34	PUESCSETSILEDGEMODE	*Set the silhouette edge mode*
−35	PUESCINQSILEDGEMODE	*Inquire the silhouette edge mode*
−40	PUESCSETAAMODE	*Set workstation mode for image anti-aliasing*
−41	PUESCINQAAMODE	*Obtain workstation mode for image anti-aliasing*
−42	PUESCINQWIDESTROKE	*Obtain workstation facilities for wide-stroke control*
−43	PUESCSETTRANSPQUAL	*Set transparency-quality/performance trade-off*
−44	PUESCINQTRANSPQUAL	*Obtain transparency quality*
−45	PUESCSETLEFTVIEWREP3	*Set a view representation for the left eye*
−46	PUESCINQLEFTVIEWREP3	*Obtain view representation for the left eye*

LIDR The dimension of *IDR* (returned by the PACK DATA RECORD subroutine).

IDR An escape data record packed by the PACK DATA RECORD subroutine containing the input data record required by the escape function specified in *FCTID*. See the man pages for individual escapes for the format of this data record.

MLODR The maximum length of the output data record, *ODR*.

FORTRAN Output Parameters

LODR The length used in the output data record array, *ODR*.

ODR The array that stores the output data record. It can be unpacked by using UNPACK DATA RECORD. See the man pages for individual escapes for the format of this data record.

ERRORS

002 Ignoring function, function requires state (PHOP, *, *, *)

350 Warning, the specified escape is not available on one or more workstations in this implementation. The escape will be processed by those workstations on which it is available

351 Ignoring function, one of the fields within the escape data record is in error

SEE ALSO

PACK DATA RECORD (3P)
UNPACK DATA RECORD (3P)
SET ELEMENT POINTER AT PICK IDENTIFIER (3PP)

NAME

ESCAPE -1 – enforce synchronized error reporting

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*not used.*
Pescape_out_data	**out;	*OUT not used*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -1 allows the enforcement of synchronized error reporting.

C Input Parameters

func_id The function identifier for this escape is PUESC_ERRSYNC, defined in phigs.h to be -1.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u1 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u1 {
                Perrsync    sync_on;
        } escape_in_u1;
} Pescape_in_data;
```

sync_on One of the following enumerated values:

0	PERRSYNC_OFF	Asynchronous error reporting
1	PERRSYNC_ON	Synchronous error reporting

store Not used.

C Output Parameters

out There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCERRSYNC, which is defined in phigs77.h to be -1.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with the following parameters:

IL The number of integers = 1.

IA Contains one integer value, specifying the desired error synchronization mode, one of the following enumerated values:

0	PERRSYNCOFF	Asynchronous error reporting
1	PERRSYNCON	Synchronous error reporting

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

As an performance-enhancing measure, errors detected by a SunPHIGS function may not be reported before that function call returns to the application. All errors detected are reported eventually, during some call into a SunPHIGS function. When the error is reported, the function number given is that of the function in which the error was *detected*. (That is, it is not necessarily the number of the function currently invoked when the error is *reported*.) Some errors are not reported until a subsequent call into SunPHIGS.

When an application requires synchronous error reporting, use ESCAPE -1; the error synchronization escape function. If the error synchronization mode is *synchronous error reporting*, then all errors detected inside a SunPHIGS function are reported before that function returns to the application. If the error synchronization mode is *asynchronous error reporting* (the default), then errors detected inside a SunPHIGS function are reported either before that function returns or during a subsequent SunPHIGS function call.

ERRORS

002 Ignoring function, function requires state (PHOP, ∗, ∗, ∗)

SEE ALSO

ERROR HANDLING (3P)

NAME

ESCAPE -3 – disable or enable PHIGS DC bounds checking

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*not used*
Pescape_out_data	**out;	*OUT not used*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -3 disables or enables Device Coordinate (DC) bounds checking by PHIGS functions with DC input parameters (such as SET WORKSTATION VIEWPORT and the input device initialization functions).

C Input Parameters

func_id The function identifier for this escape is PUESC_IGNORE_DC_ERRORS, which is defined in phigs.h to be -3.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u3 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u3 {
                Pint    ignore_DC_errors;    /* ignore errors if non-zero */
        } escape_in_u3;
} Pescape_in_data;
```

ignore_DC_errors indicates if DC bounds checking is enabled (0, the default) or disabled (non-zero).

store Not used.

C Output Parameters

out There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCIGNOREDC, which is defined in phigs77.h to be -3.

LIDR The dimension of IDR (returned by PACK DATA RECORD (3P)).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD (3P) when called with the following parameters:

IL	The number of integers = 1.
IA	Contains one integer value, which is either 0, to enable bounds checking (the default condition) or non-zero to disable bounds checking.
RL	The number of real values = 0.
SL	The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

This escape disables or enables bounds checking of DC parameters specified to PHIGS functions. A number of PHIGS workstation functions have DC values as parameters. Normally these values are checked against the current DC size of the workstation when the function is called, and an error is reported if any of the values lie outside the current limits. The function then returns without performing the desired operation. This escape is used to override that behavior by indicating that DC values outside the current workstation limits should be accepted. This allows such DC values to be entered into the workstation state list and used in subsequent PHIGS operations. The use of these values in not defined by the PHIGS standard, but PHIGS will use them as it would values that are within range. This may produce results that do not conform to the PHIGS standard. It is the application's responsibility to ensure that the results of PHIGS' use of these out-of-range values are meaningful.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

SEE ALSO

ESCAPE (3P)

NAME

ESCAPE -4 – map an X-Drawable point to a PHIGS pick path

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      pointer to the buffer
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID         function identification
INTEGER         LIDR          dimension of input data record array
CHARACTER*80    IDR(LIDR)     input data record
INTEGER         MLODR         maximum length of output data record
INTEGER         LODR          OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, WSOP)

DESCRIPTION

Purpose

ESCAPE -4 attempts to map a single X-Drawable coordinate point to a PHIGS pick path using the specified workstation.

ESCAPE -4 provides similar functionality to a PHIGS pick device, but it uses a point provided by the application rather than directly by the operator. This escape is not associated with any PHIGS input device and can be used without initializing one. It is allows the application to use the standard X Event mechanism to implement its own pick device.

C Input Parameters

Applications using the C binding must create a buffer for this function to use as memory space for storing data associated with the device state. This buffer is passed as the *store* argument.

The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The data record within the store buffer is accessed by the pointer pointed to by *out*.

func_id The function identifier for this escape is PUESC_DRAWABLE_POINT_TO_PICK, which is defined in phigs.h to be -4.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u4 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u4 {
                Pint            ws_id;          /* workstation identifier */
                Ppoint          point;          /* a drawable point -- not DC */
                Pfloat          ap_size;        /* aperture size, half-width in DC
                                                    units */
                Ppath_order     order;
                Pint            depth;          /* depth of path to return */
                Pint            pet;
                Pecho_switch    echo_switch;
                Plimit3         echo_volume;
                Pfilter         filter;         /* detectability filter */
        } escape_in_u4;                 /* drawable point to pick */
} Pescape_in_data;                      /* drawable point to pick */
```

ws_id Indicates the workstation to use. The structures posted to this workstation and the current workstation state will be used to attempt the mapping.

point A point, in X drawable units, that indicates the location of the pick aperture. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;              /* x coordinate */
        Pfloat      y;              /* y coordinate */
} Ppoint;
```

ap_size The size of the pick aperture, specified as the half-width of the aperture box in DC units.

order The order of the returned pick path. The order can be either PORDER_TOP_FIRST or PORDER_BOTTOM_FIRST. If it is PORDER_TOP_FIRST, then the pick path is returned in traversal order. If it is PORDER_BOTTOM_FIRST, then the pick path is returned in the reverse of traversal order. The order and depth parameters specify the portion of the complete pick path, if any, that is returned.

depth The portion of the pick path to return. If the order parameter is PORDER_TOP_FIRST, then the first *depth* elements are returned. If the order parameter is PORDER_BOTTOM_FIRST, then the first *depth* elements in the inversely ordered pick path are returned.

pet Indicates the echo method to use for picked primitives.

echo_switch
Indicates whether or not picked primitives should be echoed. Pecho_switch is an enumerated type defined in phigs.h as:

```
typedef enum {
        PSWITCH_NO_ECHO,
        PSWITCH_ECHO
    } Pecho_switch;
```

echo_volume
The echo volume. Plimit3 is defined in phigs.h as follows:

```
typedef struct {
        Pfloat    x_min;    /* x minimum */
        Pfloat    x_max;    /* x maximum */
        Pfloat    y_min;    /* y minimum */
        Pfloat    y_max;    /* y maximum */
        Pfloat    z_min;    /* z minimum */
        Pfloat    z_max;    /* z maximum */
} Plimit3;
```

filter The detectability filter to use for the pick. Pfilter is defined in phigs.h as follows:

```
typedef struct {
        Pint_list incl_set;   /* inclusion set */
        Pint_list excl_set;   /* exclusion set */
} Pfilter;
```

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to ensure that there is sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the Pescape_out_u4 member used by this escape function. PHIGS copies the pick status and pick path, if any, to this member of the Pescape_out_data union. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u4{
                Pin_status      status;
                Ppick_path      pick;
        } escape_out_u4;
} Pescape_out_data;
```

status contains the status of the pick.

PIN_STATUS_OK indicates that the pick was successful and a pick path is returned in the *pick* field.

PIN_STATUS_NO_IN indicates that a pick was attempted, but no primitive was selected.

pick contains the pick path if the status is PIN_STATUS_OK. Ppick_path is defined in phigs.h as:

```
typedef struct {
        Pint                depth;        /* pick path_list depth */
        Ppick_path_elem     *path_list;   /* pick path */
} Ppick_path;
```

Ppick_path_elem is defined in phigs.h as:

```
typedef struct {
        Pint      struct_id;      /* structure identifier */
        Pint      pick_id;        /* hierarchical pick identifier */
        Pint      elem_pos;       /* element sequence number */
} Ppick_path_elem;
```

The application must allocate the memory for the pick path.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCDRPTTOPICK, which is defined in phigs77.h as -4.

LIDR The dimension of IDR (returned by PACK DATA RECORD (3P)).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD (3P) when called with the following parameters:

IL The number of integers, equal to 7, plus the number of structures in the pick filter inclusion and exclusion sets.

IA Contains 6 integer values:
Element 1 is the workstation identifier

Element 2 is the order of the returned pick path. The order can be either PPOTOP, in which case the pick path will be returned in traversal order, or PPOBOT, in which case the pick path will be returned in the reverse of traversal order.

Element 3 is the depth of the pick path to return.

Element 4 is the echo method to use for picked primitives.

Element 5 indicates whether or not picked primitives should be echoed and can be either PECHO or PNECHO.

Element 6 is the number of values in the pick filter inclusion set.

Element 7 is the number of values in the pick filter exclusion set.

The next, possibly empty, set of elements is the list of values in the pick filter inclusion set, equal in number to the value of element 6.

The next, possibly empty, set of elements is the list of values in the pick filter exclusion set, equal in number to the value of element 7.

RL The number of real values = 9.

RA Contains a list of floating point values.
Elements 1 and 2 are the x and y coordinates, respectively, in X drawable coordinates of the location of the pick aperture. Element 3 is the half-width size of the pick aperture, in DC units.

Elements 4 and 5 are the minimum and maximum x values of the echo volume.

Elements 6 and 7 are the minimum and maximum y values of the echo volume.

Elements 8 and 9 are the minimum and maximum z values of the echo volume.

SL The number of strings = 0.

FORTRAN Output Parameters

MLODR
The maximum length of the output data record, ODR.

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD (3P), the following fields should be returned:

IL The number of integers = 2 + (3 × the depth of the resulting pick path)

IA Contains IL integer values:
Element 1 is the status of the pick. A value of POK indicates that the pick was successful and a pick path is returned; a value of PNPICK indicates that no primitive met the pick criteria.

Element 2 indicates the depth of the resulting pick path. This value is 0 if the pick was not successful.

If the pick is successful, Elements IA(i+3), IA(i+4), and IA(i+5), where *i* ranges from 0 to the pick depth minus 1, contain the structure identifier, pick identifier, and element sequence number for each of the elements in the pick path.

SL The number of strings = 0.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

054 Ignoring function, the specified workstation is not open

254 Ignoring function, invalid echo area / volume; XMIN >= XMAX, YMIN >= YMAX, or ZMIN > ZMAX

255 Ignoring function, one of the echo area / volume boundary points is outside the range of the device

900 Storage overflow has occurred in PHIGS

SEE ALSO

ESCAPE (3P)
INITIALIZE PICK 3 (3P)
CREATE STORE (3P)
DELETE STORE (3P)

NAME

ESCAPE -5 – map a list of X-Drawable points to PHIGS World Coordinate (WC) points

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

```
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

```
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -5 attempts to map a list of X-Drawable coordinate points to a list of PHIGS World Coordinate (WC) points using the views of the specified workstation. This provides similar functionality to a PHIGS stroke device, but uses points provided by the application rather than directly by the operator. The method used to map the points is the same one used by PHIGS stroke devices.

This escape is not associated with any PHIGS input device and can be used without initializing one. It allows the application to use the standard X Event mechanism to implement its own stroke device.

C Input Parameters

func_id The function identifier for this escape is PUESC_DRAWABLE_POINTS_TO_WC, which is defined in phigs.h to be -5.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u5 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u5 {
                Pint            ws_id;          /* workstation identifier */
                Ppoint_list3    points;         /* x,y are drawable coords,
                                                   z is DC*/
        } escape_in_u5;                 /* drawable points to WC */
} Pescape_in_data;
```

ws_id Indicates the workstation to use. The views on this workstation are used in the conversion to World Coordinates (WC) mapping.

points The list of points to map. The x and y coordinates are drawable coordinate values, as typically provided by X events. The Z coordinate is a PHIGS DC Z value. Ppoint_list3 is defined in phigs.h as follows:

```
struct {
        Pint        num_points;     /* number of Ppoint3s in the
                                       list */
        Ppoint3     *points;        /* list of points */
} Ppoint_list3;
```

Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;              /* x coordinate */
        Pfloat      y;              /* y coordinate */
        Pfloat      z;              /* z coordinate */
} Ppoint3;
```

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the Pescape_out_u5 member used by this escape function. PHIGS copies the converted points, if any, to this member of the Pescape_out_data union. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u5 {
                Pint            view_index;
                Ppoint_list3    points;
        } escape_out_u5;
} Pescape_out_data;
```

view_index
The index of the highest priority view that contains all the points and is used to transform the points to World Coordinates (WC). The view index is meaningful only if the number of points returned is greater than zero.

points The WC points. The points array in this structure must be allocated prior to calling this function and must be large enough to hold at least the number of points specified in the input parameters. If the input points do not all lie within one of the workstation's current view representations, then the number of points returned is 0.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCDRPTTOWC, which is defined in phigs77.h to be -5.

LIDR The dimension of IDR (returned by PACK DATA RECORD (3P)).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD (3P) when called with the following parameters:

IL The number of integers = 2.

IA Contains 2 integer values:
Element 1 is the workstation identifier.

Element 2 is the number of X-drawable points to be mapped to PHIGS WC coordinates.

RL The number of real values = 3 ∗ IA(2)

RA Contains a list of floating point values.
RA(1) through RA(IA(2)) are the x values of the input points, in X-drawable coordinate space.

RA(IA(2)+1) through RA(2∗IA(2)) are the y values of the input points, in X-drawable coordinate space.

RA(IA(2)+1) through RA(2∗IA(2)) are the z values of the input points, in DC (Device Coordinate) space.

SL The number of strings = 0.

FORTRAN Output Parameters

MLODR
The maximum length of the output data record, ODR.

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD (3P), the following fields should be returned:

IL The number of integers = 2.

IA Contains 2 integer values:
Element 1 is the index of the highest priority view that contains all the points and is used to transform the points to World Coordinates (WC).

Element 2 is the number of WC points that have been returned. This will be the same as IA(2) in the input data record, IDR.

RL The number of real values = 3 ∗ IA(2)

RA Contains a list of floating point values.
RA(1) through RA(IA(2)) are the x coordinates of the transformed points in WC.

RA(IA(2)+1) through RA(2∗IA(2)) are the y coordinates of the transformed points in WC.

RA(IA(2)+1) through RA(2∗IA(2)) are the z coordinates of the the transformed points in WC.

SL The number of strings = 0.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)

054 Ignoring function, the specified workstation is not open

900 Storage overflow has occurred in PHIGS

SEE ALSO

ESCAPE (3P)
INITIALIZE STROKE 3 (3P)

NAME

ESCAPE -6 – redraw a list of a PHIGS workstation window regions

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      not used
Pescape_out_data  **out;      OUT not used
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -6 redraws the specified list of regions of a PHIGS workstation window without updating the workstation state list. This function allows applications to redraw the workstation window in response to X exposure events, without updating the state of the workstation state list. When this function is called the workstation redraws the specified regions.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_REDRAW_REGIONS, which is defined in phigs.h to be -6.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u6 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u6 {
                Pint          ws_id;          /* workstation identifier */
                Pint          num_regions;    /* number of regions */
                XRectangle    *regions;       /* regions to redraw */
        } escape_in_u6;              /* redraw regions */
} Pescape_in_data;
```

ws_id Indicates the workstation to redraw.

num_regions
The number of regions specified.

regions The regions of the workstation window to redraw.

store This parameter is not used in this escape function.

C Output Parameters

out This parameter is not used by this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCREDRAWREG, which is defined in phigs77.h to be -6.

LIDR The dimension of IDR (returned by PACK DATA RECORD (3P)).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD (3P) when called with the following parameters:

IL The number of integers = 2 + (4 ∗ the number of regions)

IA An array of integer values:
IA(1) is the workstation identifier.

IA(2) is the number of regions to be redrawn.

IA(3) through IA(IA(2)+2) are the x coordinates of each of the regions to be redrawn.

IA(IA(2)+3) through IA(2∗IA(2)+2) are the y coordinates of each of the regions to be redrawn.

IA(2∗IA(2)+3) through IA(3∗IA(2)+2) are the widths of each of the regions to be redrawn.

IA(3∗IA(2)+3) through IA(4∗IA(2)+2) are the heights of each of the regions to be redrawn.

RL The number of real values = 0

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output parameter for this escape.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)

054 Ignoring function, the specified workstation is not open

SEE ALSO

ESCAPE (3P)

NAME

ESCAPE -7 – synchronize with a workstation

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint               func_id;    escape function identifier
Pescape_in_data    *in;        input data for the function
Pstore             store;      not used
Pescape_out_data   **out;      OUT not used
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -7 synchronizes with a workstation. This function calls XSync() on all connections PHIGS is using to communicate with the server associated with the specified workstation. This has the effect of flushing X connections and waiting for requests sent to the workstation to be processed.

C Input Parameters

func_id The function identifier for this escape is PUESC_WS_SYNCH, which is defined in phigs.h to be -7.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u7 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u7 {
                Pint        ws_id;          /* workstation identifier */
        } escape_in_u7;                     /* workstation synch */
} Pescape_in_data;
```

ws_id Indicates the workstation to synchronize with.

store This parameter is not used in this escape function.

C Output Parameters

out This parameter is not used by this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCWSSYNC, which is defined in phigs77.h to be -7.

LIDR The dimension of IDR (returned by PACK DATA RECORD (3P)).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD (3P) when called with the following parameters:

IL	The number of integers = 1.
IA	Contains one integer, the workstation identifier of the workstation to be synchronized with.
RL	The number of real values = 0.
SL	The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)

054 Ignoring function, the specified workstation is not open

SEE ALSO

ESCAPE (3P)

NAME

ESCAPE -8 – raster resize

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*not used*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, PWSOP *, *)

DESCRIPTION

Purpose

ESCAPE -8 allows an application to notify PHIGS that a resize event has occurred on a drawable. It signals to the application that a window resize has occurred and maintains the correct aspect ratio.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_RASTER_RESIZE, which is defined in phigs.h to be -8.

in A pointer to a Pescape_in_data union containing the input data record, escape_in_u8. It is defined as:

```
typedef union {
        struct Pescape_in_u8 {
                Pint    ws_id;   /* workstation identifier */
        } escape_in_u8;          /* raster resize */
} Pescape_in_data;
```

store Unused

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCRASRESIZE, which is defined in phigs77.h to be -8.

LIDR The dimension of IDR (returned by the PACK DATA RECORD subroutine).

IDR(LIDR)
An ESCAPE data record packed by the PACK DATA RECORD subroutine containing the input data record required by the ESCAPE function specified in *FCTID*.

IL	The number of integers = 1.
IA	Contains one integer value, specifying the workstation id.
RL	The number of real values = 0.
SL	The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this ESCAPE.

Execution

SunPHIGS requires notification that a resize event has occurred on the drawable on which a workstation lies. For X Tool workstation types this does not present a problem since SunPHIGS performs its own event processing internally. However, for X-Drawable workstation types, the application performs all necessary event processing. Thus, when using X-Drawable workstations, SunPHIGS requires notification that a resize event has occurred on the drawable. If ESCAPE -8 is not called when the drawable is resized, the workstation will be scaled improperly within the drawable.

This escape can signal the application that a window resize has occurred. However, it does not perform the resize in SunPHIGS, which uses fixed device coordinates as the default. The application can resize a window if:

1. DC limits are changed from the default (FIXED) by using WORKSTATION TYPE SET and the attribute PHIGS_DC_MODEL with the value PHIGS_DC_LIMITS_ADJUST_TO_WINDOW.
2. PUESC_RASTER RESIZE is called.
3. The new window size (height and width) from the event report is put into a Plimit data structure as the *x_max* and *y_max* elements (X_MIN and Y_MIN should be zero) and passed to SET WORKSTATION VIEWPORT or SET WORKSTATION VIEWPORT 3 with the workstation ID.

The purpose of ESCAPE -8 is to maintain the correct aspect ratio. If SET WORKSTATION VIEWPORT is called without it, then the resulting PHIGS output is "rubber sheeted" into any non-square window.

Note: This ESCAPE is a no-op for anything but X-Drawable workstations.

The *escape* input data record contains the workstation id of the workstation which has been resized.

SEE ALSO

ESCAPE (3P)

NAME

ESCAPE -9 – Set X Drawable Region WS Rect

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint             func_id;    escape function identifier
Pescape_in_data  *in;        input data for the function
Pstore           store;      not used
Pescape_out_data **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER        FCTID         function identification
INTEGER        LIDR          dimension of input data record array
CHARACTER*80   IDR(LIDR)     input data record
INTEGER        MLODR         maximum length of output data record
INTEGER        LODR          OUT number of array elements used in ODR
CHARACTER*80   ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP *, *)

DESCRIPTION

Purpose

ESCAPE -9 allows an application to specify a new location on an X drawable in which to place an already-opened X drawable region workstation. For example, an application may want to change the way in which its X drawable region WSs are laid out after a RESIZE event has occurred on the drawable.

Note: This function is supported only for X drawable region workstations.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_REGION_WS_RECT, which is defined in phigs.h to be -9.

in A pointer to a Pescape_in_data union containing the input data record, escape_in_u9. It is defined as:
sp .4

```
typedef union {
        struct Pescape_in_u9 {
                Pint    ws_id;    /* workstation identifier */
                Prect   rect;     /* rectangle on drawable to place the
                                     region WS in — origin is at lower
                                     left, coords are in pixels */
        } escape_in_u9;           /* set X-drawable region WS rect */
} Pescape_in_data;
```

store Unused.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETREGRECT, which is defined in phigs77.h to be -9.

LIDR The dimension of IDR (returned by the PACK DATA RECORD subroutine).

IDR(LIDR)
An ESCAPE data record packed by the PACK DATA RECORD subroutine containing the input data record required by the ESCAPE function specified in FCTID.

IL The number of integers = 1.

IA Contains one integer value, specifying the workstation id.

RL The number of real values = 4.

RA Contains 4 real values:
Element 1 is the lower left X coordinate of the rectangle.
Element 2 is the lower left Y coordinate of the rectangle.
Element 3 is the upper right X coordinate of the rectangle.
Element 4 is the upper right Y coordinate of the rectangle.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this ESCAPE.

Execution

If the WS is not transparent, the WS is cleared in its old location. (If the Z buffer exits, it is cleared, as well). If the WS is transparent, no clearing of the old location takes place. The WS is then updated internally to reflect its new location, and the state of visual representation in its workstation state list is set to PVISUAL_ST_DEFER (deferred). No retraversal is caused by this escape; the application is responsible for updating the display. (See *REDRAW ALL STRUCTURES (3P)*, *UPDATE WORKSTATION (3P)*, and *ESCAPE -18 (3P)*) for more information.

If the specified rectangle is not completely on the drawable, an error is returned and the function is ignored.

Usage

This escape can be used at any time on an open X drawable region WS. It is intended, however, to be called by an application after a resize of the drawable has occurred.

After receiving a resize event from a drawable, the application must inform each SunPHIGS workstation that the drawable has changed size. For non-X drawable region WSs, ESCAPE -8 (Raster Resize) should be used. For X drawable region WSs, either ESCAPE -8 (Raster Resize) or ESCAPE -9 (Set Drawable Region WS Rect) can be used. ESCAPE -9 must be called if the application is to position the region workstation differently on the drawable as a result of the resize event.

Note: One of these two escapes must be called for each drawable region workstation after a resize event.

A possible sequence of events might be:

1. The application receives a resize event.
2. It recomputes the locations and sizes of its X drawable region WSs.
3. For each X drawable region WS, the application calls ESCAPE -9 with its new rectangle. (At this point, all the region WSs are in the proper places on the drawable, but their visual states are deferred)
4. For each WS, call UPDATE WORKSTATION, REDRAW ALL STRUCTURES, or ESCAPE -18 (Update a Set of Region WSs on a Drawable).

ERRORS

03	Ignoring function, function requires state (PHOP, WSOP, *, *)
54	Ignoring function, the specified workstation is not open
–168	Ignoring function, drawable region workstation is invalid

SEE ALSO

ESCAPE (3P)
INTRO DRAWABLE REGION (7P)

NAME

ESCAPE -10 – query for character set facilities

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -10 returns a list of character sets and their widths supported by a workstation type; *width* is the number of bytes needed to specify a character in a character set.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_CHAR_SET_FACILITIES, which is defined in phigs.h to be -10.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u10. It is defined in phigs.h as follows:

```
typedef union {
        struct Pescape_in_u10 {
                Pint    ws_type;    /* workstation type */
        } escape_in_u10;
} Pescape_in_data;
```

ws_type Workstation type.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area such that there is sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to a Pescape_out_data union, which will be allocated from *store* and will contain the output data record Pescape_out_u10. It is defined in phigs.h as follows:

```
typedef union {
        struct Pescape_out_u10 {
                Pint                err_ind;        /* error indicator */
                Phigs_char_set_facs facs;           /* list of available
                                                       character sets */
        } escape_out_u10;
} Pescape_out_data;
```

The fields in this record correspond in their usage to the usual output parameters of a facilities inquiry (see INQUIRE TEXT FACILITIES).

err_ind An integer in which the system returns the error number of any error detected by this function.

facs A Phigs_char_set_facs data structure in which the system returns a listing of the character set facilities available on this type of workstation. Phigs_char_set_facs is defined in phigs.h as:

```
typedef struct {
        Pint            num_char_sets;  /* number of character
                                           sets */
        Phigs_char_set  *char_sets;     /* list of character sets */
} Phigs_char_set_facs;
```

Phigs_char_set is defined in phigs.h as:

```
typedef struct {
        Pint            char_set;       /* index of character set */
        Pint            width;          /* width of character set */
} Phigs_char_set;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQCSFAC, which is defined in phigs77.h to be -10.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with the following parameters:

IL The number of integers = 2.

IA Contains two integer values:

Element 1 is the workstation type.

Element 2 is the index of the character set list element desired. A value of

0 may be used to determine the size of the list in PHIGS.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 4.

IA Contains four integer values.

Element 1 is the error indicator.

Element 2 is the length of the list of character sets.

Element 3 is the Nth element of the character set index list, as specified by element 2 in the input data record.

Element 4 is the Nth element of the character set width list, as specified by element 2 in the input data record.

RL The number of real values = 0.

SL The number of strings = 0.

Execution

The *Inquire Character Set Facilities* escape returns a list of character sets and their *widths* supported by a workstation. *width* is the number of bytes needed to specify a character in a character set. For more information on character sets, see INTRO INTERNATIONALIZATION (7P).

ERRORS

002	Ignoring function, function requires state (PHOP, *, *, *)
052	Ignoring function, workstation type not recognized by the implementation
062	Ignoring function, this information is not available for this MO workstation type
351	Ignoring function, one of the fields within the escape data record is in error
900	Storage overflow has occurred in PHIGS

SEE ALSO

INTRO INTERNATIONALIZATION (7P)

NAME

ESCAPE -11 – query for fonts available for character set

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID         function identification
INTEGER         LIDR          dimension of input data record array
CHARACTER*80    IDR(LIDR)     input data record
INTEGER         MLODR         maximum length of output data record
INTEGER         LODR          OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -11 returns a list of fonts and precisions supported for the specified character set on the specified workstation type.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_FONTS_FOR_CHAR_SET which is defined in phigs.h to be -11.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u11. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u11 {
                Pint    ws_type;    /* workstation type */
                Pint    char_set;   /* character set to get font
                                       information for */
        } escape_in_u11;
} Pescape_in_data;
```

ws_type Workstation type.

char_set
The character set for which the the font information is desired.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area such that there is sufficient memory for the specific inquiry.

The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to a Pescape_out_data union, which will be allocated from *store* and will contain the output data record Pescape_out_u11. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u11 {
                Pint              err_ind;        /* error indicator */
                Phigs_font_facs   facs;           /* text facilities */
        } escape_out_u11;
} Pescape_out_data;
```

err_ind An integer in which the system returns the error number of any error detected by this function.

facs A Phigs_font_facs data structure in which the system returns a listing of the text fonts and precisions available for the specified character set on this type of workstation. Phigs_font_facs is defined in phigs.h as:

```
typedef struct {
        Pint              num_font_precs;   /* number of fonts and
                                               precisions */
        Ptext_font_prec   *font_precs;      /* list of fonts and
                                               precisions */
} Phigs_font_facs;
```

Ptext_prec is defined in phigs.h as:

```
typedef struct {
        Pint         font;    /* text font */
        Ptext_prec   prec;    /* text precision */
} Ptext_font_prec;
```

Ptext_font_prec is defined in phigs.h as:

```
typedef enum {
        PPREC_STRING,
        PPREC_CHAR,
        PPREC_STROKE,
} Ptext_prec;
```

The application must allocate memory for the *facs.font_precs* list.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQFTCS, which is defined in phigs77.h to be -11.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD called with:

IL The number of integers = 3.

IA Contains three integer values:

Element 1 is the workstation type.

Element 2 is the character set for which the facilities information is desired.

Element 3 is the index of the font and precision list element desired. A value of 0 may be used to inquire the length of the list in PHIGS.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD the following fields should be returned:

IL The number of integers = 4.

IA Contains four integer values:

Element 1 is the error indicator.

Element 2 is the length of the list of fonts and precisions.

Element 3 is the Nth element of the font list, as specified by element 2 in the input data record.

Element 4 is the Nth element of the precision list.

RL The number of real values = 0.

SL The number of strings = 0.

Execution

The *Inquire Extended Fonts* escape returns a list of the fonts and precisions available for the specified character set on the specified workstation type. See INTRO INTERNATIONALIZATION (7P) for information on character sets and fonts.

ERRORS

–155	Specified character set is invalid.
002	Ignoring function, function requires state (PHOP, *, *, *)
052	Ignoring function, workstation type not recognized by the implementation

062	Ignoring function, this information is not available for this MO workstation type
351	Ignoring function, one of the fields within the escape data record is in error
900	Storage overflow has occurred in PHIGS

SEE ALSO

INTRO INTERNATIONALIZATION (7P)

NAME

ESCAPE -12 – query for extended text extent

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -12 performs the same function as INQUIRE TEXT EXTENT using an I-String instead of a regular ASCII string.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_EXTENDED_TEXT_EXTENT, which is defined in phigs.h to be -12.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u12. It is defined as follows:

```
typedef union {
        struct Pescape_in_u12 {
                Pint              ws_type;    /* workstation type */
                Pint              font;       /* text font */
                Pfloat            expan;      /* char expansion factor */
                Pfloat            space;      /* char spacing */
                Pfloat            ht;         /* char height */
                Ptext_path        path;       /* text path */
                Phor_text_align   hor;        /* horizontal alignment */
                Pvert_text_align  vert;       /* vertical alignment */
                Phigs_i_string    i_string;   /* text I-String */
        } escape_in_u12;
} Pescape_in_u12;
```

The fields in this record correspond to the input parameters of INQUIRE TEXT EXTENT with the ASCII string being replace by an I-String. See INTRO INTERNATIONALIZATION for a description of I-Strings.

ws_type	The workstation type to use to resolve the font.
font	The *text font,* specified as an index to a workstation's table of available fonts. The font is used for the entire I-String. See SET TEXT FONT for a list of available fonts for each character set.
expan	The character expansion factor. See SET CHARACTER EXPANSION FACTOR for the valid values.
space	The character spacing. See SET CHARACTER SPACING for the valid values.
ht	The character height. See SET CHARACTER HEIGHT for the valid values.
path	The text path. See SET TEXT PATH for the valid values.
hor	The horizontal alignment. See SET TEXT ALIGNMENT for the valid values.
vert	The vertical alignment. See SET TEXT ALIGNMENT for the valid values.
i_string	contains a list of mono-encodings. Phigs_i_string is defined in phigs.h as follows:

```
typedef struct {
        Pint                  num_encodings;      /* number of
                                                     encodings */
        Phigs_mono_encoding   *mono_encodings;    /* list of
                                                     encodings */
} Phigs_i_string;
```

Each mono-encoding (substring) contains a character set, an encoding state, and a NULL-terminated string. Phigs_mono_encoding is defined in phigs.h as follows:

```
typedef struct {
        Pint    char_set;           /* character set */
        char    encoding_state;     /* encoding state */
        char    *string;            /* character string */
} Phigs_mono_encoding;
```

Valid values for char_set are defined in phigs.h as follows:

PCS_ASCII	0
PCS_GREEK	–1
PCS_SYMBOL	–2
PCS_CARTOGRAPHIC	–3
PCS_KANJI	–4

encoding_state
Is used to encode within a mono-encoding a character encoding

method (for example, EUC, CT). This parameter is not actually used by the SunPHIGS library. However, it can be used in conjunction with the INQUIRE ELEMENT CONTENT function to restore mono-encoded text to its original encoding method.

string A pointer to the NULL-terminated character string.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameter

out The address of a a pointer to a Pescape_out_data union, which will be allocated from *store* and will contain the output data record, Pescape_out_u12. It is defined as follows:

```
typedef union {
        struct Pescape_out_u12 {
                Pint      err_ind;      /* error indicator */
                Prect     rect;         /* extent rectangle */
                Ppoint    offset;       /* concatenation offset */
        } escape_out_u12;
} Pescape_out_data;
```

The fields in this record correspond to the output parameters to INQUIRE TEXT EXTENT.

err_ind The error indicator.

rect A Prect structure in which to return the text extent. Prect is defined in phigs.h as:

```
typedef struct {
        Ppoint    p;    /* lower left */
        Ppoint    q;    /* upper right */
} Prect;
```

Ppoint is defined in phigs.h as:

```
typedef struct {
        Pfloat    x;    /* x coordinate */
        Pfloat    y;    /* y coordinate */
} Ppoint;
```

offset A Ppoint structure in which to return the concatenation offset.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQXTXEXT, which is defined in phigs77.h to be -12.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with:

IL The number of integers = S + 2 × SL.

IA Contains an array of integer values.

Element 1 is the workstation type.

Elements 2 is the font index. See SET TEXT FONT for the valid values.

Element 3 is the text path. See SET TEXT PATH for the valid values.

Element 4 is the horizontal alignment. See SET TEXT ALIGNMENT for the valid values.

Element 5 is the vertical alignment. See SET TEXT ALIGNMENT for the valid values.

The next SL integers are the character sets for each encoding.

The next SL integers are the encoding states for each encoding.

RL The number of real values = 3.

RA Contains three real values.

Element 1 is the character expansion factor. See SET CHARACTER EXPANSION FACTOR for the valid values.

Element 2 is the character spacing. See SET CHARACTER SPACING for the valid values.

Element 3 is the character height. See SET CHARACTER HEIGHT for the valid values.

SL The number of strings. This is the number of encodings in the I-String.

LSTR An array of length SL, containing the length of each string in the I-String.

STR An array of SL strings, one for each encoding.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output data record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 1.

IA Contains the error indicator.

RL The number of real values = 6.

RA	Contains six real values: Element 1 is the *x* value of the lower left corner of the extent box. Element 2 is the *y* value of the lower left corner of the extent box. Element 3 is the *x* value of the upper right corner of the extent box. Element 4 is the *y* value of the upper right corner of the extent box. Element 5 is the *x* value of the concatenation offset. Element 6 is the *y* value of the concatenation offset.
SL	The number of strings = 0.

Execution

The *Inquire Extended Text Extent* escape performs the same function as INQUIRE TEXT EXTENT using an I-String as input instead of a regular ASCII string. See INTRO INTERNATIONALIZATION (7P) for a description of I-Strings.

ERRORS

–155	Specified character set is invalid.
–156	Warning, the specified font is not available for one or more of the character sets
002	Ignoring function, function requires state (PHOP, *, *, *)
051	Ignoring function, this information is not yet available for this generic workstation type; open a workstation of this type and use the specific workstation type.
052	Ignoring function, workstation type not recognized by the implementation
059	Ignoring function, the specified workstation does not have output capability
106	Ignoring function, the specified font is not available for the requested text precision on the specified workstation
351	Ignoring function, one of the fields within the escape data record is in error

SEE ALSO

INTRO INTERNATIONALIZATION (7P)
INQUIRE TEXT EXTENT (3P)
SET TEXT FONT (3P)

NAME

ESCAPE -13 – query for whole physical window size

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -13 returns the current physical window size in pixels.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_WS_PHYS_SIZE, which is defined in phigs.h to be -13.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u13. It is defined as:

```
typedef union {
        struct Pescape_in_u13 {
                Pint    ws_id;
        } escape_in_u13;
} Pescape_in_data;
```

ws_id The workstation identifier associated with this workstation.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to a Pescape_out_data union, which will be allocated from *store* and will contain the output data record Pescape_out_u13. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u13 {
                Pint    err_ind;    /∗ error indicator ∗/
                Pint    width;      /∗ window physical width ∗/
                Pint    ht;         /∗ window physical height ∗/
        } escape_out_u13;
} Pescape_out_data;
```

err_ind An integer in which the system returns the error number of any error detected by this function.

width An integer representing the physical width of the current window in pixels.

ht An integer representing the physical height of the current window in pixels.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCPHYSSIZE, which is defined as -13 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An escape data record packed by PACK DATA RECORD and called with these arguments:

IL The number of integers = 1.

IA Contains one integer value.

Element 1 is the workstation identifier.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array ODR.

ODR(MLODR)
The output data record is stored in this array. When the output data record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 3.

IA Contains three integer values.

Element 1 is the error indicator.

Element 2 is the current window width.

	Element 3 is the current window height.
RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

The Inquire Physical Window Size escape returns the current physical window size in pixels.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
–159	Ignoring function, the requested information is not available

SEE ALSO

INQUIRE DISPLAY SPACE SIZE (3P)
INQUIRE DISPLAY SPACE SIZE 3 (3P)

NAME

ESCAPE -14 – inquire XGL window raster

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID           function identification
INTEGER         LIDR            dimension of input data record array
CHARACTER*80    IDR(LIDR)       input data record
INTEGER         MLODR           maximum length of output data record
INTEGER         LODR            OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)      OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, WSOP)

DESCRIPTION

Purpose

ESCAPE -14 acquires the XGL window raster associated with the specified workstation.

It is intended to allow the application to mix PHIGS with XGL function calls.

C Input Parameters

Applications using the C binding must create a buffer to be used by this function as memory space for storing data associated with the device state. This buffer is passed as the *store* argument.

func_id The function identifier for this escape is PUESC_INQ_XGL_RASTER, which is defined in phigs.h as -14.

in A pointer to a Pescape_in_data union, which contains the Pescape_in_u4 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u14 {
                Pint            ws_id;          /* workstation identifier */
        } escape_in_u14;
} Pescape_in_data;
```

ws_id Indicates the workstation to use.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The data record within the store buffer is accessed by the pointer pointed to by *out*.

C Output Parameter

out The address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the Pescape_out_u14 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u14 {
                Pint            err_ind;        /∗ error indicator ∗.
                void            ∗win_ras;       /∗ XGL window raster object ∗/
        } escape_out_u14;
} Pescape_out_data;
```

err_ind contains the status of the inquiry. When *err_ind* is 0, it indicates that the inquiry was successful and an XGL window raster is returned in the *win_ras* field. When *err_ind* is non-zero, the content of the *win_ras* field is undefined.

win_ras contains the XGL window raster object handle. Ppick_path is defined in xgl.h as:

Xgl_win_ras

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQXGLRAS, which is defined in phigs77.h to be -14.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with the following parameters:

IL The number of integers = 1.

IA Contains one integer, the workstation identifier of the workstation to be used.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

MLODR
The maximum length of the output data record, ODR.

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 2.

IA An array of integer values:

Element 1 is the error indicator, containing the status of the inquiry; a 0 value indicates that the inquiry was successful.

Element 2 contains the XGL window raster object handle.

RL	The number of real values = 0.
SL	The number of strings = 0.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)
054	Ignoring function, the specified workstation is not open
254	Ignoring function, invalid echo area / volume; XMIN >= XMAX, YMIN >= YMAX, or ZMIN > ZMAX
255	Ignoring function, one of the echo area / volume boundary points is outside the range of the device

SEE ALSO

ESCAPE (3P)
CREATE STORE (3P)
DELETE STORE (3P)

NAME

ESCAPE -15 – specify the type of archive

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*not used*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -15 allows the selection of the format of archive files.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_ARCHIVE_TYPE, defined in phigs.h to have the value -15.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u15. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u15 {
                Phigs_ar_mode  ar_mode;
        } escape_in_u15;
} Pescape_in_data;
```

The valid values for *ar_mode* are defined in phigs.h as:

0	PHIGS_AR_CLEAR_TEXT	*Write archive in clear text format*
1	PHIGS_AR_PEX	*Write archive in PEX binary format*

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier for this escape is PUESCSETARMODE, defined in phigs77.h to have the value -15.

LIDR The dimension of the input data record array, IDR, (returned by PACK DATA RECORD). ERROR HANDLING will be invoked if the specified dimension is less than zero.

IDR An escape input data record packed by PACK DATA RECORD containing the input data for this escape. The arguments to PACK DATA RECORD should be as follows:

IL The number of integers = 1.

IA Contains one integer value, specifying the archive mode. The valid values are:

PARMODECLR	*Write archive in clear text format*
PARMODEPEX	*Write archive in PEX binary format*

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

The *Archive Mode* escape provides control over the format of archive files written. This escape should be called before the archive file is opened. Two formats of archive files are supported: clear text format and a binary PEX format. The PEX format is supported for users who want compact archives over standard conformance.

ERRORS

002 Ignoring function, function requires state (PHOP, *, *, *)

351 Ignoring function, one of the fields within the escape data record is in error

SEE ALSO

CLOSE ARCHIVE FILE (3P)

NAME

ESCAPE -16 – query for workstation type support

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID         function identification
INTEGER         LIDR          dimension of input data record array
CHARACTER*80    IDR(LIDR)     input data record
INTEGER         MLODR         maximum length of output data record
INTEGER         LODR          OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP,WSOP,*,*)

DESCRIPTION

Purpose

ESCAPE -16 retrieves values describing whether a workstation supports certain features in hardware, software, or not at all. Since SunPHIGS provides many features in software that are not present in hardware, this information allows a performance-sensitive application to avoid use of those features.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_WS_TYPE_SUPPORT, defined as the value –16 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u16. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u16 {
                Pint  ws_type;  /* workstation type identifier */
        } escape_in_u16;
} Pescape_in_data;
```

ws_type The workstation type returned from INQUIRE WORKSTATION CONNECTION AND TYPE.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record Pescape_out_u16. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u16 {
                Pint                    err_ind;        /∗ error indicator ∗/
                Phigs_ws_type_support   ws_type_sup;    /∗ workstation type
                                                           support ∗/
        } Pescape_out_u16;
} Pescape_out_data;
```

err_ind A pointer to the location in which to store the error number of an error this function detects.

ws_type_sup
A structure *Phigs_ws_type_support*, containing three structures on the input, input/output, and output workstations. *Phigs_ws_type_support* is defined in phigs.h as:

```
typedef struct {
        Phigs_ws_input_support   in_sup;      /∗ input
                                                 workstation ∗/
        Phigs_ws_in_out_support  in_out_sup;  /∗ input/output
                                                 workstation ∗/
        Phigs_ws_output_support  out_sup;     /∗ output
                                                 workstation ∗/
} Phigs_ws_type_support;
```

The *in_sup* component specifies the support of input. Currently, this field is reserved for future use. Content of this component is subject to change. The structure *Phigs_ws_input_support* is specified as:

```
typedef struct {
        Phigs_support_level  reserved;
} Phigs_ws_input_support;
```

The *Phigs_support_level* is specified as:

```
typedef enum {
        PHIGS_NOT_SUPPORTED,  /∗ not supported ∗/
        PHIGS_SOFTWARE,       /∗ software or inefficient
                                 support ∗/
        PHIGS_ACCELERATED     /∗ hardware or microcode
```

```
                                                support */
    PHIGS_SERVER_DEP         /* server dependent */
} Phigs_support_level;
```

The *in_out_sup* component specifies input / output workstation support. Content of this component is subject to future change. The *Phigs_ws_in_out_support* structure is as follows:

```
typedef struct {
    Phigs_support_level  picking;  /* picking support */
} Phigs_ws_in_out_support;
```

The *out_sup* component specifies the output support. Content of this component is subject to future change. The structure of *Phigs_ws_out_support* is as follows:

```
typedef struct {
    Phigs_support_level  double_buffer;      /* hardware
                                                double */
                                                buffering */
    Phigs_support_level  zbuff_hlhsr;        /* z-buffer
                                                hlhsr */
    Phigs_support_level  indexed_colr;       /* indexed
                                                colour */
    Phigs_support_level  true_colr;          /* true colour */
    Phigs_support_level  depth_cueing;       /* depth cueing */
    Phigs_support_level  lighting;           /* lighting */
    Phigs_support_level  shading;            /* shading */
    Phigs_support_level  anti_aliasing;      /* anti-aliasing */
    Phigs_rndr_support   rndr_supp           /* renderer
                                                support */
    Pchar                device_name[80];    /*graphics device
                                                name */
} Phigs_ws_output_support;
```

Anti-aliasing support includes both GSE –7 (*Set Stroke Anti-Aliasing Flag*) and ESCAPE –40 (*Set Workstation Mode for Anti-aliasing*).

rndr_sup describes the type of protocol PHIGS uses to communicate with the graphics device for the specified workstation. *Phigs_rndr_support* is defined in phigs.h as:

```
typedef enum {
    PHIGS_XGL_DGA   = 0,     /* Direct Graphics Access (TM) */
    PHIGS_XGL_XLIB  = 1,
    PHIGS_PEX_WKS   = 2,     /* PEX workstation subset */
    PHIGS_DLX_DGA   = 3      /* Display List
```

```
                                                Acceleration / DGA */
            } Phigs_rndr_support;
```

The PHIGS_XGL_DGA value is returned when PHIGS communicates directly with a local frame buffer or accelerator, such as GX or GS, using Direct Graphics Access. PHIGS_XGL_XLIB is returned when PHIGS renders by communicating with an X server using the X protocol. PHIGS_PEX_WKS means that PHIGS communicates with the server using the PEX protocol workstation subset. PHIGS_DLX_DGA is returned if rendering is done directly to a local display list accelerator, such as the GT, using Direct Graphics Access.

device_name is a string describing the specific graphics device that is used for rendering. The currently defined names are:

Sun:color-8	An 8-bit unaccelerated frame buffer
Sun:color-24	A 24-bit unaccelerated frame buffer
Sun:GX	The GX / cgsix
Sun:GS	A 24-bit accelerated frame buffer (the GS / cgtwelve)
Sun:GT	A 24-bit high-performance accelerated frame buffer (the GT / dlx)
XLib	Rendering is done via the X protocol
vendorname:PEX	Rendering is done by a PEX server from the named vendor

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQWSTSUP, which is defined as -16 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD and called with the following parameters:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the workstation type.

Element 2 is the specific type support to obtain. Valid values are defined in phigs77.h as:

PHIGSINOUTSUPPICKING	*Picking support*
PHIGSOUTSUPDBLBUFF	*Hardware double buffering*
PHIGSOUTSUPZBUFFHLHSR	*Z-buffer HLHSR*
PHIGSOUTSUPINDEXCOLOUR	*Indexed colour*
PHIGSOUTSUPTRUECOLOUR	*True colour*
PHIGSOUTSUPDEPTHCUEING	*Depth cueing*
PHIGSOUTSUPLIGHTING	*Lighting*
PHIGSOUTSUPSHADING	*Shading*
PHIGSOUTSUPANTIALIASING	*Anti-aliasing*

PHIGSOUTSUPRENDERER	*Renderer support type*
PHIGSOUTSUPDEVNAME	*Device name*

Anti-aliasing support includes both GSE –7 (*Set Stroke Anti-aliasing Identifier*) and ESCAPE –40 (*Set Workstation Mode for Anti-aliasing*).

RL The number of real values = 0.

SL The number of strings = 0.

MLODR The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array *ODR*.

ODR(MLODR)
The output data record is stored in this array. When the output data record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 1 or 2.

IA Contains one or two integer values (1, if the requested support type is PHIGSOUTSUPDEVNAME; 2, otherwise).

Element 1 is the error indicator.

Element 2, if present, is the support level of the obtained type. For requested support types other than PHIGSOUTSUPRENDERER, the valid values are defined in phigs77.h as:

0	PHIGSNOTSUP	*Not supported*
1	PHIGSSUPSW	*Software or inefficient support*
2	PHIGSSUPACC	*Hardware or microcode support (accelerated)*
3	PHIGSSRMVDEP	*Server dependent*

When the requested support type is PHIGSOUTSUPRENDERER, Element 2 is the renderer support for the specified workstation type. Valid values are defined in phigs77.h as:

0	PRSXGLDGA	*DGA*
1	PRSXGLXLB	*XLib*
2	PRSPEXWKS	*PEX workstation subset*
3	PRSDLXDGA	*Display list acceleration / DGA*

The value PRSXGLDGA is returned when PHIGS communicates directly with a local frame buffer or accelerator, such as GX or GS, using Direct Graphics Access. PRSXGLXLB is returned when PHIGS renders by communicating with an X server using the X protocol. PRSPEXWKS means that PHIGS communicates with the server using the PEX protocol workstation subset. PRSDLXDGA is returned when rendering is done directly to a local display list accelerator, such as the GT.

RL The number of real values = 0.

SL The number of strings = 0 or 1 (1, if the requested support type is PHIGSOUTSUPDEVNAME; 0, otherwise).

LSTR The length of the string returned in STR, if required.

STR For requested support type PHIGSOUTSUPNAME, a string containing the device name for the specified workstation. The currently-defined names are:

Sun:color-8	An 8-bit unaccelerated frame buffer
Sun:color-24	A 24-bit unaccelerated frame buffer
Sun:GX	The GX / cgsix
Sun:GS	A 24-bit accelerated frame buffer (the GS / cgtwelve)
Sun:GT	A 24-bit high-performance accelerated frame buffer (the GT / dlx)
XLib	Rendering is done via the X protocol
vendorname:PEX	Rendering is done by a PEX server from the named vendor

Execution

Currently, the workstation must be open before the inquiry can be made. In future releases, this may not be necessary. It is also necessary to make a call to INQUIRE WORKSTATION CONNECTION AND TYPE to get the specific opened workstation's type before calling this function. Otherwise, Error 051 is returned.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

051 Ignoring function, this information is not yet available for this generic workstation type; open a workstation of this type and use the specific workstation type

052 Ignoring function, workstation type is not recognized by the implementation

SEE ALSO

ESCAPE (3P)
INQUIRE WORKSTATION CONNECTION AND TYPE (3P)
WORKSTATION TYPE SET (3P)
WORKSTATION TYPE GET (3P)

Example program *WHAT_HW*

NAME

ESCAPE -17 – set visual representation state

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -17 sets the state of the visual representation in the specified workstation's state list.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_VIS_REP, defined in phigs.h to have the value -17.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u17. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u17 {
                Pint          ws_id;        /* workstation identifier */
                Pvisual_st    visual_st;    /* update state */
        } escape_in_u17;
}Pescape_in_data;
```

ws_id The workstation identifier.

visual_st
Pvisual_st is defined in phig.h as:

```
typedef enum {
        PVISUAL_ST_CORRECT,
        PVISUAL_ST_DEFER,
        PVISUAL_ST_SIMULATED
} Pvisual_st;
```

The input data record holds the workstation identifier and visual representation state.

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier for this escape is PUESCSETVISREP, which is defined as -17 in phigs77.h.

LIDR The dimension of the input data record array, *IDR*, (returned by PACK DATA RECORD). ERROR HANDLING will be invoked if the specified dimension is less than zero.

IDR(LIDR)
An escape input data record packed by PACK DATA RECORD containing the input data for this escape. The arguments to PACK DATA RECORD should be as follows:

IL The number of integers = 2.

IA Contains two integer values:

Element 1 specifies the workstation identifier.

Element 2 specifies the visual representation state. Valid values are defined in phigs77.h as:

0	PVROK	*Correct*
1	PVRDFR	*Deferred*
2	PVRSIM	*Simulated*

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

This escape sets the visual representation state for the specified workstation to the specified value.

ERRORS

002 Ignoring function, function requires state (PHOP, *, *, *)

350 Warning, the specified escape is not available on one or more workstations in this implementation. The escape will be processed by those workstations on which it is available

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
INQUIRE DISPLAY UPDATE STATE (3P)

NAME

ESCAPE -18 – update set of region workstations on a drawable

SYNOPSIS

C Syntax

void pescape (func_id, in, store, out)

```
Pint               func_id;   escape function identifier
Pescape_in_data    *in;       input data for the function
Pstore             store      handle to Store object
Pescape_out_data   **out;     OUT output data of the function
```

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

```
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -18 updates a set of region workstations on a drawable.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_UPDATE_WS_SET, defined in phigs.h to have the value -18.

in A pointer to a Pescape_in_data union, which contains the input data record Pescape_in_u18 member used by this escape function. This member is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u18 {
                XID             drawable_id;
                Pint_list       ws_list;
        } escape_in_u18;
} Pescape_in_data;              /* update set of workstations */
```

drawable_id
The drawable identifier passed to OPEN WORKSTATION as part of each region workstation's connection identifier.

ws_list The list of region workstations to be updated. Pint_list is defined in phigs.h as:

```
typedef struct {
```

```
                Pint    num_ints;     /* number of Pints in list */
                Pint    *ints;        /* list of integers */
        } Pint_list;
```

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier for this escape is PUESCUPDATESET, which is defined in phigs77.h to be -18.

LIDR The dimension of the input data record array, IDR, (returned by PACK DATA RECORD). ERROR HANDLING will be invoked if the specified dimension is less than zero.

IDR(LIDR)
An escape input data record packed by PACK DATA RECORD containing the input data for this escape. The arguments to PACK DATA RECORD should be as follows:

IL The number of integers = the number of workstations to be updated, plus 1.

IA Element 1 specifies the drawable identifier passed to OPEN WORKSTATION as part of each region workstation's connection identifier.

Elements 2 to IL specify the workstation identifiers of the workstations to be updated.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

This escape updates the specified set of region workstations on the drawable. The overall effect is as if update workstation is called for each of the listed workstations in priority order.

This escape is only supported on workstations of type X-Drawable Region.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

350 Warning, the specified escape is not available on one or more workstations in this implementation. The escape will be processed by those workstations on which it is available

351 Ignoring function, one of the fields within the escape data record is in error

SEE ALSO

ESCAPE (3P)
INTRO DRAWABLE REGION (7P)

NAME

ESCAPE -19 – add input device association

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID          function identification
INTEGER         LIDR           dimension of input data record array
CHARACTER*80    IDR(LIDR)      input data record
INTEGER         MLODR          maximum length of output data record
INTEGER         LODR           OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -19 adds an input device association.

C Input Parameters

func_id The function identifier for this escape is PUESC_ADD_DEVICE_ASSOCIATION, which is defined in phigs.h to be -19.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u19. It is defined as:

```
typedef union {
        struct Pescape_in_u19 {
                Pint                        ws_id;   /* workstation
                                                        identifier */
                Phigs_in_assoc_master       master;
                Phigs_in_assoc_slave        slave;
        } escape_in_u19;
} Pescape_in_data;
```

ws_id The *ws_id* component specifies the workstation identifier of the workstation with which the master and slave input devices are associated.

master The *master* component is a Phigs_in_assoc_master structure used to specify the master device with which a slave device is to be associated. Phigs_in_assoc_master is defined in phigs.h as follows:

```
typedef struct {
        Pin_class         dev_class;      /* device class */
        Pint              num;            /* device number */
        union {
                struct {
                          Pin_status      status;
                          Pint            choice;
                } choice;
        } value;
} Phigs_in_assoc_master;
```

class The *class* component is a Pin_class enumerated type specifying the master device class. Values for Pin_class are defined in phigs.h as follows:

PIN_NONE
PIN_LOC
PIN_STROKE
PIN_VAL
PIN_CHOICE
PIN_PICK
PIN_STRING

num The *num* component specifies the master device number.

choice If the master device class is of type CHOICE, the *choice* component of the *value* union is used to control the triggering of the slave device.

status The *status* component is an enumerated type which may take on the values PIN_STATUS_OK, or PIN_STATUS_NO_IN.

choice If it is specified as PIN_STATUS_OK, the associated slave device (specified by the *slave* component of the input data record) will be triggered when the operator selects the *choice* value specified in the *choice* component. If it is specified as PIN_STATUS_NO_IN, the associated slave device will be triggered when the operator selects any choice value.

slave The *slave* component is a Phigs_in_assoc_slave structure used to specify the slave device.

Phigs_in_assoc_slave is defined in phigs.h as follows:

```
typedef struct {
        Pin_class         dev_class;      /* device class */
        Pint              num;            /* device number */
```

} Phigs_in_assoc_slave;

class The *class* component is a Pin_class enumerated type specifying the slave device class.

num The *num* component specifies the slave device number.

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCADDASSOC, which is defined in phigs77.h to be -19.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with the following parameters:

IL The number of integers is either 6 or 7.

IA Contains either six or seven integer values (depending on the device classes involved).

Element1 is the workstation identifier of the workstation with which the master and slave input devices are associated.

Elements 2 and 3 specify the master device with which a slave device is to be associated; element 2 specifies the master device class, and element 3 the master device number. Device class values are specified in phigs77.h as follows:

PNCLAS	*None*
PLOCAT	*Locator*
PSTROK	*Stroke*
PVALUA	*Valuator*
PCHOIC	*Choice*
PPICK	*Pick*
PSTRIN	*String*

Elements 4 and 5 specify the slave device; element 4 specifies the slave device class, and element 5 the slave device number.

If the master device class is of type CHOICE, elements 6 and 7 (if needed) are used to control the triggering of the slave device. Element 6 specifies a choice status, which may be either POK (OK) or PNCHOI (NOCHOICE). If it is specified as POK, the associated slave device will be triggered when the operator selects the choice value specified in element 7. If it is specified as PNCHOI, the associated slave device will be triggered when the operator selects any choice value.

RL The number of real values = 0.

SL	The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

Selected input devices can have associated with them a set of other devices that will also generate input events when the device is triggered. An association is made with the *Add Input Device Association* escape and removed with the *Remove Input Device Association* escape. These functions accept a triggering-device / triggered-device pair. This pair indicates an additional device to trigger (the slave device) when the specified triggering device (the master device) is triggered.

For some devices, separate associations can be made with each possible value of the master device (for example, choice numbers), in which case selection of that value by the operator will trigger the devices associated with that value. For example, this allows operator selection of a specific choice value on a specified CHOICE device to also trigger one or more LOCATOR devices.

When a device and its associated devices are triggered, a set of simultaneous events are generated, one event for each device. Note that both the master and slave devices must be in event mode for this to take place.

Device associations are currently only supported for CHOICE devices as the master, and LOCATOR and PICK devices as the slave. Others can be set and removed but the association will not generate an event.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, specified workstation is not open
061	Ignoring function, specified workstation category is not INPUT or OUTIN
250	Ignoring function, specified device is not available on the specified workstation

SEE ALSO

INITIALIZE CHOICE (3P)
INITIALIZE CHOICE 3 (3P)
INITIALIZE LOCATOR (3P)
INITIALIZE LOCATOR 3 (3P)
INITIALIZE PICK (3P)
INITIALIZE PICK 3 (3P)
ESCAPE -20 (3P)

NAME

ESCAPE -20 – remove input device association

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint               func_id;    escape function identifier
Pescape_in_data    *in;        input data for the function
Pstore             store;      handle to Store object
Pescape_out_data   **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID         function identification
INTEGER          LIDR          dimension of input data record array
CHARACTER*80     IDR(LIDR)     input data record
INTEGER          MLODR         maximum length of output data record
INTEGER          LODR          OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -20 removes an input device association.

C Input Parameters

func_id The function identifier for this escape is PUESC_REMOVE_DEVICE_ASSOCIATION, which is defined in phigs.h to be -20.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u20. It is defined as:

```
typedef union {
        struct Pescape_in_u20 {
                Pint                     ws_id;   /* workstation identifier */
                Phigs_in_assoc_master    master;
                Phigs_in_assoc_slave     slave;
        } escape_in_u20;
} Pescape_in_data;
```

ws_id The *ws_id* component specifies the workstation identifier of the workstation with which the master and slave input devices are associated.

master The *master* component is a Phigs_in_assoc_master structure used to specify a master device from which a currently associated slave device is to be disassociated. Phigs_in_assoc_master is defined in phigs.h as follows:

```
typedef struct {
        Pin_class          dev_class;      /* device class */
        Pint               num;            /* device number */
        union {
                struct {
                           Pin_status     status;
                           Pint           choice;
                } choice
        } value;
} Phigs_in_assoc_master;
```

class The *class* component is a Pin_class enumerated type specifying the master device class. Values for Pin_class are defined in phigs.h as follows:

PIN_NONE
PIN_LOC
PIN_STROKE
PIN_VAL
PIN_CHOICE
PIN_PICK
PIN_STRING

num The *num* component specifies the master device number.

choice If the master device class is of type CHOICE, the *choice* component of the *value* union is used to control the triggering of the slave device.

status The *status* component is an enumerated type which may take on the values PIN_STATUS_OK, or PIN_STATUS_NO_IN.

choice If it is specified as PIN_STATUS_OK, the associated slave device (specified by the *slave* component of the input data record) triggered by the *choice* value specified in the *choice* component will be disassociated from the master device. If it is specified as PIN_STATUS_NO_IN, the associated slave device will be disassociated from the master device.

slave The *slave* component is a Phigs_in_assoc_slave structure used to specify the slave device. Phigs_in_assoc_slave is defined in phigs.h as follows:

```
typedef struct {
        Pin_class       dev_class;      /* device class */
        Pint            num;            /* device number */
} Phigs_in_assoc_slave;
```

class The *class* component is a Pin_class enumerated type specifying the slave device class.

num The *num* component specifies the slave device number.

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCREMOVEASSOC, which is defined in phigs77.h to be -20.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD when called with the following parameters:

IL The number of integers is either 6 or 7.

IA Contains either six or seven integer values (depending on the device classes involved).

Element 1 is the workstation identifier of the workstation with which the master and slave input devices are associated.

Elements 2 and 3 specify a master device from which a currently associated slave device is to be disassociated; element 2 specifies the master device class, and element 3, the master device number. Device class values are specified in phigs77.h as follows:

PNCLAS	*None*
PLOCAT	*Locator*
PSTROK	*Stroke*
PVALUA	*Valuator*
PCHOIC	*Choice*
PPICK	*Pick*
PSTRIN	*String*

Elements 4 and 5 specify the slave device; element 4 specifies the slave device class, and element 5, the slave device number.

If the master device class is of type CHOICE, elements 6 and 7 (if needed) are used to control the triggering of the slave device. Element 6 specifies a choice status, which may be either POK (OK) or PNCHOI (NOCHOICE). If it is specified as POK, the associated slave device (specified by elements 4 and 5 of the input data record) triggered by the choice value specified in element 7 will be disassociated from the master device. If it is specified as PNCHOI, the associated slave device will be disassociated from the master device.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

Selected input devices can have associated with them a set of other devices that will also generate input events when the device is triggered. An association is made with the *Add Input Device Association* escape, and removed with the *Remove Input Device Association* escape. These functions accept a triggering-device / triggered-device pair. This pair indicates an additional device to trigger (the slave device) when the specified triggering device (the master device) is triggered. For some devices, separate associations can be made with each possible value of the master device (for example, choice numbers), in which case selection of that value by the operator will trigger the devices associated with that value. For example, this allows operator selection of a specific choice value on a specified CHOICE device to also trigger one or more LOCATOR devices.

When a device and its associated devices are triggered, a set of simultaneous events are generated, one event for each device. Note that both the master and slave devices must be in event mode for this to take place.

Device associations are currently only supported for CHOICE devices as the master, and LOCATOR and PICK devices as the slave. Others can be set and removed but the association will not generate an event.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
061	Ignoring function, specified workstation category is not INPUT or OUTIN
250	Ignoring function, the specified device is not available on the specified workstation

SEE ALSO

ESCAPE -19 (3P)

NAME

ESCAPE -21 – set workstation’s cursor type and colour

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store       handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -21 allows the application to change the type and colour of the cursor used normally or during structure traversal.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_CURSOR defined in phigs.h to have the value -21.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u21. It is defined as:

```
typedef union {
        struct Pescape_in_u21 {
                Pint                  ws_id;           /* workstation id */
                Phigs_cursor_which    cursor_which;    /* cursor to change */
                Phigs_cursor_type     cursor_type;     /* cursor type */
                Prgb                  cursor_colr;     /* cursor colour */
        } escape_in_u21;
} Pescape_in_data;
```

cursor_which

The *cursor_which* member is one of the following enumerated values:

0	PHIGS_NORMAL_CURSOR	*Non-traversal cursor*
1	PHIGS_TRAVERSAL_CURSOR	*Traversal cursor*

cursor_type

The *cursor_type* member is one of the following enumerated values:

0	PHIGS_CURSOR_NONE	*No cursor*
1	PHIGS_CURSOR_DEFAULT	*Window parent's cursor*
3	PHIGS_CURSOR_CROSS	*Tracking cross*
12	PHIGS_CURSOR_NOT_SIGN	*Not sign*

cursor_colr

The cursor colour is specified as an RGB value. Prgb is defined in phigs.h as follows:

```
typedef struct {
        Pfloat      red;        /* red, hue, and so on */
        Pfloat      green;      /* green, saturation, lightness,
                                   and so on */
        Pfloat      blue;       /* blue, value, saturation, and so on */
} Prgb;
```

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier for this escape is PUESCSETCURSOR, defined in phigs77.h to have the value -21.

LIDR The dimension of the input data record array, IDR, (returned by PACK DATA RECORD). ERROR HANDLING will be invoked if the specified dimension is less than zero.

IDR(LIDR)

An escape input data record packed by PACK DATA RECORD containing the input data for this escape. The arguments to PACK DATA RECORD should be as follows:

IL The number of integers = 3.

IA Contains three integer values.

Element 1 is the workstation identifier.

Element 2 specifies which cursor to change, one of the following enumerated values:

0	PHIGSNORMCURS	*Non-traversal cursor*
1	PHIGSTRAVCURS	*Traversal cursor*

Element 3 is the cursor type, one of the following enumerated values:

0	PHIGSCURSNONE	*No cursor*
1	PHIGSCURSDEFAULT	*Window parent's cursor*
3	PHIGSCURSCROSS	*Tracking cross*
12	PHIGSCURSNOTSIGN	*Not sign*

RL The number of real values = 3.

RA Contains the cursor colour. Elements 1 through 3 specify the red, green, and blue colour values, respectively.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

The *Set Cursor Type* escape provides control over the cursor on X Tool workstation displays. The workstation displays different cursors under different conditions. When the workstation is traversing posted structure networks, it displays the traversal cursor, using the specified colour. When the workstation is not traversing, its cursor is specified by normal cursor, using the specified colour. The type and colour of the normal cursor may be overridden by active input devices.

The cursor type can be set to *none* (no cursor), one of several specific types such as a tracking cross, or *default* (the window parent's cursor).

This escape may be called any time the specified workstation is open.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

054 Ignoring function, the specified workstation is not open

059 Ignoring function, the specified workstation does not have output capability (that is, the workstation category is not OUTPUT, OUTIN, or MO

113 Ignoring function, the colour index value is less than zero

351 Ignoring function, one of the fields within the escape data record is in error

SEE ALSO

WORKSTATION TYPE SET (3P)
INITIALIZE LOCATOR (3P)
INITIALIZE LOCATOR 3 (3P)

NAME

ESCAPE -22 – set double buffer mode for quick update method

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*not used*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -22 sets the double buffer mode for quick update method.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_DBL_BUF_QUM, which is defined in phigs.h to be -22.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u22. It is defined as:

```
typedef union {
        struct Pescape_in_u22 {
                Pint                  ws_id;          /* workstation
                                                         identifier */
                Phigs_db_qum_mode     db_qum_mode;    /* double buffer
                                                         QUM mode */
        } escape_in_u22;
} Pescape_in_data;
```

ws_id The workstation identifier.

db_qum_mode
Mode of double buffering for quick update method. The Phigs_db_qum_mode is specified as follows:

```
typedef enum {
        PHIGS_DB_QUM_OFF        = 0,    /* disable double buffer for
                                           quick update method */
        PHIGS_DB_QUM_ON         = 1     /* enable double buffer for
                                           quick update method */
} Phigs_db_qum_mode;
```

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETDBQUM, which is defined in phigs77.h to be -22.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the workstation identifier.

Element 2 is the mode setting of the double buffering for quick update method.

PHIGSDBQUMOFF	*Disable double buffer for quick update method*
PHIGSDBQUMON	*Enable double buffer for quick update method*

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

ESCAPE -22 sets the double buffer mode for quick update methods. The value of this mode is interpreted only by workstations that support double buffering. When used, if the mode is on, quick updates are drawn into the back buffer and are then flipped to the front buffer when drawing is done. When the mode is off, all quick updates are drawn in the front buffer.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

054 Ignoring function, the specified workstation is not open

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
ESCAPE -23 (3P)

NAME

ESCAPE -23 – query double buffer mode for quick update method

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID         function identification
INTEGER         LIDR          dimension of input data record array
CHARACTER*80    IDR(LIDR)     input data record
INTEGER         MLODR         maximum length of output data record
INTEGER         LODR          OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -23 queries the current double buffer mode for quick update method.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_DBL_BUF_QUM, which is defined in phigs.h to be -23.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u23. It is defined as:

```
typedef union {
        struct Pescape_in_u23 {
                Pint    ws_id;          /* workstation identifier */
        } escape_in_u23;
} Pescape_in_data;
```

ws_id The workstation identifier.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out The address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record Pescape_out_u23 to be filled by this escape function. It is defined as:

```
typedef union {
        struct Pescape_out_u23 {
                Pint                    err_ind;            /* error indicator */
                Phigs_db_qum_mode       db_qum_mode;        /* double buffer
                                                               QUM mode */
        } escape_out_u23;
} Pescape_out_data;
```

err_ind The error indicator of any error that this escape detects. See *ERRORS* below for possible values.

db_qum_mode
The current double buffer mode for quick update method. The Phigs_db_qum_mode is specified as follows:

```
typedef enum {
        PHIGS_DB_QUM_OFF        = 0,    /* disable double buffer for
                                           quick update method */
        PHIGS_DB_QUM_ON         = 1     /* enable double buffer for
                                           quick update method */
} Phigs_db_qum_mode;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQDBQUM, which is defined in phigs77.h to be -23.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 1.

IA Contains one integer value.

Element 1 is the workstation identifier.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. When the output record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL	The number of integers = 2.
IA	Contains two integer values:
	Element 1 is the error indicator.
	Element 2 is the current mode of double buffering for quick update method.

PHIGSDBQUMOFF	*Disable double buffer for quick update method*
PHIGSDBQUMON	*Enable double buffer for quick update method*

RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

ESCAPE -23 queries for the current mode setting of the double buffer for quick update methods.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
059	Ignoring function, the specified workstation does not have output capability (that is, the workstation category is not OUTPUT, OUTIN, or MO)

SEE ALSO

ESCAPE (3P)
ESCAPE -22 (3P)

NAME

ESCAPE -25 – define user-definable marker in PHIGS description table.

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store out )
Pint               func_id;    escape function identifier
Pescape_in_data    *in;        input data for the function
Pstore             store;      not used
Pescape_out_data   **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSCL, STCL, ARCL)

DESCRIPTION

Purpose

ESCAPE -25 defines a user-definable marker in the PHIGS description table.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_DEFINE_MARKER, which is defined in phigs.h to be -25.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u25. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u25 {
                Pint                type;       /* marker type */
                Ppoint_list_list    marker;     /* polyline marker data */
        } escape_in_u25;
} Pescape_in_data;
```

type An integer value specifying a user-definable marker type.

marker A Ppoint_list_list structure containing a list of polylines that defines the polymarker. The range that is applied to the nominal marker coordinates is from –1.0 to 1.0.

```
typedef struct {
        Pint           num_point_lists;    /* number of polylines */
        Ppoint_list    *point_lists;       /* list of polylines */
} Ppoint_list_list;
```

The *num_point_lists* component specifies the number of polylines in the polyline set that defines the polymarker. The *point_lists* component is a pointer to a list, *num_point_lists* long, of Ppoint_list structures. Each Ppoint_list structure defines one of the polylines in the set. Ppoint_list is defined in phigs.h as follows:

```
typedef struct {
        Pint      num_points;    /* number of points in the list */
        Ppoint    *points;       /* pointer to list of points */
} Ppoint_list;
```

The *num_points* component specifies the number of points used to define a polyline. The *points* component is a pointer to a list, *num_points* long, of Ppoint structures containing the *x* and *y* coordinates of each vertex of the polyline in MC. Ppoint is defined in phigs.h as follows:

```
typedef struct {
        Pfloat    x;    /* x coordinate */
        Pfloat    y;    /* y coordinate */
} Ppoint;
```

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCDEFINEMKR, which is defined in phigs77.h to be -25.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 2 + IA(2).

IA Contains 2 + IA(2) integer values.

Element 1 is the marker type.

Element 2 is the number of polylines, to be specified.

Elements 3 through IA(2 + IA(2)) contains an array of IA(2) integers containing the end indices for each point list in the polyline set.

RL The number of real values = 2×IA(2 + IA(2)).

RA Contains 2×IA(2 + IA(2)) floating values.

Elements 1 through IA(2 + IA(2)) contain the *x* components of the polyline set.

Elements 1 + IA(2 + IA(2)) through 2×IA(2 + IA(2)) contain the *y* components of the polyline set.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

ESCAPE -25 defines the shape for a specified user-definable polymarker type. The available user-definable marker types can be obtained by calling ESCAPE -27 (3P). The definition of user-defined polymarkers is then stored in the PHIGS description table.

A polymarker is defined by a set of unconnected polylines that is constructed by a set of point sequences; each point sequence in the set generates connected line segments. The points are specified in polymarker nominal coordinates. The polymarker nominal coordinate space is [–1.0, 1.0] in both dimensions.

ERRORS

004	Ignoring function, function requires state (PHOP, WSCL, STCL, ARCL)
351	Ignoring function, one of the fields in the escape data record is in error
–175	Ignoring function, the specified user-defined marker type is invalid

SEE ALSO

ESCAPE (3P)
ESCAPE -26 (3P)
ESCAPE -27 (3P)

NAME

ESCAPE -26 – query for user-defined marker definition from PHIGS description table

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

ESCAPE -26 returns the marker definition from the PHIGS description table for the specified user-definable marker type.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

Applications using the C binding must create a buffer to be used by this function as memory space for string the returned marker definition. This buffer is passed as the *store* argument.

The store buffer is a data area managed by SunPHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, the implementation manages this area to provide sufficient memory for the specific inquiry. The marker definition within the store buffer is accessed by the pointer pointed to by *marker*. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

func_id The function identifier for this escape is PUESC_INQ_USER_DEF_MARKER, which is defined in phigs.h to be -26.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u26. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u26 {
                Pint       type;    /∗ marker type ∗/
        } escape_in_u26;
} Pescape_in_data;
```

type An integer value specifies the user-definable marker type that is being queried.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. The memory buffer PHIGS is used to store returned information. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record, Pescape_out_u26. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u26 {
                Pint                  err_ind;     /∗ error indicator ∗/
                Ppoint_list_list      marker;      /∗ output pointer to buffer ∗/
        } escape_out_u26;
} Pescape_out_data;
```

err_ind An integer in which the system returns the error number of any error that this function detects.

marker A Ppoint_list_list structure that returns a list of polylines that defines the inquired polymarker.

```
typedef struct {
        Pint           num_point_lists;    /∗ number of polylines ∗/
        Ppoint_list    ∗point_lists;       /∗ list of polylines ∗/
} Ppoint_list_list;
```

The *num_point_lists* component specifies the number of polylines in the polyline set that defines the polymarker. The *point_lists* component is a pointer to a list, *num_point_lists* long, of Ppoint_list structures. Each Ppoint_list structure defines one of the polylines in the set. Ppoint_list is defined in phigs.h as follows:

```
typedef struct {
        Pint        num_points;    /* number of points in the list */
        Ppoint      *points;       /* pointer to list of points */
} Ppoint_list;
```

The *num_points* component specifies the number of points used to define a polyline. The *points* component is a pointer to a list, *num_points* long, of Ppoint structures containing the *x* and *y* coordinates of each vertex of the polyline in MC. Ppoint is defined in phigs.h as follows:

```
typedef struct {
        Pfloat    x;    /* x coordinate */
        Pfloat    y;    /* y coordinate */
} Ppoint;
```

FORTRAN Input Parameters

Applications using the FORTRAN binding must provide a long enough output data record for storing data. The maximum length of output data record must be passed in the MLODR argument. The required length is returned in the LODR argument. The caller can determine the required size by calling this function with MLODR set to zero, in which case PHIGS returns the length needed in LODR.

Error 2201 is returned if MLODR is too small, but not if it is zero.

FCTID The function identifier of this escape is PUESCINQUDEFMKR which is defined in phigs77.h to be -26.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 1.

IA Contains one integer value.

Element 1 is the polymarker type.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. It can be unpacked by using UNPACK DATA RECORD. The following resulting fields should be returned:

IL The number of integers = 2 + IA(2)

IA	Contains 2 + IA(2) integer values.
	Element 1 is the error indicator.
	Element 2 is the number of polylines to be returned.
	Elements 3 through 2 + IA(2) contains an array of IA(2) integers containing the end indices for each point list in the polyline set.
RL	The number of real values = 2×IA(2 + IA(2)).
RA	Contains 2×IA(2 + IA(2)) floating values.
	Elements 1 through IA(2 + IA(2)) contain the *x* components of the polyline set.
	Elements 1 + IA(2 + IA(2)) through 2×IA(2 + IA(2)) contain the *y* components of the polyline set.
SL	The number of strings = 0.

Execution

ESCAPE -26 returns the polymarker definitions for the specified marker type in the PHIGS description table.

A polymarker is defined by a set of unconnected polylines that are constructed by a set of point sequences; each point sequence in the set generates connected line segments. The points are specified in polymarker nominal coordinates. The polymarker nominal coordinate space is [–1.0, 1.0] in both dimensions. The available user-definable marker types can be obtained by calling ESCAPE -27 (3P).

If this function detects an error, then the *error indicator* indicates the error number of the detected error and no other output data is returned, except in the cases mentioned in *C Parameters* and *FORTRAN Parameters* sections above. If the function detects no error, then the *error indicator* is set to zero, and the information queried is available in the output parameters. Since this is an inquiry function, ERROR HANDLING is not invoked when this function detects an error.

ERRORS

002	Ignoring function, function requires state (PHOP, *, *, *)
105	Ignoring function, the specified marker type is not available on the specified workstation
351	Ignoring function, one of the fields in the escape data record is in error
900	Storage overflow has occurred in PHIGS
2201	Ignoring function, output parameter size insufficient (FORTRAN only)
–175	Ignoring function, the specified user-defined marker type is invalid

SEE ALSO

ESCAPE (3P)
ESCAPE -25 (3P)
ESCAPE -27 (3P)

NAME

ESCAPE -27 – query for list of available user-definable marker types from the PHIGS description table.

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

Use ESCAPE -27 to query for a list of user-definable polymarker types in the PHIGS description table.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_UDEF_MARKER_TYPES, which is defined in phigs.h to be -27.

in There is no input data record for this escape.

length The length of the list of *ints* items to return in the application allocated array, *type_list.ints* in the output data record. A *length* of zero may be specified, in order to have the *total_length* field (in the output record) return the total number of user-definable marker types.

start The first element of the list of user-definable marker types to return.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record, Pescape_out_u27. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_out_u27 {
                Pint          err_ind;          /* error indicator */
                Pint_list     type_list;        /* list of user-defined
                                                   marker types */
        } escape_out_u27;
} Pescape_out_data;
```

err_ind An integer in which the system returns the error number of any error detected by this function.

type_list
A Pint_list structure in which the list of user-definable polymarker types is returned. Pint_list is defined in phigs.h as follows:

```
typedef struct {
        Pint        num_ints;     /* number of Pints in list */
        Pint        *ints;        /* list of integers */
} Pint_list;
```

The *num_ints* component specifies the number of user-definable polymarker types. The *ints* list the values for those user-definable polymarker types.

The application must allocate memory for the *type_list.ints* list.

FORTRAN Input Parameters

Applications using the FORTRAN binding must provide a long enough output data record for storing data. The maximum length of output data record must be passed in the MLODR argument. The required length is returned in the LODR argument. The caller can determine the required size by calling this function with MLODR set to zero, in which case PHIGS returns the length needed in LODR.

Error 2201 is returned if MLODR is too small, but not if it is zero.

FCTID The function identifier of this escape is PUESCINQUDMKTPS which is defined in phigs77.h to be -27.

LIDR = 0, there is no input data record for this escape.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR)
The output data record is stored in this array. It can be unpacked by using

UNPACK DATA RECORD. The resulting fields should be returned:

IL	The number of integers = 2 + IA(2)
IA	Contains 2 + IA(2) integer values. Element 1 is the error indicator. Element 2 is the number of user-definable marker types. Elements 3 through 2 + IA(2) contains an array of IA(2) integers containing the values for all of the user-definable marker types.
RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

ESCAPE -27 returns the list of available user-definable marker types stored in the PHIGS description table.

If this function detects an error, then the *error indicator* indicates the error number of the detected error and no other output data is returned, except in the cases mentioned in *C Parameters* and *FORTRAN Parameters* sections above. If the function detects no error, then the *error indicator* is set to zero, and the inquired information is available in the output parameters. Since this is an inquiry function, ERROR HANDLING is not invoked when this function detects an error.

ERRORS

002	Ignoring function, function requires state (PHOP, *, *, *)
351	Ignoring function, one of the fields in the escape data record is in error
900	Storage overflow has occurred in PHIGS
2201	Ignoring function, output parameter size insufficient (FORTRAN only)

SEE ALSO

ESCAPE (3P)
ESCAPE -25 (3P)
ESCAPE -26 (3P)

NAME

ESCAPE -28 – simulate pick

SYNOPSIS

C Syntax

void
pescape(func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc(FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

Use ESCAPE -28 to simulate a pick, taking an aperture in device coordinates and returning the pick paths of the primitives within the aperture.

C Input Parameters

func_id The function identifier for this escape is PUESC_SIMULATE_PICK, defined in phigs.h to have the value -28.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u28 by this escape function. It is defined as:

```
typedef union {
        struct Pescape_in_u28 {
                Pint        ws_id;          /* workstation
                                               identifier */
                Pint        max_picks;      /* max number of picks
                                               to return */
                Pint        max_depth;      /* max depth each of
                                               pick */
                Pint        options;        /* option flags */
                Plimit      pick_ap;        /* pick aperture */
                Pfilter     filter          /* pick inclusion / ex-
                                               clusion filter */
        } escape_in_u28;
} Pescape_in_data;
```

ws_id The workstation on which to pick.

max_picks
The maximum number of pick paths to return.

max_depth
The maximum depth of each pick path to return.

options The options to use when picking. By default, the picks are returned in top first order, the last set of picks are returned, and all primitives that cross the pick aperture are returned. The options can be used to change the defaults. These options are defined in phigs.h as follows:

1	PHIGS_BOT_FIRST	*Return bottom elements first in the pick paths*
2	PHIGS_FIRST_PICKS	*Return the first set of primitives that are picked*
4	PHIGS_ONLY_VISIBLE	*Pick only the visible features of filled primitives (see the Execution section, below)*

Options may be combined via ORing.

pick_ap Primitives that cross this aperture are picked. The aperture is defined in drawable coordinates in pixels. The positive direction of the *x* axis of a drawable coordinate is from left to right; the positive *y* axis is from top to bottom. Plimit is defined in phigs.h as:

```
typedef struct {
        Pfloat   x_min;   /* x min */
        Pfloat   x_max;   /* x max */
        Pfloat   y_min;   /* y min */
        Pfloat   y_max;   /* y max */
} Plimit;
```

filter The pick filter. Pfilter is defined in phigs.h as:

```
typedef struct {
        Pint_list   incl_set;   /* inclusion set */
        Pint_list   excl_set;   /* exclusion set */
} Pfilter;
```

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record Pescape_out_u28 to be filled by this escape function. It is defined as:

```
typedef union {
        struct Pescape_out_u28 {
                Pint            err_ind;                /∗ error indicator ∗/
                Pint            num_picks;              /∗ number of picks
                                                           returned ∗/
                Ppick_path      ∗picks;                 /∗ returned pick paths ∗/
                Pint            total_picks;            /∗ total number of
                                                           picks ∗/
                Pint            total_max_depth;        /∗ max depth of any
                                                           pick ∗/
        } escape_out_u28;
} Pescape_out_data;
```

err_ind Any errors generated by this function.

num_picks
The number of picks returned.

picks The pick paths of the picked primitives. A maximum of *max_picks* is returned. Ppick_path is defined in phigs.h as:

```
typedef struct {
        Pint                depth;          /∗ pick path depth ∗/
        Ppick_path_elem     ∗path_list;     /∗ pick path list ∗/
} Ppick_path;
```

The depth returned for each path is the true depth of the pick, and is affected by the maximum pick depth. The number of elements returned in *path_list* may be less than depth. (The number is the depth or the maximum pick depth, whichever is smaller.) The picks are returned in traversal order when the first picks are being returned, and in reverse traversal order when the last picks are being returned.

total_picks
The total number of picks.

total_max_depth
The maximum depth of any pick.

FORTRAN Input Parameters

Applications using the FORTRAN binding must provide a long enough output data record for storing the data. The maximum length of the output data record is passed in the *MLODR* argument. The actual length used will be returned in LODR. The application can determine the required size by calling this function with MLODR set to zero, in which case SunPHIGS returns the length needed in LODR.

If MLODR is not zero, and the requested data is larger than the size specified by MLODR, error PEBUFSPAC (2200) is returned.

FCTID The function identifier of this escape is PUESCSIMPICK, which is defined in phigs77.h to be -28.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record packed by PACK DATA RECORD called with these arguments:

IL The number of integer values = 6 + IA(3) + IA(5).

IA Contains four integer values.
Element 1 is the workstation identifier.

Element 2 is the maximum number of picks to return.

Element 3 is the maximum pick depth to return.

Element 4 is the options field. By default, the picks are returned in top first order, the last set of picks are returned, and all primitives that cross the pick aperture are returned. Options can be used to change these defaults. The options are defined in phigs77.h as:

1	PHIGSBOTFIRST	*Return bottom elements first in the pick paths*
2	PHIGSFIRSTPICKS	*Return the first set of primitives that are picked*
4	PHIGSONLYVISIBLE	*Pick only the visible features of filled primitives (see the Execution section, below)*

The options can be combined by adding.

Element 5 specifies the number of names in the inclusion set.

Element 6 specifies the number of names in the exclusion set.

Element 7 through 8 + IA(3) are the names in the inclusion set.

Element 9 + IA(3) through 10 + IA(3) + IA(4) are the names in the exclusion set.

RL The number of real values = 4.

RA Contains 4 real values:

Element 1 is the minimum X value for the pick aperture.

Element 2 is the maximum X value for the pick aperture.

Element 3 is the minimum Y value for the pick aperture.

Element 4 is the maximum Y value for the pick aperture.

SL The number of strings = 0.

MLODR
The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output record array, ODR. If MLODR is 0, the total length needed is returned.

ODR(MLODR)
The output data record is stored in this array. When this output data record is passed to UNPACK DATA RECORD, the following fields are returned:

IL The number of integers. This is four, plus one for each pick returned, plus three for each level of each pick path returned. The maximum is: 4 + (max number of picks) + (3 × max number of picks × max depth of each pick).

IA Contains integer values.
Element 1 is the error indicator.

Element 2 is the number of picks returned.

Element 3 is the total number of picks. This may exceed the number of picks returned.

Element 4 is the maximum depth of any pick.

Elements 5 through IA(3) + 3 contain the true depth of each pick path. These may be more than the maximum depth specified in the input. The number of picks depths returned is either the total number of picks or the number requested, whichever is smaller.

The remaining elements contain the pick paths for the selected primitives. For each pick path, the structure ID, pick ID, and element number is returned. The picks are in traversal order when the first picks are being returned, and in reverse traversal order when the last picks are being returned. Top first or bottom first order of each pick is specified by the input option. The number of integer triplets that each pick path uses are the pick depth or the maximum pick depth, whichever is smaller. The number of picks depths returned is either the total number of picks or the number requested, whichever is smaller.

RL The number of real values = 0

SL The number of string values = 0

Execution

SIMULATE PICK returns the list of primitives that cross or are completely within the specified pick aperture. If HLHSR is active (based on the workstation's HLHSR mode and the current HLHSR ID), then primitives will be picked only when some part of the HLHSR affected primitive crosses or is completely within the pick aperture.

The list of returned picks can be limited to the first set of primitives, or the last set of primitives that satisfy the pick test. The order of the returned pick paths can be set to return the top pick elements first, or the bottom pick elements first.

Filled primitives can be picked based only on their visible features. By default, the interiors of filled primitives with interior styles empty and hollow are pickable. If the PHIGS_ONLY_VISIBLE option is selected, only the visible features of these primitives (if any) will be pickable.

The length of the pick paths can be limited. Information about the total number of picks is also returned.

The aperture is defined in drawable coordinates in pixels. The positive direction of the *x* axis of a drawable coordinate is from left to right; the positive *y* axis is from top to bottom. The front and back of the pick aperture are the front and back of device space.

The pick filter is compared to the traversal-time *current name set* of each primitive to determine if the primitive is pickable. The filter contains and inclusion set and an exclusion set of names. During traversal, a primitive is pickable by this function if at least one name in the *current name set* is in the inclusion set and no name in the *current name set* is in the exclusion set. This means the exclusion set has precedence over the inclusion set. Each name in the name set, inclusion set, and exclusion set is a small, positive, integer.

This escape is available on X Tool, X Drawable and X Drawable Region workstations.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
060	Ignoring function, the specified workstation is not of category OUTIN
207	Ignoring function, the specified path depth is less than zero (0)
350	Warning, the specified escape is not available on one or more workstations in this implementation. The escape is processed by those workstations on which it is available
351	Ignoring function, one of the fields in the escape data record is in error
900	Storage overflow has occurred in PHIGS
–172	Ignoring function, the specified pick aperture is outside the range of device coordinates.
–173	Ignoring function, the specified pick aperture is invalid; XMIN >= XMAX or YMIN >= YMAX

SEE ALSO

ADD NAMES TO SET (3P)
ESCAPE (3P)
INITIALIZE PICK (3P)
REMOVE NAMES FROM SET (3P)
SET PICK FILTER (3P)

NAME

ESCAPE -32 – set workstation’s NURB cache mode

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, out )
Pint            func_id;    escape function identifier
Pescapein       *in;        input data for the function
Pstore          store;      not used
Pescapeout      **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -32 sets the NURB cache mode for a workstation

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_NURB_CACHE_MODE, which is defined in phigs.h to be -32.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u32. It is defined as:

```
typedef union {
        struct Pescape_in_u32 {
                Pint                    wsid;           /* workstation
                                                           identifier */
                Phigs_nurb_cache_mode   cache_mode;     /* NURB cache
                                                           mode */
        } escape_in_u32;
} Pescape_in_data
```

wsid The workstation identifier.

cache_mode
The NURB cache mode for the workstation. Phigs_nurb_cache_mode is specified as follows:

```
typedef enum {
    PHIGS_NURB_CACHE_OFF = 0,   /* no caching of NURB primitives */
    PHIGS_NURB_CACHE_ON = 1     /* cache tessellated NURBs */
} Phigs_nurb_cache_mode;
```

store Unused

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETNURBCACHEMODE, which is defined in phigs77.h to be -32.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the workstation identifier.

Element 2 is the NURB cache mode for the workstation, the valid values are:

0	PHIGSNURBCACHEOFF	*No caching of NURB primitives*
1	PHIGSNURBCACHEON	*Cache tessellated NURB primitives*

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

ESCAPE -32 sets the NURB cache mode entry in the workstation state list for the specified workstation. Subsequent rendering of NURB primitives use this value.

NURB primitives are rendered by first tessellating into polylines and / or polygons. Subsequent displays of the primitive may be able to use the same tessellation. When NURB caching is active, the tessellation is cached for subsequent displays, possibly saving computation at the expense of memory. When NURB caching is not active, NURB primitives are recalculated each time they are displayed.

By default, NURB caching is active.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

054 Ignoring function, the specified workstation is not open

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
ESCAPE -33 (3P)
NON-UNIFORM B-SPLINE CURVE (3PP)
NON-UNIFORM B-SPLINE SURFACE (3PP)

NAME

ESCAPE -33 – inquire for workstation’s NURB cache mode

SYNOPSIS

C Syntax

void
pescape (func_id, in, out)

Pint	func_id;	*escape function identifier*
Pescapein	∗in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescapeout	∗∗out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER∗80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER∗80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, WSOP, ∗, ∗)

DESCRIPTION

Purpose

ESCAPE -33 inquires for the NURB cache mode for a workstation.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_NURB_CACHE_MODE, which is defined in phigs.h to be -33.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u33. It is defined as:

```
typedef union {
        struct Pescape_in_u33 {
                Pint    wsid;    /∗ workstation identifier ∗/
        } escape_in_u33;
} Pescape_in_data;
```

wsid The workstation identifier.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer *is* accessed via the pointer returned in Pescape_out_data.

C Output Parameters

out Address of a pointer to a Pescape_out_data union, which is allocated from *store* and contains the output data record Pescape_out_u33. It is defined as:

```
typedef union {
        struct Pescape_out_u33 {
                Pint                      error_ind;    /* error indicator */
                Phigs_nurb_cache_mode     cache_mode;   /* NURB cache
                                                           mode */
        } escape_out_u33;
} Pescape_out_data;
```

error_ind
The error indicator of any error that this escape detects. See *ERRORS*, below, for possible values.

cache_mode
The NURB cache mode for the workstation. Phigs_nurb_cache_mode is specified as follows:

```
typedef enum {
    PHIGS_NURB_CACHE_OFF = 0,      /* no caching of NURB primitives */
    PHIGS_NURB_CACHE_ON = 1        /* cache tessellated NURBs */
} Phigs_nurb_cache_mode;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQNURBCACHEMODE, which is defined in phigs77.h to be -33.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 1.

IA Contains one integer value.
Element 1 is the workstation identifier.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

MLODR
The length of the output data record, ODR.

LODR The length used in the output data record array ODR.

ODR(MLODR)
The output data record is stored in this array. When the output data record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the error indicator. See *ERRORS* below for possible values.

Element 2 is the NURB cache mode for the workstation. The valid values are:

0	PHIGSNURBCACHEOFF	*No caching of NURB primitives*
1	PHIGSNURBCACHEON	*Cache tessellated NURB primitives*

Execution ESCAPE -33 returns the NURB cache mode from the workstation state list for the specified workstation.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)

054 Ignoring function, the specified workstation is not open

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
ESCAPE -32 (3P)
NON-UNIFORM B-SPLINE CURVE (3PP)
NON-UNIFORM B-SPLINE SURFACE (3PP)

NAME

ESCAPE -34 – set workstation's silhouette edge mode

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, out )
Pint          func_id;    escape function identifier
Pescapein     *in;        input data for the function
Pstore        store;      not used
Pescapeout    **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID         function identification
INTEGER          LIDR          dimension of input data record array
CHARACTER*80     IDR(LIDR)     input data record
INTEGER          MLODR         maximum length of output data record
INTEGER          LODR          OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -34 sets the silhouette edge mode for a workstation

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_SIL_EDGE_MODE, which is defined in phigs.h to be -34.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u34. It is defined as:

```
typedef union {
        struct Pescape_in_u34 {
                Pint                   wsid;              /* workstation
                                                             identifier */
                Phigs_sil_edge_mode    sil_edge_mode;     /* silhouette edge
                                                             mode */
        } escape_in_u34
} Pescape_in_data;
```

wsid The workstation identifier.

sil_edge_mode
The silhouette edge mode for the workstation. Phigs_sil_edge_mode is specified as follows:

```
        typedef enum {
                PHIGS_SIL_EDGE_OFF = 0,         /* no silhouette edges */
                PHIGS_SIL_EDGE_ON = 1           /* enable silhouette edges */
        } Phigs_sil_edge_mode;
```

store unused.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETSILEDGEMODE, which is defined in phigs77.h to be -34.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the workstation identifier.

Element 2 is the silhouette edge mode. The valid values are:

0	PHIGSSILEDGEOFF	*No silhouette edges*
1	PHIGSSILEDGEON	*Enable silhouette edges*

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

ESCAPE -34 immediately sets the *requested* silhouette edge mode entry in the workstation's state list to the specified mode. The effect of calling ESCAPE -34 is not visible until the *requested* silhouette edge mode replaces the *current* silhouette edge mode. The time at which this occurs depends on the workstation's display update state.

This assignment is performed immediately and the silhouette edge update state is set to *Not Pending* if either of the following is true:

- The workstation display update state allows update.
- The *display surface empty* status in the workstation state list is EMPTY.

Otherwise, the silhouette edge update state is set to *Pending* and the requested silhouette edge mode will not replace the current silhouette edge mode until the next time the workstation is updated. The silhouette edge update state will be set to *Not Pending* at that time.

During traversal, when the workstation's current silhouette edge mode entry is on and the silhouette edge flag in the traversal state list is on, SunPHIGS attempts to render the silhouette edges, if any, implicit in area-filling primitives.

The silhouette edges of an area-filling primitive are edges drawn along the implicit boundary where the primitive switches from front-facing to back-facing. Specifically, on area-filling primitives with vertex normals, for each facet of the primitive, the facet is broken into triangles. For each of these triangles, if two of the vertices are front-facing and one is back-facing, then a silhouette edge is drawn between the two front-facing vertices.

Silhouette edges are drawn using the current edge attributes, independent of the the interior style of the primitive and independent of other edges on the primitive. The interaction between silhouette edges and other primitive edges with respect to patterning and wide line caps and joins is undefined. Silhouette edges are not drawn for culled primitive facets.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)
054	Ignoring function, the specified workstation is not open
351	Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
ESCAPE -35 (3P)
GENERALIZED STRUCTURE ELEMENT -11 (3P)

NAME

ESCAPE -35 – inquire for workstation’s silhouette edge mode

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, out )
Pint           func_id;    escape function identifier
Pescapein      *in;        input data for the function
Pstore         store;      handle to store object
Pescapeout     **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -35 inquires for the silhouette edge mode for a workstation

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_SIL_EDGE_MODE, which is defined in phigs.h to be -35.

in A pointer to a Pescape_in_data union containing the input data record, Pescape_in_u35. It is defined as:

```
typedef union {
        struct Pescape_in_u35 {
                Pint    ws_id;    /* workstation identifier */
        } escape_in_u35;
} Pescape_in_data;
```

wsid The workstation identifier.

store The store is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE (3P), PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data.

C Output Parameters

out A pointer to a Pescapeout union containing the output data record, Pescape_out_u35. It is defined as:

```
typedef union {
        struct Pescape_out_u35 {
                Pint                 error_ind;    /* error indicator */
                Pupd_st              state;        /* silhouette edge
                                                      update state */
                Phigs_sil_edge_mode  req_mode;     /* requested sil. edge
                                                      mode */
                Phigs_sil_edge_mode  cur_mode;     /* current sil. edge
                                                      mode */
        } escape_out_u35;
} Pescape_out_data;
```

error_ind
The error indicator of any error that this escape detects. See *ERRORS*, below, for possible values.

state The state of the silhouette edge mode (PUPD_NOT_PEND, PUPD_PEND).

req_mode,
req_mode is the requested silhouette edge mode for the workstation. cur_mode is the current silhouette edge mode for the workstation. Phigs_sil_edge_mode is specified as follows:

```
typedef enum {
        PHIGS_SIL_EDGE_OFF = 0,        /* no silhouette edges */
        PHIGS_SIL_EDGE_ON = 1,         /* enable silhouette edges */
} Phigs_sil_edge_mode;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQSILEDGEMODE, which is defined in phigs77.h to be -35.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR)
An ESCAPE data record, packed by PACK DATA RECORD with the following arguments:

IL The number of integers = 1.

IA Contains one integer value.

Element 1 is the workstation identifier.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

MLODR
The maximum length of the output data record, ODR

LODR The length used in the output data record array ODR.

ODR(MLODR)
The output data record is stored in this array. When the output data record is passed to UNPACK DATA RECORD, the following fields should be returned:

IL The number of integers = 4.

IA Element 1 is the error indicator. See *ERRORS*, below, for possible values.

Element 2 is the silhouette edge mode update state valid values are:

0	PNPEND	*Not pending*
1	PPEND	*Pending*

Element 3 is the requested silhouette edge mode.

Element 4 is the current silhouette edge mode.

The valid values for both the requested and current modes are:

0	PHIGSSILEDGEOFF	*No silhouette edges*
1	PHIGSSILEDGEON	*Enable silhouette edges*

Execution

ESCAPE -35 returns the workstation's silhouette edge mode update state, the requested silhouette edge mode, and the current silhouette edge mode for the specified workstation.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)

054 Ignoring function, the specified workstation is not open

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE (3P)
ESCAPE -34 (3P)
GENERALIZED STRUCTURE ELEMENT -11 (3P)

NAME

ESCAPE -40 – set workstation mode for anti-aliasing

SYNOPSIS

C Syntax

void
pescape (func_id, in, store, out)

Pint	func_id;	*escape function identifier*
Pescape_in_data	*in;	*input data for the function*
Pstore	store;	*handle to Store object*
Pescape_out_data	**out;	*OUT output data of the function*

FORTRAN Syntax

SUBROUTINE pesc (FCTID, LIDR, IDR, MLODR, LODR, ODR)

INTEGER	FCTID	*function identification*
INTEGER	LIDR	*dimension of input data record array*
CHARACTER*80	IDR(LIDR)	*input data record*
INTEGER	MLODR	*maximum length of output data record*
INTEGER	LODR	*OUT number of array elements used in ODR*
CHARACTER*80	ODR(MLODR)	*OUT output data record*

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -40 sets the workstation mode for anti-aliasing, which interacts with the anti-aliasing identifier set by GENERALIZED STRUCTURE ELEMENT -7 to enable various forms of anti-aliasing supported on some devices.

When the anti-aliasing mode is *None*, the anti-aliasing record is unused; and no anti-aliasing is performed, regardless of the value set by GENERALIZED STRUCTURE ELEMENT -7 (3P) (Set Stroke Anti-aliasing Identifier).

On GT workstations, the *Stroke* anti-aliasing mode enables Calligraphic (stroke) quality anti-aliasing of strokes of width 1, as controlled by GSE -7. The anti-aliasing record controls whether a one-pass or two-pass traversal is performed on the GT. A one-pass traversal is adequate if there is no anti-aliasing or when anti-aliasing blends the strokes into the constant background colour. Blending the colours of the strokes into the arbitrary colour found at the time the stroke is written introduces order-dependencies. (Optimum appearance requires drawing back-to-front.) Using two-pass stroke anti-aliasing reduces order-dependent artifacts, providing higher quality at the expense of performance. SunPHIGS renders opaque objects in the first pass; then disables Z-buffer writes but allows Z-buffer reads; and then renders transparent objects, including anti-aliased strokes.

Note: ESCAPE -43 (3P) (Set Transparency / Quality Trade Off) can also request a two-pass traversal. Its effect on performance with two-pass stroke anti-aliasing is not cumulative. If either mode requests a two-pass traversal, only two passes are performed. Anti-aliased strokes are transparent. If HLHSR is enabled, they are Z-buffer-tested, but the Z-buffer is not updated. If two passes are performed, anti-aliased strokes are drawn in the second pass.

When the anti-aliasing mode is *Image Anti-aliasing* on the GT, multiple samples with different sub-pixel positioning are averaged to form the final pixel value for display. A sample count in the anti-aliasing record controls the quality-performance trade-off. When the sample count is 1 (or less), anti-aliasing is disabled. When the sample count is greater than 1, image anti-aliasing is implemented with multi-pass stochastic anti-aliasing on the GT, with that number of passes (up to a maximum of about 20). The performance degradation factor is many times greater than the number of passes and increases with window size. GSE -7 (3P) is ignored on the GT when Image Anti-aliasing is enabled.

Note: The effect on performance of two-pass Alpha transparency (as set by ESCAPE -43 (3P)) *is* cumulative with Image Anti-aliasing on the GT — it doubles the number of passes, reducing the performance.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape i is PUESC_SET_AA_MODE, which is defined as -40 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u40. It is defined in phigs.h as:

```
typedef union {
        struct Pescape_in_u40 {
                Pint                ws_id;        /∗ workstation identifier ∗/
                Phigs_aalias_mode   aa_mode;      /∗ none, stroke, or image ∗/
                Phigs_aalias_rec    aa_record;    /∗ anti-aliasing record ∗/
        } escape_in_u40;
} Pescape_in_data;
```

ws_id The identifier of the workstation for which anti-aliasing is being set.

aa_mode The desired technique for anti-aliasing — none, stroke, or image. Phigs_aalias_mode is an enumerated type defined in phigs.h as:

```
typedef enum {
     PHIGS_AA_MODE_NONE,          /∗ do not anti-alias this
                                     workstation ∗/
     PHIGS_AA_MODE_STROKE,        /∗ anti-alias width-1 strokes ∗/
     PHIGS_AA_MODE_IMAGE          /∗ anti-alias multiple samples
                                    per pixel ∗/
} Phigs_aalias_mode;
```

aa_record A data record that provides additional information to control the workstation's anti-aliasing. This information is anti-aliasing mode-specific. The aa_record is unused for PHIGS_AA_MODE_NONE.

```
Phigs_aalias_rec is defined in phigs.h as:
typedef union {
    /* PHIGS_AA_MODE_STROKE: */
    Phigs_stroke_ aalias_passes     passes;/* number of traversal
                                              passes */
    /* PHIGS_AA_MODE_IMAGE: */
    Pint               sample_count;  /* number of samples to
                                              average */
} Phigs_aalias_rec;                   /* anti-aliasing record */
```

```
Phigs_stroke_aalias_passes is an enumerated type defined in
phigs.h as:
typedef enum {
    PHIGS_STROKE_AA_1_PASS,          /* no GSE with
                                        PHIGS_AA_ID_TO_ARBIT_BG */
    PHIGS_STROKE_AA_2_PASS,          /* use with
                                        PHIGS_AA_ID_TO_ARBIT_BG
                                        to reduce order-dependent
                                        artifacts; good for
                                        anti-aliased edges. */
} Phigs_stroke_aalias_passes;
```

For PHIGS_AA_MODE_STROKE, *passes* selects between a one-pass traversal, which is appropriate for anti-aliasing to the constant background colour, and a two-pass traversal, which reduces order-dependent artifacts when anti-aliasing to arbitrary background colours, at reduced performance.

For PHIGS_AA_MODE_IMAGE, *sample_count* is the number of samples to be averaged. When the sample count is one or less, image anti-aliasing is disabled. When the sample count is greater than one, image anti-aliasing is implemented with multi-pass stochastic anti-aliasing, with that number of passes (up to the maximum).

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETAAMODE, which is defined as -40 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD and called with these arguments:

IL The number of integers = 3.

IA Contains three integer values.

Element 1 is the identifier of the workstation for which the anti-aliasing mode is being set.

Element 2 is the desired anti-aliasing mode. Valid values are defined in phigs77.h:

0	PHIGSAAMODENONE	*Do not anti-alias this workstation*
1	PHIGSAAMODESTROKE	*Anti-alias width-1 strokes*
2	PHIGSAAMODEIMAGE	*Anti-alias multiple samples per pixel*

Element 3 implements the anti-aliasing record to provide additional information to control the workstation's anti-aliasing. Its use is anti-aliasing mode-specific. It is unused (although it should be present) for PHIGSAAMODENONE.

For PHIGSAAMODESTROKE, Element 3 selects between a one-pass traversal, which is appropriate for anti-aliasing to the constant background colour, and a two-pass traversal, which reduces order-dependent artifacts when anti-aliasing to arbitrary background colours, at reduced performance. Valid values are defined in phigs77.h:

1	PHIGSSTROKEAA1PASS	*Use when no GSE has ARBIT*
2	PHIGSSTROKEAA2PASS	*Use with ARBIT to reduce order-dependent artifacts; good for anti-aliased edges*

For PHIGSAAMODEIMAGE, Element 3 is an integer number of samples to be averaged. If the sample count is one or less, image anti-aliasing is disabled. If the sample count is greater than one, image anti-aliasing is implemented with multi-pass stochastic anti-aliasing, with that number of passes (up to the maximum).

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters There is no output data record for this escape.

Execution The *Set Anti-Aliasing Mode* escape sets the *anti-aliasing mode* and data record in the SunPHIGS workstation state list. This provides workstation control over anti-aliasing, much like the workstation's HLHSR mode is used with the HLHSR identifiers encountered during traversal, to control HLHSR.

Note: The width-1 stroke anti-aliasing identifier (GENERALIZED STRUCTURE ELEMENT -7) is ignored when anti-aliasing mode is None or Image.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)
054	Ignoring function, the specified workstation is not open
350	Warning, the specified escape is not available on one or more workstations in this implementation. The escape will be processed by those workstations on which it is available.
351	Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

GENERALIZED STRUCTURE ELEMENT -7 (3P)
ESCAPE -41 (3P)
ESCAPE -43 (3P)
COLOUR (7P)

NAME

ESCAPE -41 – inquire workstation mode for anti-aliasing

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;     escape function identifier
Pescape_in_data   *in;         input data for the function
Pstore            store;       handle to Store object
Pescape_out_data  **out;       OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER           FCTID          function identification
INTEGER           LIDR           dimension of input data record array
CHARACTER*80      IDR(LIDR)      input data record
INTEGER           MLODR          maximum length of output data record
INTEGER           LODR           OUT number of array elements used in ODR
CHARACTER*80      ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -41 returns the workstation anti-aliasing mode and data record. The data record contains mode-specific values, as described by ESCAPE -40.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_AA_MODE, which is defined as –41 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u41. It is defined as follows:

```
typedef union {
        struct Pescape_in_u41 {
                Pint         ws_id;     /* workstation identifier */
                Pinq_type    type;      /* PSET or PREALIZED */
        } escape_in_u41;
} Pescape_in_data;
```

ws_id The identifier of the workstation for which the anti-aliasing mode is being sought.

type An enumerated value specifying whether the values to be returned are those originally specified by the application (PSET) or those resulting after PHIGS mapped them to ones available on the workstation (PREALIZED).

The *Pinq_type* enumeration is defined as:

```
typedef enum {
        PSET,
        PREALIZED
} Pinq_type;
```

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union that is allocated from *store* and will contain the output data record, Pescape_out_u41. It is defined as follows:

```
typedef union {
        struct Pescape_out_u41 {
                Pint                err_ind;     /* error indicator */
                Phigs_aalias_mode   aa_mode;     /* aa mode */
                Phigs_aalias_rec    aa_record;   /* num of aa passes */
        } escape_out_u41;
} Pescape_out_data;
```

err_ind The error number of any error detected by this function.

aa_mode The desired technique for anti-aliasing — none, stroke, or image. Phigs_aalias_mode is an enumerated type defined in phigs.h as:

```
typedef enum {
        PHIGS_AA_MODE_NONE,       /* do not anti-alias this
                                     workstation */
        PHIGS_AA_MODE_STROKE,     /* anti-alias width-1 strokes */
        PHIGS_AA_MODE_IMAGE       /* anti-alias multiple samples
                                     per pixel */
} Phigs_aalias_mode;
```

aa_record A data record that provides additional information to control the workstation's anti-aliasing. This information is anti-aliasing mode-specific. The aa_record is unused for PHIGS_AA_MODE_NONE. Phigs_aalias_rec is defined in phigs.h as:

```
typedef union {
        /* PHIGS_AA_MODE_STROKE: */
        Phigs_stroke_aalias_passes  passes;        /* number of
                                                      traversal
                                                      passes */
        /* PHIGS_AA_MODE_IMAGE: */
        Pint                    sample_count;  /* number of
                                                  samples to
                                                  average */
    } Phigs_aalias_rec;                        /* anti-aliasing
                                                  record */
```

Phigs_stroke_aalias_passes is an enumerated type defined in phigs.h as:

```
typedef enum {
    PHIGS_STROKE_AA_1_PASS,     /* no GSE with
                                   PHIGS_AA_ID_TO_ARBIT_BG */
    PHIGS_STROKE_AA_2_PASS,     /* use with
                                    PHIGS_AA_ID_TO_ARBIT_BG
                                    to reduce order-dependent
                                    artifacts; good for
                                    anti-aliased edges. */
} Phigs_stroke_aalias_passes;
```

For PHIGS_AA_MODE_STROKE, *passes* selects between a one-pass traversal, which is appropriate for anti-aliasing to the constant background colour, and a two-pass traversal, which reduces order-dependent artifacts when anti-aliasing to arbitrary background colours, at reduced performance.

For PHIGS_AA_MODE_IMAGE, *sample_count* is the number of samples to be averaged. When the sample count is one or less, image anti-aliasing is disabled. When the sample count is greater than one, image anti-aliasing is implemented with multi-pass stochastic anti-aliasing, with that number of passes (up to the maximum).

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQAAMODE, which is defined as –41 in phigs77.h.

LIDR The dimension of *IDR* (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the identifier of the workstation for which image anti-aliasing is sought.

Element 2 is an enumerated variable specifying whether the values desired are those originally specified by the application program (*Set*) or those resulting after PHIGS mapped them to those available on the workstation (*Realized*). Valid values are:

PSET	*Set*

PREALI	*Realized*

RL The number of real values = 0.

SL The number of strings = 0.

MLODR The maximum length of the output data record, *ODR*.

FORTRAN Output Parameters

LODR The length used in the output data record array, *ODR*.

ODR(MLODR) The output data record is stored in this array. When the output data record is unpacked using UNPACK DATA RECORD, the resulting fields should be returned:

IL The number of integers = 3.

IA Contains three integer values.

Element 1 is the error number of any error detected by this function.

Element 2 is the anti-aliasing mode. Valid values are defined in phigs77.h:

0	PHIGSAAMODENONE	*Do not anti-alias this workstation*
1	PHIGSAAMODESTROKE	*Anti-alias width-1 strokes*
2	PHIGSAAMODEIMAGE	*Anti-alias multiple samples per pixel*

Element 3 is the integer in the anti-aliasing data record. When the anti-aliasing mode is PHIGSAAMODENONE, this element is zero.

For PHIGSAAMODESTROKE, Element 3 selects between a one-pass traversal, which is appropriate for anti-aliasing to the constant background colour, and a two-pass traversal, which reduces order-dependent artifacts when anti-aliasing to arbitrary background colours, at reduced performance. Valid values are defined in phigs77.h:

1	PHIGSSTROKEAA1PASS	*Use when no GSE has PHIGSAAIDARBIT*
2	PHIGSSTROKEAA2PASS	*Use with PHIGSAAIDARBIT to reduce order-dependent artifacts; good for anti-aliased edges*

For PHIGSAAMODEIMAGE, Element 3 is an integer number of samples to be averaged. When the sample count is one or less, image anti-aliasing is disabled. When the sample count is greater than one, image anti-aliasing is implemented with multi-pass stochastic anti-aliasing, with that number of passes.

RL The number of real values = 0.

SL	The number of strings = 0.

Execution

The *Inquire Anti-Aliasing Mode* escape returns the workstation's anti-aliasing mode and data record.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
350	Warning, the specified escape is not available on one or more workstations in this implementation. The escape will be processed by those workstations on which it is available.
351	Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE -40 (3P)

NAME

ESCAPE -42 – inquire workstation facilities for wide-stroke control.

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint               func_id;    escape function identifier
Pescape_in_data    *in;        input data for the function
Pstore             store;      handle to Store object
Pescape_out_data   **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER            FCTID          function identification
INTEGER            LIDR           dimension of input data record array
CHARACTER*80       IDR(LIDR)      input data record
INTEGER            MLODR          maximum length of output data record
INTEGER            LODR           OUT number of array elements used in ODR
CHARACTER*80       ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -42 returns the workstation facilities for controlling wide-stroke end-caps and joints between stroke segments, for one attribute set.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_WIDE_STROKE_FACILITIES, which is defined as –42 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u42. It is defined as follows:

```
typedef union {
        struct Pescape_in_u42 {
                Pint      ws_type;      /* bound to an open workstation */
                Pattrs    primitive;    /* attribute set of interest */
        } escape_in_u42;
} Pescape_in_data;
```

ws_type The type of the workstation the inquiry is for.

primitive The primitive the wide-stroke attribute is being applied to. It is defined by the Pattrs enumeration:

```
typedef enum {
```

```
                PATTR_LINE,
                PATTR_MARKER,
                PATTR_TEXT,
                PATTR_INT,
                PATTR_EDGE
        } Pattrs;
```

Note that PATTR_INT refers to the HOLLOW interior style.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union that is allocated from *store* and contains the output data record, Pescape_out_u42. It is defined as follows:

```
typedef union {
        struct Pescape_out_u42 {
                Pint        err_ind;       /* error indicator */
                Pint_list   end_caps;      /* list of end-caps supported */
                Pint_list   join_styles;   /* list of join-styles supported */
        } escape_out_u42;
} Pescape_out_data;
```

err_ind The error number of any error detected by this function.

end_caps The types of end caps supported as listed in Pint_list:

0	PHIGS_END_BUTT
1	PHIGS_END_SQUARE
2	PHIGS_END_ROUND

Pint_list is defined in phigs.h as follows:

```
typedef struct {
        Pint        num_ints;       /* number of Pints in list */
        Pint        *ints;          /* list of integers      */
} Pint_list;
```

The *num_ints* component specifies the number of elements in the list. The *ints* component is a pointer to a list *num_ints* long.

join_styles The type of joint styles supported as listed in Pint_list:

0	PHIGS_JOIN_BUTT
1	PHIGS_JOIN_BEVELED
2	PHIGS_JOIN_MITRED
3	PHIGS_JOIN_ROUND

Pint_list is defined above.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQWIDESTROKE, which is defined as –42 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

IL The number of integers = 4.

IA Contains four integer values.

Element 1 identifies the workstation type the inquiry is for.

Element 2 is an enumerated variable that identifies the attribute set of interest. Valid values are:

0	PPLATT	*Polyline attribute*
1	PPMATT	*Polymarker attributes*
2	PTXATT	*Text attribute*
3	PINATT	*Interior attributes*
4	PEDATT	*Edge attributes*

Note that polymarker and hollow interior style strokes cannot be wide.

Element 3 is the index of the wide-stroke end-cap desired. A value of 0 can be used to inquire for the length of the list in PHIGS.

Element 4 is the index of the wide-stroke join style desired. A value of 0 can be used to inquire for the length of the list in PHIGS.

RL The number of real values = 0.

SL The number of strings = 0.

MLODR The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR) The output data record is stored in this array. It can be unpacked by using UNPACK DATA RECORD, the resulting fields should be returned:

IL The number of integers = 5.

IA Contains five integer values.

Element 1 is the error number of any error detected by this function.

Element 2 is the total number of supported end-cap styles.

Element 3 is the Nth element of the wide-stroke end cap list, as specified by Element 3 in the input data record.

0	PHIGSENDBUTT
1	PHIGSENDSQUARE

2	PHIGSENDROUND

Element 4 is the total number of supported join styles.

Element 5 is the Nth element of the wide-stroke join style list, as specified by Element 4 in the input data record.

0	PHIGSJOINBUTT
1	PHIGSJOINBEVELED
2	PHIGSJOINMITRED
3	PHIGSJOINROUND

RL The number of real values = 0.

SL The number of strings = 0.

Execution The *Inquire Wide-Stroke Control Information* escape returns the supported types of wide-stroke end-caps and types of joints between stroke segments. The end-cap choices include rounded, squared, and butt. The joint choices include beveled, mitred, rounded, and butt. The default value for both end-cap and joints is butt for SunPHIGS workstations.

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

051 Ignoring function, this information is not yet available for this generic workstation type; open a workstation of this type and use the specific workstation type

052 Ignoring function, workstation type is not recognized by the implementation

351 Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

GENERALIZED STRUCTURE ELEMENT (3P)
GENERALIZED STRUCTURE ELEMENT -9 (3P)
GENERALIZED STRUCTURE ELEMENT -10 (3P)

NAME

ESCAPE -43 – set transparency-quality / performance trade-off

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER         FCTID         function identification
INTEGER         LIDR          dimension of input data record array
CHARACTER*80    IDR(LIDR)     input data record
INTEGER         MLODR         maximum length of output data record
INTEGER         LODR          OUT number of array elements used in ODR
CHARACTER*80    ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -43 selects the technique for the transparency PHIGS PLUS surface attribute, which is a trade-off between quality and performance.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_TRANSP_QUALITY, which is defined as –43 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u43. It is defined as follows:

```
typedef union {
        struct Pescape_in_u43 {
                Pint            ws_id;      /* workstation identifier */
                Phigs_transp    quality;    /* transparency quality trade-off */
        } escape_in_u43;
} Pescape_in_data;
```

ws_id The identification code of the workstation for which transparency quality is being set.

quality The technique used for the transparency PHIGS PLUS surface attribute. It can be *none*, *workstation-dependent*, *screen-door*, or *alpha 2 pass* as defined by the Phigs_transp enumeration:

```
typedef enum {
        PHIGS_TRANSP_NONE = 0,
        PHIGS_TRANSP_WS_DEPENDENT = 1,
        PHIGS_TRANSP_SCREEN_DOOR = 2,
        PHIGS_TRANSP_ALPHA_2_PASS = 3
} Phigs_transp;
```

The default technique is *none* for all devices.

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETTRANSPQUAL, which is defined as –43 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 identifies the workstation ID for which transparency quality is being set.

Element 2 is an enumerated variable that identifies the technique used for the transparency PHIGS PLUS surface attribute. It can be *none*, *workstation-dependent*, *screen-door*, or *alpha 2 pass* as follows:

```
PHIGSTRANSPNONE,
PHIGSTRANSPWSDEPENDENT,
PHIGSTRANSPSCREENDOOR,
PHIGSTRANSPALPHA2PASS
```

The default technique is *none* for all devices.

RL The number of real values = 0.

SL The number of strings = 0.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

The *Set Transparency Quality* escape selects the technique for the transparency PHIGS PLUS surface attribute. The technique applies to the entire image (the entire traversal). It allows the description of a surface to specify what fraction of its color should be visible; the underlying objects provide the remainder of the image color in that area.

The techniques for GT true colour workstations are *none*, *screen-door transparency*, and *alpha transparency*. The techniques for GT indexed workstations are *none* and *screen-door transparency*.

Screen-door transparency is faster and independent of rendering order, but the display quality is not as good. It can be used by indexed colour or true colour workstations. Screen-door transparency allows certain pixels to be overwritten, but leaves other pixels under the area untouched (not overwritten). The fraction of the pixels untouched is the user's transparency coefficient, which is set with SET AREA PROPERTIES (3PP). Screen-door transparency normally spreads out at random the pixels of each kind. However, superimposing more than one transparent object at the same transparency degree will look the same as a single such object. The best results with screen-door transparency are for transparency coefficients of 0.25, 0.50, and 0.75.

Alpha transparency blends a fraction of the pixel color being written into the color of the pixel from other objects. It thus provides improved visual results, but is order-dependent. Superimposing transparent objects with alpha transparency will give an additive effect. It requires two passes through the display list, thus reducing performance by about half. The first pass renders opaque objects with the Z-buffer enabled. The second pass blends in transparent objects assuming a back-to-front rendering order. The second pass Z-buffer tests where HLHSR (hidden line and hidden surface removal) is enabled, but does not write the Z-buffer because transparent objects don't obscure other objects. Alpha transparency is not supported on indexed colour workstations; screen-door transparency is substituted on indexed colour workstations. True colour workstations also substitute screen-door transparency in place of alpha transparency when depth-cueing is enabled.

The quality / performance trade-off must be made by the application. For best results using alpha transparency:

1. Draw objects in back-to-front order; an object viewed from an arbitrary angle may not produce the best results.
2. Reduce the transparency coefficient below 0.2 to reduce rendering artifacts.
3. If drawing multiple transparent objects that are inside each other, draw the inside objects before the outside objects and cull back-facing surfaces.
4. If none of the above is possible, keep the colours of the transparent objects as similar as possible. Since specular lighting modifies the intrinsic colours, avoiding it for transparent objects helps to keep the colours similar.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, *, *)
054	Ignoring function, the specified workstation is not open
351	Ignoring function, one of the fields in the escape data record is in error

SEE ALSO

ESCAPE -44 (3P)
SET REFLECTANCE PROPERTIES (3PP)
SET INTERIOR REPRESENTATION PLUS (3PP)

NAME

ESCAPE -44 – inquire for transparency quality

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -44 inquires for the workstation's technique for the transparency PHIGS PLUS surface attribute.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_TRANSP_QUALITY, which is defined as –44 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_44. It is defined as follows:

```
typedef union {
        struct Pescape_in_u44 {
                Pint          ws_id;   /* workstation identifier */
                Pinq_type     type;    /* PSET or PREALIZED */
        } escape_in_u44;
} Pescape_in_data;
```

ws_id The identifier of the workstation for which transparency quality sought.

type An enumerated value specifying whether the values to be returned are those originally specified by the application (PSET), or those resulting after PHIGS mapped them to those available on the workstation (PREALIZED). A Pinq_type structure is defined as:

```
typedef enum {
        PSET,
        PREALIZED
} Pinq_type;
```

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union that is allocated from *store* and contains the output data record, Pescape_out_u44. It is defined as follows:

```
typedef union {
        struct Pescape_out_u44 {
                Pint            err_ind;        /∗ error indicator ∗/
                Phigs_transp    quality;        /∗ transparency quality trade-off ∗/
        } escape_out_u44;
} Pescape_out_data;
```

err_ind The error number of any error detected by this function.

quality The technique used for the transparency PHIGS PLUS surface attribute. It may be *none*, *workstation-dependent*, *screen-door*, or *alpha 2 pass* as defined by the Phigs_transp enumeration:

```
typedef enum {
        PHIGS_TRANSP_NONE = 0,
        PHIGS_TRANSP_WS_DEPENDENT = 1,
        PHIGS_TRANSP_SCREEN_DOOR = 2,
        PHIGS_TRANSP_ALPHA_2_PASS = 3
} Phigs_transp;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQTRANSPQUAL, which is defined as −44 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 identifies the workstation for which transparency quality is sought.

Element 2 is an enumerated variable specifying whether the values desired are those originally specified by the application program (*Set*) or those resulting after PHIGS mapped them to those available on the workstation (*Realized*). Valid values are:

PSET	*Set*
PREALI	*Realized*

RL The number of real values = 0.

SL The number of strings = 0.

MLODR The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR) The output data record is stored in this array. It may be unpacked by using UNPACK DATA RECORD. The resulting fields should be returned:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is the error number of any error detected by this function.

Element 2 is the technique used for the transparency PHIGS PLUS surface attribute. It may be *none*, *workstation-dependent*, *screen-door*, or *alpha 2 pass* as follows:

0	PHIGSTRANSPNONE
1	PHIGSTRANSPWSDEPENDENT
2	PHIGSTRANSPSCREENDOOR
3	PHIGSTRANSPALPHA2PASS

RL The number of real values = 0.

SL The number of strings = 0.

Execution

The *Inquire Transparency Quality* escape inquires about the technique used for the transparency PHIGS PLUS surface attribute. The technique applies to the entire image (the entire traversal).

ERRORS

003 Ignoring function, function requires state (PHOP, WSOP, *, *)

054 Ignoring function, the specified workstation is not open

057 Ignoring function, specified workstation is of category MI

SEE ALSO

ESCAPE -43 (3P)

NAME

ESCAPE -45 – set view representation for left eye

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint              func_id;    escape function identifier
Pescape_in_data   *in;        input data for the function
Pstore            store;      handle to Store object
Pescape_out_data  **out;      OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID         function identification
INTEGER          LIDR          dimension of input data record array
CHARACTER*80     IDR(LIDR)     input data record
INTEGER          MLODR         maximum length of output data record
INTEGER          LODR          OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)    OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -45 defines a view representation in the workstation’s left view table that can be used for the left eye to create a stereo (binocular) image on certain devices.

The *current view index* selects a view representation from the workstation’s left view table only when the device is stereo-compatible and *stereo mode* is selected for both the device and the PHIGS workstation. The current view index selects a representation from the standard PHIGS workstation view table for the right eye.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_SET_LEFT_VIEW_REP3, which is defined as -45 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u45. It is defined as follows:

```
typedef union {
        struct Pescape_in_u45 {
                Pint          ws_id;        /* workstation identifier */
                Pint          ind;          /* index of view representation to
                                               be set */
                Pview_rep3    *view_rep;    /* view representation, left eye */
        } escape_in_u45;
} Pescape_in_data;
```

ws_id The identifier of the workstation for which a view representation is being defined.

ind The index of the view representation that is to be defined.

view_rep
A pointer to the view representation for the left eye contained in the Pview_rep3 structure:

```
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;                    /∗ output data record (structure) ∗/
```

Pmatrix3 is defined in phigs.h as:

```
    typedef Pfloat Pmatrix3[4][4];
```

Plimit3 is defined in phigs.h as:

```
typedef struct {
        Pfloat   x_min;   /∗ x minimum ∗/
        Pfloat   x_max;   /∗ x maximum ∗/
        Pfloat   y_min;   /∗ y minimum ∗/
        Pfloat   y_max;   /∗ y maximum ∗/
        Pfloat   z_min;   /∗ z minimum ∗/
        Pfloat   z_max;   /∗ z maximum ∗/
} Plimit3;
```

Pclip_ind is an enumerated type defined in phigs.h as:

```
typedef enum {
        PIND_NO_CLIP,
        PIND_CLIP
} Pclip_ind;
```

store Not used.

C Output Parameters

There is no output data record for this escape.

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCSETLEFTVIEWREP3, which is defined as –45 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

IL	The number of integers = 5.
IA	Contains five integer values.
	Element 1 identifies the workstation for which view representation is being defined.
	Element 2 is the index of the view representation that is being defined.
	Element 3 is the *x-y* clip indicator for the *x* and *y* axes.
	Element 4 is the back clipping indicator, which controls clipping against the *back* plane, sometimes called the *yon* plane.
	Element 5 is the front clipping indicator, which controls clipping against the *front* plane, sometimes called the *fore* or *hither* plane.
	Valid clipping indicators as defined in phigs77.h are one of the enumeration values:

PNCLIP	*No clipping*
PCLIP	*Clipping*

RL	The number of real values = 38.
RA	Contains 38 real values.
	Elements 1 through 16 contain the 4 × 4 homogeneous view orientation matrix in row-major order.
	Elements 17 through 32 contain the 4 × 4 homogeneous view mapping matrix in row-major order.
	Elements 33 through 38 contain the 6 clipping limits:Xmin, Xmax, Ymin, Ymax, Zmin, and Zmax.
SL	The number of strings = 0.

MLODR The maximum length of the output data record, *ODR*.

FORTRAN Output Parameters

There is no output data record for this escape.

Execution

The *Set Left Eye View Representation* escape sets an entry in the workstation's left view table that can be used, along with the corresponding entry in the workstation's view table to create a stereo (binocular) image on certain devices that require a stereo-compatible display and shutter system. This entry requires a stereo-compatible display and shutter system.

When a frame buffer (the entire monitor) is in stereo mode, the image for one eye is refreshed from one area of the frame buffer, and then the image for the other eye is refreshed from another area. Some windows on the screen may contain non-stereo information, such as text-editing windows. The information for these windows is written into both the left and right areas of the frame buffer using a special frame buffer mode. This provides the same information to both eyes. PHIGS workstations may display different views for the left and right eyes on such hardware.

To see the stereo effect, the viewer must use *shutter* hardware. This may consist of a polarized screen covering the monitor and polarized glasses. Goggles with internal shutters controlled by synchronizing signals from the frame buffer might also be used.

On a GT device, the initialization program *gtconfig(8)* may select stereo (binocular) or nonstereo (monocular) display mode for a stereo-compatible display system. WORKSTATION TYPE SET (3P) must enable the PHIGS_STEREO_MODE for the PHIGS workstation to use both the left and right views during traversal. The default mode is monocular; this escape sets the left view representation, but it is not used during traversal.

To produce the orientation and mapping matrices, use the utility functions EVALUATE VIEW ORIENTATION MATRIX 3 (3P) and EVALUATE VIEW MAPPING MATRIX 3 (3P). Set the right eye's view representation by using the standard PHIGS function SET VIEW REPRESENTATION 3 (3P). The current index attribute at traversal time selects corresponding left and right views for each primitive. If the right view is set without the left, then the default left view is used. The *current view index* attribute at traversal time selects corresponding right and left views for each primitive. If the right view is set without the left, the default left view will be used. No escape sets the left eye's 2D view; the 3D view is a superset.

The left and right eye's view representations are typically identical, except for slightly offset Projection Reference Points. See the example program stereo.c for a commented example of view calculation. If the X-Y clipping indicator is enabled, then the window limits passed for the two eyes to EVALUATE VIEW ORIENTATION MATRIX 3 typically match, to display the common area. Otherwise, the left and right window limits may slightly differ.

Locator and stroke input is transformed by using the standard (right eye) view representations.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)
054	Ignoring function, the specified workstation is not open
057	Ignoring function, specified workstation is of category MI
115	Ignoring function, the view index value is less than one
150	Ignoring function, setting this view table entry would exceed the maximum number of entries allowed in the workstation's view table
153	Ignoring function, invalid view clipping limits; XMIN ≥ XMAX, YMIN ≥ YMAX, or ZMIN > ZMAX
154	Ignoring function, the view clipping limits are not within NPC range

SEE ALSO

ESCAPE -46 (3P)
EVALUATE VIEW MAPPING MATRIX (3P)
EVALUATE VIEW ORIENTATION MATRIX (3P)
SET VIEW INDEX (3P)
SET VIEW REPRESENTATION 3 (3P)

SET VIEW TRANSFORMATION INPUT PRIORITY (3P)
UPDATE WORKSTATION (3P)
WORKSTATION TYPE SET (3P)
SunPHIGS stereo.c example program

NAME

ESCAPE -46 – inquire for view representation for left eye

SYNOPSIS

C Syntax

```
void
pescape ( func_id, in, store, out )
Pint               func_id;     escape function identifier
Pescape_in_data    *in;         input data for the function
Pstore             store;       handle to Store object
Pescape_out_data   **out;       OUT output data of the function
```

FORTRAN Syntax

```
SUBROUTINE pesc ( FCTID, LIDR, IDR, MLODR, LODR, ODR )
INTEGER          FCTID          function identification
INTEGER          LIDR           dimension of input data record array
CHARACTER*80     IDR(LIDR)      input data record
INTEGER          MLODR          maximum length of output data record
INTEGER          LODR           OUT number of array elements used in ODR
CHARACTER*80     ODR(MLODR)     OUT output data record
```

Required PHIGS Operating States

(PHOP, WSOP, *, *)

DESCRIPTION

Purpose

ESCAPE -46 inquires about a left-eye view representation from a workstation enabled for a binocular image. The standard PHIGS view representation is used for the right eye.

A window can only have a binocular image when the device is in stereo mode. Use the device initialization program gtconfig(1M) to select stereo (binocular) or nonstereo (monocular) display mode for the monitor. The default mode is monocular; the same image for both eyes.

Note: This function may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

func_id The function identifier for this escape is PUESC_INQ_LEFT_VIEW_REP3, which is defined as -46 in phigs.h.

in A pointer to a Pescape_in_data union containing the input data record Pescape_in_u46. It is defined as follows:

```
typedef union {
        struct Pescape_in_u46 {
                Pint    ws_id;    /*workstation identifier */
                Pint    ind;      /* index of view representation to be */
                                  /* inquired for */
        } escape_in_u46;
} Pescape_in_data;
```

ws_id The identifier of the workstation for which a view representation is being inquired about.

ind The index of the view representation that is being inquired about.

store The store buffer is a data area managed by PHIGS. While the application is responsible for creating the initial buffer through a call to CREATE STORE, PHIGS manages this area to provide sufficient memory for the specific inquiry. The escape data record within the store buffer is accessed via the pointer returned in Pescape_out_data. This buffer must exist prior to calling this function (see CREATE STORE (3P)).

C Output Parameters

out Address of a pointer to a Pescape_out_data union that is allocated from *store* and contains the output data record, Pescape_out_u46. It is defined as follows:

```
typedef union {
        struct Pescape_out_u46 {
                Pint           err_ind;     /∗ error indicator ∗/
                Pupd_st        upd_st;      /∗ view transformation update ∗/
                                               state ∗/
                Pview_rep3     ∗cur_rep;    /∗ current left eye view ∗/
                                               representation ∗/
                Pview_rep3     ∗req_rep;    /∗ requested left eye view ∗/
                                               representation ∗/
        } Pescape_out_u46;
} Pescape_out_data;
```

err_ind The error number of any error detected by this function.

upd_st The *view transformation update state*. Pupdatest is either PUPD_PENDING (Pending) or PUPD_NOT_PENDING (Not Pending).

cur_rep A pointer to the definition of the *current left eye view representation*.

req_rep A pointer to the definition of the *requested left eye view representation*.

Pviewrep3 is defined in phigs.h as follows:

```
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;
```

Note that both the current and the requested left eye view representation is returned along with the view transformation's update state (PUPD_PENDING or PUPD_NOT_PENDING). This is the same type of information as is returned by INQUIRE VIEW REPRESENTATION, which is used for the right eye.

Pmatrix3 is defined in phigs.h as:

```
      typedef Pfloat Pmatrix3[4][4];
```

Plimit3 is defined in phigs.h as:

```
typedef struct {
        Pfloat   x_min;   /* x minimum */
        Pfloat   x_max;   /* x maximum */
        Pfloat   y_min;   /* y minimum */
        Pfloat   y_max;   /* y maximum */
        Pfloat   z_min;   /* z minimum */
        Pfloat   z_max;   /* z maximum */
} Plimit3;
```

Pclip_ind is an enumerated type defined in phigs.h as:

```
typedef enum {
        PIND_NO_CLIP,
        PIND_CLIP
} Pclip_ind;
```

FORTRAN Input Parameters

FCTID The function identifier of this escape is PUESCINQLEFTVIEWREP3, which is defined as –46 in phigs77.h.

LIDR The dimension of IDR (returned by PACK DATA RECORD).

IDR(LIDR) An escape data record packed by PACK DATA RECORD called with these arguments:

- *IL* The number of integers = 3.
- *IA* Contains three integer values.

  Element 1 identifies the workstation for which view representation is sought.

  Element 2 is the index of the view representation that is sought.

  Element 3 is whether current or requested values are to be returned (PCURVL, PRQSVL). The requested values can differ from the current values if the update state is Pending.
- *RL* The number of real values = 0.
- *SL* The number of strings = 0.

MLODR The maximum length of the output data record, ODR.

FORTRAN Output Parameters

LODR The length used in the output data record array, ODR.

ODR(MLODR) The output data record is stored in this array. It may be unpacked by using UNPACK DATA RECORD. The resulting fields should be returned:

IL The number of integers = 5.

IA Contains five integer values.

Element 1 is the error number of any error this function detects.

Element 2 is the *view transformation update state*. Possible values are one of the enumeration values:

0	PNPEND	*Not pending*
1	PPEND	*Pending*

Element 3 is the X-Y clipping indicator which is either PNCLIP (*Not Clipping*) or PCLIP (*Clipping*).

Element 4 is the *back* clipping indicator.

Element 5 is the *front* clipping indicator.

Valid clipping indicators as defined in phigs77.h are one of the enumeration values:

0	PNCLIP	*No clipping*
1	PCLIP	*Clipping*

RL The number of real values = 38.

RA Contains 38 real values.

Elements 1 through 16 contain the 4 × 4 array in which the view orientation matrix is returned.

Elements 1a7 through 32 contain the 4 × 4 array in which the view mapping matrix is returned.

Elements 33 through 38 contain an array in which the view clipping limits are returned; the first two elements of this array give the minimum and maximum clipping values in NPC for *x*, the next two for *y*, and the last two for *z*.

SL The number of strings = 0.

Execution

The ESCAPE -46 (3P) (Inquire Left Eye View Representation) inquires about a left-eye view representation for a binocular image. The standard PHIGS view representation is used for the right eye. When a frame buffer (the entire monitor) is in stereo mode, the image for one eye is refreshed from one area of the frame buffer, and then the image for the other eye is refreshed from another area. Some windows on the screen may contain non-stereo information, such as text-editing windows. The information for these windows is written into both the left and right areas of the frame buffer using a special frame buffer mode. This provides the same information to both eyes.

To see the stereo effect, the viewer must use *shutter* hardware. This may consist of a polarized screen covering the monitor and polarized glasses. Goggles with internal shutters controlled by synchronizing signals from the frame buffer might also be used.

ERRORS

003	Ignoring function, function requires state (PHOP, WSOP, ∗, ∗)
054	Ignoring function, the specified workstation is not open
057	Ignoring function, specified workstation is of category MI
101	Ignoring function, the specified representation has not been defined
114	Ignoring function, the view index value is less than zero

SEE ALSO

ESCAPE -45 (3P)
INQUIRE VIEW REPRESENTATION (3P)
INQUIRE PREDEFINED VIEW REPRESENTATION (3P)
INQUIRE VIEW FACILITIES (3P)
SET VIEW INDEX (3P)
INQUIRE VIEW REPRESENTATION (3P)

NAME

GENERALIZED DRAWING PRIMITIVE – create 2D GDP elements such as circle, ellipse, or rectangular grid

SYNOPSIS

C Syntax

void
pgdp (point_list, gdp_id, gdp_data)

Ppoint_list	∗point_list;	*array of points*
Pint	gdp_id;	*gdp function identifier*
Pgdp_data	∗gdp_data;	*data record pointer*

FORTRAN Syntax

SUBROUTINE pgdp (N, PXA, PYA, PRIMID, LDR, DATREC)

INTEGER	N	*number of points (>= 0)*
REAL	PXA(∗), PYA(∗)	*coordinates of points (MC)*
INTEGER	PRIMID	*GDP identifier*
INTEGER	LDR	*dimension of data record array*
CHARACTER∗80	DATREC(LDR)	*data record*

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE creates the following GDP structure elements:

- circle
- circular arc
- circular arc close
- annotation circle
- annotation circular arc
- annotation circular arc close
- ellipse
- elliptical arc
- elliptical arc close
- annotation ellipse
- annotation elliptical arc
- annotation elliptical arc close
- image
- rectangular grid
- radial grid
- mono-encoded text
- mono-encoded annotation text relative

If the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE function is inserted into the open structure after the element pointed to by the element pointer. If the current edit mode is REPLACE, the GENERALIZED DRAWING PRIMITIVE element replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new structure element.

Each generalized drawing primitive has its own page in the reference manual. The name of the page is generalized drawing primitive-xx, where xx is replaced by the GDP identifiers; for example, GENERALIZED DRAWING PRIMITIVE -2.

Note: Support for GDPs is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id An integer specifying the GENERALIZED DRAWING PRIMITIVE to be performed. Predefined values defined in phigs.h are:

-2	PUGDP_CIRCLE	*Circle*
-3	PUGDP_CIRC_ARC	*Circular arc*
-4	PUGDP_CIRC_ARC_CLOSE	*Circular arc close*
-5	PUGDP_ANNOT_CIRCLE	*Annotation circle*
-6	PUGDP_ANNOT_CIRC_ARC	*Annotation circular arc*
-7	PUGDP_ANNOT_CIRC_ARC_CLOSE	*Annotation circular arc close*
-8	PUGDP_ELLIPSE	*Ellipse*
-9	PUGDP_ELLP_ARC	*Elliptical arc*
-10	PUGDP_ELLP_ARC_CLOSE	*Elliptical arc close*
-11	PUGDP_ANNOT_ELLIPSE	*Annotation ellipse*
-12	PUGDP_ANNOT_ELLP_ARC	*Annotation elliptical arc*
-13	PUGDP_ANNOT_ELLP_ARC_CLOSE	*Annotation elliptical arc close*
-14	PUGDP_IMAGE	*Image*
-15	PUGDP_RECT_GRID	*Rectangular grid*
-16	PUGDP_RAD_GRID	*Radial grid*
-17	PUGDP_MONO_ENC_TXT	*Mono-encoded text*
-18	PUGDP_MONO_ENC_ATXT_REL	*Mono-encoded annotation text relative*

gdp_data
A pointer to a Pgdp_data union containing the information needed to perform the function specified by *gdp_id.* Pgdp_data is defined in phigs.h as:

```
typedef union {
        struct Pgdp_r1 {
                Pint            unused;
        } gdp_r1;
        Pdata   unsupp;         /∗ unsupp GDP data record∗/
                                /∗ implementation dependent    ∗/
        struct Pgdp_u2 {
                Ppoint          center;         /∗ center point ∗/
                Pfloat          radius;         /∗ radius ∗/
        } gdp_u2;                               /∗ circle ∗/
```

```
struct Pgdp_u3 {
        Ppoint      center;       /* center point */
        Pfloat      radius;       /* radius */
        Pfloat      st_ang;       /* start angle in radian */
        Pfloat      end_ang;      /* end angle in radian */
} gdp_u3;                         /* circular arc */
struct Pgdp_u4 {
        Ppoint      center;       /* center point */
        Pfloat      radius;       /* radius */
        Pfloat      st_ang;       /* start angle */
        Pfloat      end_ang;      /* end angle */
        Pint        cls_type;     /* close type */
} gdp_u4;                         /* circular arc close */
struct Pgdp_u5 {
        Ppoint      center;       /* center point */
        Pfloat      radius;       /* radius */
} gdp_u5;                         /* annotation circle */
struct Pgdp_u6 {
        Ppoint      center;       /* center point */
        Pfloat      radius;       /* radius */
        Pfloat      st_ang;       /* start angle in radian */
        Pfloat      end_ang;      /* end angle in radian */
} gdp_u6;                         /* annotation circular arc */
struct Pgdp_u7 {
        Ppoint      center;       /* center point */
        Pfloat      radius;       /* radius */
        Pfloat      st_ang;       /* start angle */
        Pfloat      end_ang;      /* end angle */
        Pint        cls_type;     /* close type */
} gdp_u7;                         /* annotation circular arc close */
struct Pgdp_u8 {
        Ppoint      center;       /* center point */
        Pvec        major;        /* major axis */
        Pvec        minor;        /* minor axis */
} gdp_u8;                         /* ellipse */
struct Pgdp_u9 {
        Ppoint      center;       /* center point */
        Pvec        major;        /* major axis */
        Pvec        minor;        /* minor axis */
        Pfloat      st_ang;       /* start angle */
        Pfloat      end_ang;      /* end angle */
} gdp_u9;                         /* elliptical arc */
struct Pgdp_u10 {
```

```
            Ppoint      center;     /∗ center point ∗/
            Pvec        major;      /∗ major axis ∗/
            Pvec        minor;      /∗ minor axis ∗/
            Pfloat      st_ang;     /∗ start angle ∗/
            Pfloat      end_ang;    /∗ end angle ∗/
            Pint        cls_type;   /∗ close type ∗/
    } gdp_u10;                      /∗ elliptical arc close ∗/
    struct Pgdp_u11 {
            Ppoint      center;     /∗ center point ∗/
            Pvec        major;      /∗ major axis ∗/
            Pvec        minor;      /∗ minor axis ∗/
    } gdp_u11;                      /∗ annotation ellipse ∗/
    struct Pgdp_u12 {
            Ppoint      center;     /∗ center point ∗/
            Pvec        major;      /∗ major axis ∗/
            Pvec        minor;      /∗ minor axis ∗/
            Pfloat      st_ang;     /∗ start angle ∗/
            Pfloat      end_ang;    /∗ end angle ∗/
    } gdp_u12;                      /∗ annotation elliptical arc ∗/
    struct Pgdp_u13 {
            Ppoint      center;     /∗ center point ∗/
            Pvec        major;      /∗ major axis ∗/
            Pvec        minor;      /∗ minor axis ∗/
            Pfloat      st_ang;     /∗ start angle ∗/
            Pfloat      end_ang;    /∗ end angle ∗/
            Pint        cls_type;   /∗ close type ∗/
    } gdp_u13;                      /∗ annotation elliptical arc close ∗/
    struct Pgdp_u14 {
            Ppoint      ref_pt;     /∗ Reference point in MC ∗/
            Pint_size   dim;        /∗ Device space dimensions of
                                       image ∗/
            Pint        colr_type;  /∗ PINDIRECT, PRGB, PCIE, PHSV, PHLS,
                                       and so on ∗/
            union {
                        char        ∗chars;     /∗ 8-bit indices ∗/
                        Pint        ∗ints;      /∗ 32-bit XBGR ∗/
            } colrs;
    } gdp_u14;
    struct Pgdp_u15 {
            Ppoint      origin;     /∗ center point ∗/
            Pfloat      dx;         /∗ spacing along the x axis ∗/
            Pfloat      dy;         /∗ spacing along the y axis ∗/
    } gdp_u15;
```

```
        struct Pgdp_u16 {
                Ppoint      center;     /* center point */
                Pfloat      d_angle;    /* spacing angle between rays */
                Pfloat      d_dist;     /* spacing between points on a
                                           ray */
        } gdp_u16;
        struct Pgdp_u17 {
                Ppoint              text_pt;     /* text point */
                Phigs_i_string      i_string;    /* I-String */
        } gdp_u17;
        struct Pgdp_u18 {
                Ppoint              ref_pt;      /* reference point */
                Pvec                anno_offset; /* annotation offset */
                Phigs_i_string      i_string;    /* annotation I-String */
        } gdp_u18;
} Pgdp_data;
```

FORTRAN Input Parameters

N Number of points passed in the *PXA* and *PYA* arrays.

PXA An array of *N* real values containing *x* coordinates in MC.

PYA An array of *N* real values containing *y* coordinates in MC.

PRIMID
An integer specifying the GENERALIZED DRAWING PRIMITIVE to be performed. Predefined values defined in phigs77.h are:

-2	PUGDPCIRC	*Circle*
-3	PUGDPCIRCARC	*Circular arc*
-4	PUGDPCIRCARCCL	*Circular arc close*
-5	PUGDPACIRC	*Annotation circle*
-6	PUGDPACIRCARC	*Annotation circular arc*
-7	PUGDPACIRCARCCL	*Annotation circular arc close*
-8	PUGDPELLP	*Ellipse*
-9	PUGDPELLPARC	*Elliptical arc*
-10	PUGDPELLPARCCL	*Elliptical arc close*
-11	PUGDPAELLP	*Annotation ellipse*
-12	PUGDPAELLPARC	*Annotation elliptical arc*
-13	PUGDPAELLPARCCL	*Annotation elliptical arc close*
-14	PUGDPIMG	*Image*
-15	PUGDPRECTGRID	*Rectangular grid*
-16	PUGDPRADGRID	*Radial grid*
-17	PUGDPMONOENCTXT	*Mono-encoded text*
-18	PUGDPMONOENCATXTREL	*Mono-encoded annotation text relative*

LDR The dimension of *DATREC* (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by *PRIMID*.

Execution Circle, Circular Arc, Circular Arc Close, Ellipse, Elliptical Arc, Elliptical Arc Close are rendered according to type 1 curve approximation criteria (Constant Parametric Subdivision). A default curve approximation criteria type 1, and approximation value of 36.0 will be used if curve approximation criteria other than type 1 or curve approximation value of less than 36.0 is specified.

Image, mono-encoded text, and mono-encoded annotation text relative are not affected by curve approximation criteria.

ERRORS 005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

PACK DATA RECORD (3P)
GENERALIZED DRAWING PRIMITIVE -2 (3P)
GENERALIZED DRAWING PRIMITIVE -3 (3P)
GENERALIZED DRAWING PRIMITIVE -4 (3P)
GENERALIZED DRAWING PRIMITIVE -5 (3P)
GENERALIZED DRAWING PRIMITIVE -6 (3P)
GENERALIZED DRAWING PRIMITIVE -7 (3P)
GENERALIZED DRAWING PRIMITIVE -8 (3P)
GENERALIZED DRAWING PRIMITIVE -9 (3P)
GENERALIZED DRAWING PRIMITIVE -10 (3P)
GENERALIZED DRAWING PRIMITIVE -11 (3P)
GENERALIZED DRAWING PRIMITIVE -12 (3P)
GENERALIZED DRAWING PRIMITIVE -13 (3P)
GENERALIZED DRAWING PRIMITIVE -14 (3P)
GENERALIZED DRAWING PRIMITIVE -15 (3P)
GENERALIZED DRAWING PRIMITIVE -16 (3P)
GENERALIZED DRAWING PRIMITIVE -17 (3P)
GENERALIZED DRAWING PRIMITIVE -18 (3P)
GENERALIZED DRAWING PRIMITIVE 3 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE -2 – create 2D GDP circle

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                number of points (>= 0)
REAL             PXA(*), PYA(*)   coordinates of points (MC)
INTEGER          PRIMID           GDP identifier
INTEGER          LDR              dimension of data record array
CHARACTER*80     DATREC(LDR)      data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -2 creates the circle GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_CIRCLE, which is defined in phigs.h to be -2.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u2, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u2 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u2 {
                Ppoint   center;    /* center point */
                Pfloat   radius;    /* radius */
        } gdp_u2;           /* circle */
} Pgdp_data;
```

The *center* component specifies the center of the circle. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat     x;      /* x coordinate */
        Pfloat     y;      /* y coordinate */
} Ppoint;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRC, which is defined in phigs77.h to be -2.

LDR The dimension of *DATREC* (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRC should be as follows:

IL The number of integers = 0.

RL The number of real values = 3.

RA Contains array of RL real values.
Element 1 is the *x* coordinate of the center of the circle in MC.
Element 2 is the *y* coordinate of the center of the circle in MC.
Element 3 is the radius of the circle in MC.

SL The number of strings = 0.

Execution

A circle is a planar area defined by a center point and a radius. The Circle Local Coordinate (CLC) system is located at the center point in MC on the $z = 0$ plane. The horizontal and vertical axes of CLC are parallel to the *x* and *y* axes of the MC.

The center point and radius are specified in MC. At traversal, the circle in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted.

The current values of all fill area set attributes will be applied to the circle when it is drawn.

The circle is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circle GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -2 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Interior Attributes* in *Index*
See *Edge Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -3 – create 2D GDP circular arc

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list     *point_list;     array of points
Pint            gdp_id;          gdp function identifier
Pgdp_data       *gdp_data;       data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER         N                number of points (>= 0)
REAL            PXA(*), PYA(*)   coordinates of points (MC)
INTEGER         PRIMID           GDP identifier
INTEGER         LDR              dimension of data record array
CHARACTER*80    DATREC(LDR)      data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -3 creates the circular arc GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_CIRC_ARC, which is defined in phigs.h to be -3.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u3, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u3 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u3 {
                Ppoint   center;      /* center point */
                Pfloat   radius;      /* radius */
                Pfloat   st_ang;      /* start angle in radians */
                Pfloat   end_ang;     /* end angle in radians */
        } gdp_u3;          /* circular arc */
} Pgdp_data;
```

The *center* component specifies the center of the circular arc. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
} Ppoint;
```

The *radius* component specifies the radius of the circular arc in MC. The *st_ang* component specifies the start angle of the circular arc in radians. The *end_ang* component specifies the end angle of the circular arc in radians. The start and end angles are specified in a counter-clockwise direction with the MC positive *x* axis as the origin.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRCARC, which is defined in phigs77.h to be -3.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRCARC should be as follows:

IL The number of integers = 0.

RL The number of real values = 5.

RA Contains array of RL real values.
Element 1 is the *x* coordinate of the center of the circular arc in MC.
Element 2 is the *y* coordinate of the center of the circular arc in MC.
Element 3 is the radius of the circular arc in MC.
Element 4 is the start angle of the circular arc in radians.
Element 5 is the end angle of the circular arc in radians.

SL The number of strings = 0.

Execution

A circular arc is a curve defined by a center point, a radius, and a start and end angle. The Circle Local Coordinate (CLC) system is located at the center point in MC on the *z* = 0 plane. The horizontal and vertical axes of CLC are parallel to the *x* and *y* axes of the MC.

The center point and radius are specified in MC. The start and end angles are specified in radians in counter-clockwise direction with the CLC positive *x* axis as the origin. At traversal, the circular arc in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. A circular arc greater than 360 degrees is drawn as a hollow circle.

The current values of all polyline attributes will be applied to the circular arc when it is drawn.

The circular arc is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circular arc GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -3 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline shading method	polyline shading method ASF
curve approximation criteria	curve approximation criteria ASF
polyline index	
depth cue index	
name set	

ERRORS

005　Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -4 – create 2D GDP circular arc close

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                 number of points (>= 0)
REAL             PXA(*), PYA(*)    coordinates of points (MC)
INTEGER          PRIMID            GDP identifier
INTEGER          LDR               dimension of data record array
CHARACTER*80     DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -4 creates the circular arc close GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_CIRC_ARC_CLOSE, which is defined in phigs.h to be -4.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u4, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u4 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u4 {
                Ppoint   center;      /* center point */
                Pfloat   radius;      /* radius */
                Pfloat   st_ang;      /* start angle in radians */
                Pfloat   end_ang;     /* end angle in radians */
                Pint     cls_type;    /* close type */
        } gdp_u4;           /* circle arc close */
} Pgdp_data;
```

The *center* component specifies the center of the circular arc close. It is a Ppoint

structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;       /* x coordinate */
        Pfloat      y;       /* y coordinate */
} Ppoint;
```

The *radius* component specifies the radius of the circular arc close in MC. The *st_ang* component specifies the start angle of the circular arc close in radians. The *end_ang* component specifies the end angle of the circular arc close in radians. The start and end angles are specified in a counterclockwise direction with the MC positive *x* axis as the origin. The *cls_type* component is either PHIGS_ARC_CLOSE_PIE or PHIGS_ARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the circular arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the circular arc and the chord from starting point to the ending point.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRCARCCL, which is defined in phigs77.h to be -4.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRCARCCL should be as follows:

IL The number of integers = 1.

IA Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).

RL The number of real values = 5.

RA Contains array of RL real values.
Element 1 is the *x* coordinate of the center of the circular arc in MC.
Element 2 is the *y* coordinate of the center of the circular arc in MC.
Element 3 is the radius of the circular arc in MC.
Element 4 is the start angle of the circular arc in radians.
Element 5 is the end angle of the circular arc in radians.

SL The number of strings = 0.

Execution

A circular arc close is a planar area defined by a center point, a radius, and a start and end angle. The Circle Local Coordinate (CLC) system is located at the center point in MC on the *z* = 0 plane. The horizontal and vertical axes of CLC are parallel to the *x* and *y* axes of the MC.

The center point and radius are specified in MC. The start and end angles are specified in radians in counter-clockwise direction with the CLC positive *x* axis as the origin. At traversal, the circular arc close in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. A circular arc close greater than 360 degrees is drawn as a filled circle.

The current values of all fill area set attributes will be applied to the circular arc close when it is drawn.

The circular arc close is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circular arc close GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -4 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	

face culling mode
depth cue index
light source state
name set

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Interior Attributes* in *Index*
See *Edge Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -5 – create 2D GDP annotation circle

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                  number of points (>= 0)
REAL             PXA(*), PYA(*)     coordinates of points (MC)
INTEGER          PRIMID             GDP identifier
INTEGER          LDR                dimension of data record array
CHARACTER*80     DATREC(LDR)        data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -5 creates the annotation circle GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRCLE, which is defined in phigs.h to be -5.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u5, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u5 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u5 {
                Ppoint   center;    /* center point */
                Pfloat   radius;    /* radius */
        } gdp_u5;            /* annotation circle */
} Pgdp_data;
```

The *center* component specifies the center of the annotation circle. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
} Ppoint;
```

The *radius* component specifies the radius of the annotation circle in NPC.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRC, which is defined in phigs77.h to be -5.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRC should be as follows:

IL The number of integers = 0.

RL The number of real values = 3.

RA Contains an arrya of RL real values.
Element 1 is the *x* coordinate of the center of the annotation circle in MC.
Element 2 is the *y* coordinate of the center of the annotation circle in MC.
Element 3 is the radius of the annotation circle in NPC.

SL The number of strings = 0.

Execution

An annotation circle is a planar area defined by a center point and a radius. The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point on the $z = 0$ plane and the *x* and *y* axes parallel to, and in the same direction as, the NPC *x* and *y* axes.

The center point of the annotation circle is specified in MC. The radius of the annotation circle is specified in CLC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation circle.

The current values of all fill area set attributes will be applied to the annotation circle when it is drawn.

Annotation circle is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circle GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -5 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -6 – create 2D GDP annotation circular arc

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER         N                 number of points (>= 0)
REAL            PXA(*), PYA(*)    coordinates of points (MC)
INTEGER         PRIMID            GDP identifier
INTEGER         LDR               dimension of data record array
CHARACTER*80    DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -6 creates the annotation circular arc structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRC_ARC, which is defined in phigs.h to be -6.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u6, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u6 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u6 {
                Ppoint   center;     /* center point */
                Pfloat   radius;     /* radius */
                Pfloat   st_ang;     /* start angle in radians */
                Pfloat   end_ang;    /* end angle in radians */
        } gdp_u6;          /* annotation circular arc */
} Pgdp_data;
```

The *center* component specifies the center of the annotation circular arc. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
} Ppoint;
```

The *radius* component specifies the radius of the annotation circular arc in NPC. The *st_ang* component specifies the start angle of the annotation circular arc in radians. The *end_ang* component specifies the end angle of the annotation circular arc in radians. The start and end angles are specified in a counterclockwise direction with the positive *x* axis as the origin.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRCARC, which is defined in phigs77.h to be -6.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRCARC should be as follows:

IL The number of integers = 0.

RL The number of real values = 5.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation circular arc in MC.

Element 2 is the *y* coordinate of the center of the annotation circular arc in MC.

Element 3 is the radius of the annotation circular arc in NPC.

Element 4 is the start angle of the annotation circular arc in radians.

Element 5 is the end angle of the annotation circular arc in radians.

SL The number of strings = 0.

Execution

An annotation circular arc is a curve defined by a center point, a radius, and a start angle and end angle.

The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point on the $z = 0$ plane and the x and y axes parallel to, and in the same direction as, the NPC x and y axes.

The center point of the annotation circular arc is specified in MC. The radius of the annotation circular arc is specified in CLC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation circular arc. An annotation circular arc greater than 360 degrees is drawn as a hollow annotation circle.

The current values of all polyline attributes will be applied to the annotation circular arc when it is drawn.

Annotation circular arc is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circular arc GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -6 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -7 – create 2D GDP annotation circular arc close

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                 number of points (>= 0)
REAL             PXA(*), PYA(*)    coordinates of points (MC)
INTEGER          PRIMID            GDP identifier
INTEGER          LDR               dimension of data record array
CHARACTER*80     DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -7 creates the annotation circular arc close structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRC_ARC_CLOSE, which is defined in phigs.h to be -7.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u7, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u7 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u7 {
                Ppoint   center;     /* center point */
                Pfloat   radius;     /* radius */
                Pfloat   st_ang;     /* start angle in radians */
                Pfloat   end_ang;    /* end angle in radians*/
                Pint     cls_type;   /* close type */
        } gdp_u7;        /* annotation circular arc close */
} Pgdp_data;
```

The *center* component specifies the center of the annotation circular arc close. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
} Ppoint;
```

The *radius* component specifies the radius of the annotation circular arc close in NPC. The *st_ang* component specifies the start angle of the annotation circular arc close in radians. The *end_ang* component specifies the end angle of the annotation circular arc close in radians. The start and end angles are specified in a counterclockwise direction with the positive *x* axis as the origin. The *cls_type* component is either PHIGS_ARC_CLOSE_PIE or PHIGS_ARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the circular arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the circular arc and the chord from starting point to the ending point.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRCARCCL, which is defined in phigs77.h to be -7.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRCARCCL should be as follows:

IL The number of integers = 1.

IA Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).

RL The number of real values = 5.

RA Contains array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation circular arc close in MC.

Element 2 is the *y* coordinate of the center of the annotation circular arc close in MC.

Element 3 is the radius of the annotation circular arc in NPC.

Element 4 is the start angle of the annotation circular arc close in radians.

Element 5 is the end angle of the annotation circular arc close in radians.

SL The number of strings = 0.

Execution

An annotation circular arc close is a planar area defined by a center point, a radius, a start angle, and an end angle. The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point on the *z* = 0 plane and the *x* and *y* axes parallel to, and in the same direction as, the NPC *x* and *y* axes.

The center point of the annotation circular arc close is specified in MC. The radius of the annotation circular arc close is specified in CLC. At traversal, all transformations affects the center point, but only the workstation transformation affects the other parameters of the annotation circular arc. An annotation circular arc close greater than 360 degrees is drawn as a filled annotation circle.

The current values of all fill area set attributes will be applied to the annotation circular arc close when it is drawn.

Annotation circular arc close is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circular arc close GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -7 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -8 – create 2D GDP ellipse

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER         N                 number of points (>= 0)
REAL            PXA(*), PYA(*)    coordinates of points (MC)
INTEGER         PRIMID            GDP identifier
INTEGER         LDR               dimension of data record array
CHARACTER*80    DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -8 creates the ellipse element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ELLIPSE, which is defined in phigs.h to be -8.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u8, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u8 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u8 {
                Ppoint    center;    /* center point */
                Pvec      major;     /* major axis */
                Pvec      minor;     /* minor axis */
        } gdp_u8;              /* ellipse */
} Pgdp_u8;
```

The *center* component specifies the center of the ellipse. It is a Ppoint structure containing *x* and *y* coordinates specified in MC.
Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. It is a Pvec structure defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
} Pvec;
```

The *minor* component specifies the minor axis of the ellipse. The major and minor axes of the ellipse are specified in MC. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLP, which is defined in phigs77.h to be -8.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPELLP should be as follows:

IL The number of integers = 0.

RL The number of real values = 6.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the center of the ellipse in MC.
Element 2 is the *y* coordinate of the center of the ellipse in MC.
Element 3 is the *x* component of the ellipse major axis in MC.
Element 4 is the *y* component of the ellipse major axis in MC.
Element 5 is the *x* component of the ellipse minor axis in MC.
Element 6 is the *y* component of the ellipse minor axis in MC.

SL The number of strings = 0.

Execution

An ellipse is a planar area defined by a center point, a major axis, and a minor axis. The Ellipse Local Coordinate (ELC) system is located at the center point on the $z = 0$ plane. The horizontal and vertical axes of ELC are defined by the major and minor axes. Let P be the center point, and U and V be the major and minor axes, respectively. The plane of the ellipse is the $z = 0$ plane in the MC system. The origin of the ELC system is at the point P. The positive x axis of the ELC system is parallel to U. Let T be a vector perpendicular to U and in the ellipse plane. If T.V is positive, the positive y axis is parallel to T; otherwise, positive y axis is opposite to T. In the event that the magnitudes of both axes are zero, the ellipse is displayed as a dot (.) at the center point. In the event that the magnitude of only one of the axes is zero, the ellipse is displayed as a line of correct magnitude along the other axis. If the major and minor axes fail to define the local coordinate system (that is, the vectors are parallel and non-zero), then a line, which is twice the length of the major axis, parallel to the major axis, and centered at the origin of the ellipse, is used to display the ellipse.

The center point and major and minor axes are specified in MC. At traversal, the ellipse in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted.

The current values of all fill area set attributes will be applied to the ellipse when it is drawn.

The ellipse is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the ellipse GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -8 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	

curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -9 – create 2D GDP elliptical arc

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                 number of points (>= 0)
REAL             PXA(*), PYA(*)    coordinates of points (MC)
INTEGER          PRIMID            GDP identifier
INTEGER          LDR               dimension of data record array
CHARACTER*80     DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -9 creates the elliptical arc element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ELLP_ARC, which is defined in phigs.h to be -9.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u9, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u9 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u9 {
                Ppoint   center;      /* center point */
                Pvec     major;       /* major axis */
                Pvec     minor;       /* minor axis */
                Pfloat   st_ang;      /* start angle */
                Pfloat   end_ang;     /* end angle */
        } gdp_u9;          /* elliptical arc */
} Pgdp_data;
```

The *center* component specifies the center of the elliptical arc. It is a Ppoint

structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;       /* x coordinate */
        Pfloat      y;       /* y coordinate */
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. The *minor* component specifies the minor axis. The *st_ang* component specifies the start angle of the elliptical arc. The *end_ang* component specifies the end angle of the elliptical arc. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes. The angles are specified in radians in a counterclockwise direction with the positive *x* axis as the origin.

Ppoint is defined above. The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLPARC, which is defined in phigs77.h to be -9.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPELLPARC should be as follows:

IL The number of integers = 0.

RL The number of real values = 8.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the center of the elliptical arc in MC.
Element 2 is the *y* coordinate of the center of the elliptical arc in MC.
Element 3 is the *x* component of the ellipse major axis in MC.
Element 4 is the *y* component of the ellipse major axis in MC.
Element 5 is the *x* component of the ellipse minor axis in MC.

Element 6 is the *y* component of the ellipse minor axis in MC.

Element 7 is the start angle of the elliptical arc in radians.

Element 8 is the end angle of the elliptical arc in radians.

SL The number of strings = 0.

Execution

An elliptical arc is a curve defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The Ellipse Local Coordinate (ELC) system is located at the center point on the $z = 0$ plane. The horizontal and vertical axes of ELC are defined by the major and minor axes respectively.

The center point and major and minor axes of the elliptical arc are specified in MC. The start and end angle are specified in radians in a counter-clockwise direction with the ELC positive *x* axis as the origin. At traversal, the circular arc in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. An elliptical arc greater than 360 degrees is drawn as a hollow ellipse.

The current values of all polyline attributes will be applied to the elliptical arc when it is drawn.

The elliptical arc is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the elliptical arc GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -9 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline shading method	polyline shading method ASF
curve approximation criteria	curve approximation criteria ASF
polyline index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP), *, STOP), *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -10 – create a 2D GDP elliptical arc close

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                number of points (>= 0)
REAL             PXA(*), PYA(*)   coordinates of points (MC)
INTEGER          PRIMID           GDP identifier
INTEGER          LDR              dimension of data record array
CHARACTER*80     DATREC(LDR)      data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -10 creates the elliptical arc close element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ELLP_ARC_CLOSE, which is defined in phigs.h to be -10.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u10, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u10 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u10 {
                Ppoint   center;      /* center point */
                Pvec     major;       /* major axis */
                Pvec     minor;       /* minor axis */
                Pfloat   st_ang;      /* start angle */
                Pfloat   end_ang;     /* end angle */
                Pint     cls_type;    /* close type */
        } gdp_u10;          /* elliptical arc close */
} Pgdp_data;
```

The *center* component specifies the center of the elliptical arc close. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. The *minor* component specifies the minor axis. The *st_ang* component specifies the start angle of the elliptical arc close. The *end_ang* component specifies the end angle of the elliptical arc close. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes. The angles are specified in radians in a counterclockwise direction with the ELC positive *x* axis as the origin. The *cls_type* component is either PHIGS_ARC_CLOSE_PIE or PHIGS_ARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the elliptical arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the elliptical arc and the chord from starting point to the ending point.

Ppoint is defined above. The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /∗ x magnitude ∗/
        Pfloat   delta_y;   /∗ y magnitude ∗/
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLPARCCL, which is defined in phigs77.h to be -10.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPELLPARCCL should be as follows:

IL The number of integers = 1.

IA Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).

RL The number of real values = 8.

RA	Contains array of RL real values.
	Element 1 is the *x* coordinate of the center of the elliptical arc close in MC.
	Element 2 is the *y* coordinate of the center of the elliptical arc close in MC.
	Element 3 is the *x* component of the ellipse major axis in MC.
	Element 4 is the *y* component of the ellipse major axis in MC.
	Element 5 is the *x* component of the ellipse minor axis in MC.
	Element 6 is the *y* component of the ellipse minor axis in MC.
	Element 7 is the start angle of the elliptical arc close in radians.
	Element 8 is the end angle of the elliptical arc close in radians.
SL	The number of strings = 0.

Execution

An elliptical arc close is a planar area defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The ELC system is located at the center point on the $z = 0$ plane. The horizontal and vertical axes of ELC are defined by the major and minor axes respectively.

The center point and major and minor axes of the elliptical arc close are specified in MC. The start and end angles are specified in radians in counter-clockwise direction with the ELC positive *x* axis as the origin. At traversal, the circular arc in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. An elliptical arc close greater than 360 degrees is drawn as a filled ellipse.

The current values of all fill area attributes will be applied to the elliptical arc close when it is drawn.

The elliptical arc close is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the elliptical arc close GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -10 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF

back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005　Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -11 – create 2D GDP annotation ellipse

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                  number of points (>= 0)
REAL             PXA(*), PYA(*)     coordinates of points (MC)
INTEGER          PRIMID             GDP identifier
INTEGER          LDR                dimension of data record array
CHARACTER*80     DATREC(LDR)        data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -11 creates the annotation ellipse element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLIPSE, which is defined in phigs.h to be -11.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u11, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u11 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u11 {
                Ppoint   center;    /* center point */
                Pvec     major;     /* major axis */
                Pvec     minor;     /* minor axis */
        } gdp_u11;          /* annotation ellipse */
} Pgdp_data;
```

The *center* component specifies the center of the annotation ellipse. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. The *minor* component specifies the minor axis of the annotation ellipse. The *x* and *y* magnitudes of the major and minor axes of the annotation ellipse are specified in NPC.

The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /∗ x magnitude ∗/
        Pfloat   delta_y;   /∗ y magnitude ∗/
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPAELLP, which is defined in phigs77.h to be -11.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPAELLP should be as follows:

IL The number of integers = 0.

RL The number of real values = 6.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the center of the annotation ellipse in MC.
Element 2 is the *y* coordinate of the center of the annotation ellipse in MC.
Element 3 is the *x* component of the annotation ellipse major axis in MC.
Element 4 is the *y* component of the annotation ellipse major axis in NPC.
Element 5 is the *x* component of the annotation ellipse minor axis in NPC.
Element 6 is the *y* component of the annotation ellipse minor axis in NPC.

SL The number of strings = 0

Execution

An annotation ellipse is a planar area defined by a center point, a major axis, and a minor axis. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point on the *z* = 0 plane and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes. The horizontal and vertical axes of ELC are defined by the major and minor axes respectively.

The center point of the annotation ellipse is specified in MC. The major and minor axes of the annotation ellipse are specified in ELC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation ellipse.

The current values of all fill area set attributes will be applied to the annotation ellipse when it is drawn.

Annotation ellipse is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation ellipse GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -11 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -12 – create a 2D GDP annotation elliptical arc

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                 number of points (>= 0)
REAL             PXA(*), PYA(*)    coordinates of points (MC)
INTEGER          PRIMID            GDP identifier
INTEGER          LDR               dimension of data record array
CHARACTER*80     DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -12 creates the annotation elliptical arc element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLP_ARC, which is defined in phigs.h to be -12.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u12, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u12 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u12 {
                Ppoint   center;      /* center point */
                Pvec     major;       /* major axis */
                Pvec     minor;       /* minor axis */
                Pfloat   st_ang;      /* start angle */
                Pfloat   end_ang;     /* end angle */
        } gdp_u12;          /* annotation elliptical arc */
} Pgdp_data;
```

The *center* component specifies the center of the annotation elliptical arc. It is a

Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat     x;      /* x coordinate */
        Pfloat     y;      /* y coordinate */
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. The *minor* component specifies the minor axis. The major and minor axes of the annotation ellipse are specified in NPC. The *st_ang* component specifies the start angle of the elliptical arc. The *end_ang* component specifies the end angle of the elliptical arc. The angles are specified in radians in a counterclockwise direction with the ellipse local positive *x* axis as the origin.

The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat  delta_x;   /* x magnitude */
        Pfloat  delta_y;   /* y magnitude */
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPAELLPARC, which is defined in phigs77.h to be -12.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPAELLPARC should be as follows:

IL The number of integers = 0.

RL The number of real values = 8.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation elliptical arc in MC.

Element 2 is the *y* coordinate of the center of the annotation elliptical arc in MC.

Element 3 is the *x* component of the annotation elliptical arc major axis in NPC.

Element 4 is the *y* component of the annotation elliptical arc major axis in NPC.

Element 5 is the *x* component of the annotation elliptical arc minor axis in NPC.

Element 6 is the *y* component of the annotation elliptical arc minor axis in NPC.

Element 7 is the start angle of the annotation elliptical arc in radians.

Element 8 is the end angle of the annotation elliptical arc in radians.

SL The number of strings = 0.

Execution

An annotation elliptical arc is a curve defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point on the $z = 0$ plane and the *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes. The horizontal and vertical axes of ELC are defined by the major and minor axes respectively.

The center point of the annotation elliptical arc is specified in MC. The major and minor axes of the annotation elliptical arc are specified in ELC. The start and end angles are specified in radians with the ELC positive *x* axis as the origin. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation elliptical arc. An annotation elliptical arc greater than 360 degrees is drawn as a hollow annotation ellipse.

The current values of all polyline attributes will be applied to the annotation elliptical arc when it is drawn.

Annotation elliptical arc is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation elliptical arc GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -12 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -13 – create 2D GDP annotation elliptical arc close

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                 number of points (>= 0)
REAL             PXA(*), PYA(*)    coordinates of points (MC)
INTEGER          PRIMID            GDP identifier
INTEGER          LDR               dimension of data record array
CHARACTER*80     DATREC(LDR)       data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -13 creates the annotation elliptical arc close element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLP_ARC_CLOSE, which is defined in phigs.h to be -13.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u13, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u13 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u13 {
                Ppoint   center;      /* center point */
                Pvec     major;       /* major axis */
                Pvec     minor;       /* minor axis */
                Pfloat   st_ang;      /* start angle */
                Pfloat   end_ang;     /* end angle */
                Pint     cls_type;    /* close type */
        } gdp_u13;        /* annotation elliptical arc close */
} Pgdp_data;
```

The *center* component specifies the center of the annotation elliptical arc close. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
} Ppoint;
```

The *major* component specifies the length and direction of the major axis relative to the center point. It is a Pvec structure defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
} Pvec;
```

The *minor* component specifies the minor axis. The the major and minor axes of the annotation ellipse are specified in NPC. The st_ang component specifies the start angle of the elliptical arc close. The *end_ang* component specifies the end angle of the elliptical arc close. The angles are specified in radians in a counterclockwise direction with the ellipse local positive *x* axis as the origin. The *cls_type* component is either PHIGS_ARC_CLOSE_PIE or PHIGS_ARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the elliptical arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the elliptical arc and the chord from the starting point to the ending point.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPAELLPARCCL, which is defined in phigs77.h to be -13.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPAELLPARCCL should be as follows:

IL The number of integers = 1.

IA Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).

RL The number of real values = 8.

RA	Contains an array of RL real values.
	Element 1 is the *x* coordinate of the center of the annotation elliptical arc close in MC.
	Element 2 is the *y* coordinate of the center of the annotation elliptical arc close in MC.
	Element 3 is the *x* component of the annotation elliptical arc close major axis in NPC.
	Element 4 is the *y* component of the annotation elliptical arc close major axis in NPC.
	Element 5 is the *x* component of the annotation elliptical arc close minor axis in NPC.
	Element 6 is the *y* component of the annotation elliptical arc close minor axis in NPC.
	Element 7 is the start angle of the annotation elliptical arc close in radians.
	Element 8 is the end angle of the annotation elliptical arc close in radians.
SL	The number of strings = 0.

Execution

An annotation elliptical arc close is a planar area defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point on the *z* = 0 plane and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes. The horizontal and vertical axes of ELC are defined by the major and minor axes respectively.

The center point of the annotation elliptical arc is specified in MC. The major and minor axes of the annotation elliptical arc close is specified in ELC. The start and end angles are specified in radians with the NPC positive *x* axis as the origin. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation elliptical arc close. An annotation elliptical arc close greater than 360 degrees is drawn as a filled annotation ellipse.

The current values of all fill area set attributes will be applied to the annotation elliptical arc close when it is drawn.

Annotation elliptical close is *not* affected by the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation elliptical arc close GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -13 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE -14 – create a 2D GDP image

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                  number of points (>= 0)
REAL             PXA(*), PYA(*)     coordinates of points (MC)
INTEGER          PRIMID             GDP identifier
INTEGER          LDR                dimension of data record array
CHARACTER*80     DATREC(LDR)        data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -14 creates the image structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_IMAGE, which is defined in phigs.h to be -14.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u14, containing the information needed to perform the function specified by gdp_id. Pgdp_u14 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u14 {
                Ppoint        ref_pt;        /* reference point in MC */
                Pint_size     dim;           /* dimensions of array */
                Pint          colr_type;     /* indirect, RGB, CIE, HSV, HLS */
                union {
                              char           *chars;     /* 8-bit indices */
                              Pint           *ints;      /* 32-bit XBGR */
                } colrs;
        } gdp_u14;              /* image */
} Pgdp_data;
```

ref_pt The *ref_pt* component specifies the top left corner of the image. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat    x;      /* x coordinate */
        Pfloat    y;      /* y coordinate */
} Ppoint;
```

dim *dim* is a pointer to a Pint_size structure specifying the dimensions of the image in pixels. The Pint_size structure is defined in phigs.h as follows:

```
typedef struct {
        Pint   size_x;   /* dimension along x */
        Pint   size_y;   /* dimension along y */
} Pint_size;
```

colr_type
The *colr_type* specifies the colour type used by the image.

0	PINDIRECT	*Colour index specified*
1	PMODEL_RGB	*Red, green, and blue*
2	PMODEL_CIELUV	*CIE colour type*
3	PMODEL_HSV	*Hue, saturation, and value*
4	PMODEL_HLS	*Hue, lightness, and saturation*

Only PINDIRECT and PMODEL_RGB are currently supported.

colrs *colrs* is a union of pointers to the appropriate image buffer, depending on *colr_type*. The image colour data must be organized scanline by scanline. If *colr_type* is PINDIRECT, *colrs*→*chars* points to the 8-bit indexed colour data. In this case, each scanline must be padded to a 32-bit boundary. If *colr_type* is PMODEL_RGB, *colrs*→*ints* points to the 32-bit XBGR colour data. True colour 32-bit integers are packed with the true colour triplet in the lower 24 bits; that is, XBGR. Each of B, G, and R are an 8-bit index in the range of 0 to 255.

Example For example, to display a 24-bit 400 × 400 image in a PHIGS workstation, you could do the following:

```
load_image ( x, y, xbgr_data )
Pfloat  x, y;
Pint    *xbgr_data;
{
        Pgdp_data      image_gdp;

        image_gdp.gdp_u14.ref_pt.x = x;
        image_gdp.gdp_u14.ref_pt.y = y;
        image_gdp.gdp_u14.dim.size_x = 400;
        image_gdp.gdp_u14.dim.size_y = 400;

        image_gdp.gdp_u14.colr_type = PMODEL_RGB;

        image_gdp.gdp_u14.colrs.ints = xbgr_data;

        pgdp(Ppoint_list *)NULL, PUGDP_IMAGE, &image_gdp);
}
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPIMG, which is defined in phigs77.h to be −14.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A generalized drawing primitive data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID is PUGDPIMG should be as follows:

IL The number of integers.

IA An array of integers.

The first three elements of IA are always constant.

Element 1 is the *x* dimension of the image in pixels.

Element 2 is the *y* dimension of the image in pixels.

Element 3 is the colour type. This specifies the colour type for the true colour array.

0	PINDIR	*Colour index specified*
1	PRGB	*Red, green, and blue*
2	PCIE	*CIE colour type*
3	PHSV	*Hue, saturation, and value*
4	PHLS	*Hue, lightness, and saturation*

RL The number of reals.

RA An array of reals. The first two elements of RA are always constant.

Element 1 is the *x* coordinate of the top left corner of the image in MC.

Element 2 is the *y* coordinate of the top left corner of the image in MC.

SL The number of strings.

LSTR	The length of strings.
STR	The character data contained in the specified elements.

If Element 3 of IA has the value of PINDIR, then the list of character strings specifying the image is added to STR. In this case,

IL = 3, RL = 2, SL = 1, LSTR(1) = (((IA(1)–1) / 4+1)×4)×IA(2)

Otherwise, if Element 3 of IA has the value PRGB, then the list of 32-bit integers, specifying the image, is appended to IA. In this case:

IL = 3 + IA(1)×IA(2), RL = 2, SL = 0

Execution

An image is defined by a reference point, *x* and *y* dimensions, and a series of colours. The top left corner of the image is specified by the reference point in MC. The *x* and *y* dimensions specify the width and height of the image in pixels. At traversal, all transformations only affect the reference point.

The array of points passed to this subroutine in addition to the generalized drawing primitive data record is not used by the image generalized drawing primitive.

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
GENERALIZED DRAWING PRIMITIVE 3 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -14 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE -15 – create 2D GDP rectangular grid

SYNOPSIS

C Syntax

```
void
pgdp ( points, gdp_id, gdp_data )
Ppoint_list    *point_list;   list of points
Pint           gdp_id;        gdp function identifier
Pgdp_data      *gdp_data;     data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                number of points (>= 0)
REAL             PXA(*), PYA(*)   coordinates of points (MC)
INTEGER          PRIMID           GDP identifier
INTEGER          LDR              dimension of data record array
CHARACTER*80     DATREC(LDR)      data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE -15 creates the rectangular grid element. A regular grid is a set of polymarkers uniformly spaced in the *x* and *y* dimensions, with $z = 0$.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_RECT_GRID, which is defined in phigs.h to be -15.

gdp_data
A pointer to a Pgdp_data union, pgdp_u15, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u15 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_15 {
                Ppoint      origin;     /* center point */
                Pfloat      dx;         /* spacing along the x axis */
                Pfloat      dy;         /* spacing along the y axis */
        } Pgdp_u15;                     /* rectangular grid */
} Pgdp_data;
```

The origin component specifies the center of the rectangular grid. It is a Ppoint structure containing *x* and *y* coordinates specified in MC. The *dx* and *dy* components specify the uniform spacing in the *x* and *y* dimensions, respectively, in the MC. Ppoint is defined in phigs.h as:

```
typedef struct {
        Pfloat   x;    /* x coordinate */
        Pfloat   y;    /* y coordinate */
} Ppoint;
```

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPRECTGRID, which is defined in phigs77.h to be -15.

LDR The dimension of *DATREC* (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPRECTGRID should be as follows:

IL The number of integers = 0.

RL The number of real values = 4.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the origin of the rectangular grid in MC.
Element 2 is the *y* coordinate of the origin of the rectangular grid in MC.
Element 3 is the *x* component of the rectangular grid spacing in MC.
Element 4 is the *y* component of the rectangular grid spacing in MC.

SL The number of strings = 0.

Execution

A 2D rectangular grid is a planar area defined by an origin and spacings in both *x* and *y* dimensions. The Grid Local Coordinate (GLC) system is located at the origin point on the *z* = 0 plane. The horizontal and vertical axes of the GLC are parallel to the *x* and *y* axes of the MC.

When the structure is traversed, starting from the origin, the rectangular grid GDP element places a marker at each of the GLC locations specified by *dx* and *dy*.

The coordinates used to specify the GLC are MCs. These can be any coordinate units that are convenient to the application. At traversal, these coordinate values are transformed by the current local and global modelling transformations, the view representation selected by the current view index, and the workstation transformation current on the workstation to which the structure is posted. However, since the polymarker has no geometric attributes, transformations affect only the grid position rather than the appearance of the marker glyph in which the grid is constructed.

The current values of all polymarker attributes are applied to the 2D rectangular grid.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE –15 structure element. The Aspect Source Flags (ASFs) describe where to access the output display attributes. These attributes can come directly from the traversal state list or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

marker type	marker type ASF
marker size scale factor	marker size scale factor ASF
polymarker colour index	polymarker colour index ASF
polymarker index	
polymarker colour	
depth queue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

POLYMARKER (3P)
POLYMARKER 3 (3P)
GENERALIZED DRAWING PRIMITIVE 3 –15 (3P)
GENERALIZED DRAWING PRIMITIVE 3 –16 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE -16 – create 2D GDP radial grid

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                number of points (>= 0)
REAL             PXA(*), PYA(*)   coordinates of points (MC)
INTEGER          PRIMID           GDP identifier
INTEGER          LDR              dimension of data record array
CHARACTER*80     DATREC(LDR)      data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE –16 creates the radial grid element. A radial grid is a set of polymarkers of uniform radial and angular distance from each other in the *x* and *y* dimensions with *z* = 0.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_RAD_GRID, which is defined in phigs.h to be –16.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u16, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u16 is defined in phigs.h as:

```
typedef union {
        struct Pgdp _u16 {
                Ppoint    center;      /* center point */
                Pfloat    d_angle;     /* spacing angle between rays */
                Pfloat    d_dist;      /* spacing between points on a ray */
        } gdp_u16;          /* radial grid */
} Pgdp_data;
```

The *center* component specifies the center of the radial grid. It is a Ppoint

structure containing *x* and *y* coordinates specified in MC. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
} Ppoint;
```

The *d_angle* specifies the angle between rays of the grid. The *d_dist* specifies the radial distance between points on a given ray.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPRADGRID, which is defined in phigs77.h to be –16.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPRADGRID should be as follows:

IL The number of integers = 0.

RL The number of real values = 4.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the radial grid in MC.

Element 2 is the *y* coordinate of the center of the radial grid in MC.

Element 3 is the angle between rays.

Element 4 is the radial distance between points on a given ray.

SL The number of strings = 0.

Execution

A 2D radial grid is a planar area defined by a center, a radius, and the spacing along the ray. The Grid Local Coordinate (GLC) system is located at the center point on the $z = 0$ plane. The horizontal and vertical axes of GLC are parallel to the *x* and *y* axes of the MC.

The radial grid uses the positive *x*-axis of GLC as the base ray, starting from the center, polymarkers are uniformly spaced with specified *d_dist* along this axis. By using the specified center point as center, it sweeps the base ray in a counter-clockwise direction, for every multiple of the specified *d_angle*, thus defining a ray for each one until the remainder angle is less than or equal to the *d_angle*. (The number of rays for the radial grid is always less than or equal to 2*PI / *d_angle*).

A full or an arc section of the radial grid is drawn when the structure is traversed, depending on whether or not the center is inside the view volume.

The coordinates used to specify the GLC are MCs. These can be any coordinate units that are convenient to the application. At traversal, these coordinate values are transformed by the current local and global modelling transformations, the view representation selected by the current view index, and the workstation transformation current on the workstation to which the structure is posted. However, since the polymarker has no geometric attributes, transformations affect only the grid position, but not the appearance, of the marker glyph in which the grid is constructed.

The current values of all polymarker attributes are applied to the 2D radial grid.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE –16 structure element. The Aspect Source Flags (ASFs) describe where to access the output display attributes. These attributes can come directly from the traversal state list or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

marker type	marker type ASF
marker size scale factor	marker size scale factor ASF
polymarker colour index	polymarker colour index ASF
polymarker index	
polymarker colour	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

POLYMARKER (3P)
POLYMARKER 3 (3P)
GENERALIZED DRAWING PRIMITIVE –15 (3P)
GENERALIZED DRAWING PRIMITIVE 3 –15 (3P)
GENERALIZED DRAWING PRIMITIVE 3 –16 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE -17 – create structure element specifying 2D mono-encoded text primitive

SYNOPSIS

C Syntax

void
pgdp (point_list, gdp_id, gdp_data)

Ppoint_list	∗point_list;	*array of points*
Pint	gdp_id;	*gdp function identifier*
Pgdp_data	∗gdp_data;	*data record pointer*

FORTRAN Syntax

SUBROUTINE pgdp (N, PXA, PYA, PRIMID, LDR, DATREC)

INTEGER	N	
REAL	PXA(∗), PYA(∗)	*coordinates of points (MC)*
INTEGER	PRIMID	*GDP identifier*
INTEGER	LDR	*dimension of data record array*
CHARACTER∗80	DATREC(LDR)	*data record*

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

The GENERALIZED DRAWING PRIMITIVE -17 subroutine puts a structure element containing a two-dimensional mono-encoded text primitive into the currently open structure.

The mono-encoded text primitive is used by the application when mixed character sets are required or when a single character set other than ASCII is required. The mono-encoded text primitive is an I-String, which is a series of substrings, each using a single distinct character set.

When the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE -17 subroutine is inserted in the open structure after the element pointed to by the current *element pointer*. When the current edit mode is REPLACE, the new GENERALIZED DRAWING PRIMITIVE -17 element replaces the element in the structure pointed to by the element pointer. In either case, the element pointer is updated to point to the new GENERALIZED DRAWING PRIMITIVE -17 element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_data
A pointer to a Pgdp_data union, gdp_u17, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u17 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u17 {
                Ppoint          text_pt;    /* text point */
                Phigs_i_string  i_string;   /* I-String */
        } gdp_u17;
} Pgdp_data;
```

text_pt A Ppoint structure containing the *x* and *y* coordinates that locate the mono-encoded text. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;        /* x coordinate */
        Pfloat      y;        /* y coordinate */
} Ppoint;
```

i_string Contains a list of mono-encodings. Phigs_i_string is defined in phigs.h as follows:

```
typedef struct {
        Pint                      num_encodings;    /* number of
                                                       encodings */
        Phigs_mono_encoding *mono_encodings;  /* list of encodings */
} Phigs_i_string;
```

Each mono-encoding (substring) contains a character set, an encoding state, and a string. Pmono_encoding is defined in phigs.h as follows:

```
typedef struct {
        Pint     char_set;          /* character set */
        char     encoding_state;    /* encoding state */
        char     *string;           /* character string */
} Phigs_mono_encoding;
```

char_set

Valid values for *char_set* are defined in phigs.h as follows:

PCS_ASCII	0
PCS_GREEK	−1
PCS_SYMBOL	−2
PCS_CARTOGRAPHIC	−3
PCS_KANJI	−4
PCS_KATAKANA	−5

encoding_state

Is used to encode within a mono-encoding a character encoding method (for example, EUC, CT). This parameter is not actually used by the SunPHIGS library. However, it can be used in conjunction with the INQUIRE ELEMENT CONTENT function to restore mono-encoded text to its original encoding method.

string A pointer to the character string.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_MONO_ENC_TXT, which is defined in phigs.h to be -17.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPMONOENCTXT, which is defined in phigs77.h to be -17.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.
The arguments passed to PACK DATA RECORD when PRIMID = PUGDPMONOENCTXT should be as follows:

IL The number of integers = (SL × 2).

IA Contains array of IL integer values. For each substring, there is the character set and an encoding state.

Valid values for the character set are defined in phigs77.h as follows:

PCSASCII	0
PCSGREEK	–1
PCSSYMBOL	–2
PCSCARTOGRAPHIC	–3
PCSKANJI	–4
PCSKATAKANA	–5

RL The number of real values = 2.

RA Contains array of RL real values.

Element 1 is the *x* coordinate of the point locating the I-String in MC.

Element 2 is the *y* coordinate of the point locating the I-String in MC.

SL The number of mono-encoded strings.

LSTR Array of length SL containing lengths of strings in STR.

STR Contains array of SL strings.

Execution

When the structure is traversed, the GENERALIZED DRAWING PRIMITIVE 3 -17 element draws the specified I-String on the plane in the Modelling Coordinate system (MC) that is defined by the text point and the two direction vectors. The text point defines the origin of the Text Local Coordinate (TLC) system, the first direction vector defines the positive *x* axis, and the second direction vector defines the positive *y* axis. Only the directions, not the lengths, of these vectors are relevant.

The precise position of the I-String is defined in relation to this plane by the current values of the text primitive attributes CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT.

The text point is specified in MC. The mono-encoded text 3 primitive is subject to the current transformations in the transformation pipeline from the MC system to the workstation display.

Other aspects of the appearance of the I-String are controlled by the attributes TEXT FONT, TEXT PRECISION, CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, and TEXT COLOUR INDEX. These attributes are applied to the entire I-string. They cannot be changed per substring.

The array of points that is passed to this subroutine in addition to the generalized drawing primitive data record is not used by the mono-encoded text generalized drawing primitive. When the structure is traversed, the GENERALIZED DRAWING PRIMITIVE -17 element draws the specified I-String. Each mono-encoding (substring) within the I-String uses its own character set. The position of the I-String is defined in relation to text_pt by the current values of the text primitive attributes CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT.

Other aspects of the appearance of the I-String are controlled by the attributes TEXT FONT, TEXT PRECISION, CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER EXPANSION FACTOR, CHARACTER SPACING and TEXT COLOUR INDEX. These attributes are applied to the entire I-string. They cannot be changed per substring.

The text point is specified in Modelling Coordinates (MC). The mono-encoded text primitive is subject to the current transformations in the transformation pipeline from the MC System to the workstation display.

The array of points that is passed to this subroutine in addition to the generalized drawing primitive data record is not used by the mono-encoded text generalized drawing primitive.

C Example

To render an I-String consisting of four ASCII characters, followed by four Greek characters, you could do the following:

```
extern char *ascii_chars, *greek_chars;
draw_text(x, y)
Pfloat  x, y;
{
        Pgdp_data       text_gdp;
        Phigs_mono_encoding     substrings[2];

        substrings[0] = ascii_chars;

        substrings[1] = greek_chars;

        text_gdp.gdp_u17.text_pt.x = x;
        text_gdp.gdp_u17.text_pt.y = y;
        text_gdp.gdp_u17.i_string.num_encodings = 2;
        text_gdp.gdp_u17.i_string.mono_encodings = substrings;
```

```
                pgdp((Ppoint_list *)NULL, PUGDP_MONO_ENC_TXT, &text_gdp);
        }
```

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -17 structure element. The Aspect Source Flags (ASF)s describe where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

text font	text font ASF
text precision	text precision ASF
character expansion factor	character expansion factor ASF
character spacing	character spacing ASF text colour@T{
	text colour index ASF
character height	
character up vector	
text path	
text alignment	
text index	
depth cue index	
name set	

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
GENERALIZED DRAWING PRIMITIVE 3 -17 (3P)
ESCAPE -10 (3P)
ESCAPE -11 (3P)
ESCAPE -12 (3P)
TEXT (3P)
INTRO INTERNATIONALIZATION (7P)
See *Text attributes* in Index

NAME

GENERALIZED DRAWING PRIMITIVE -18 – create structure element specifying 2D mono-encoded annotation text relative primitive

SYNOPSIS

C Syntax

```
void
pgdp ( point_list, gdp_id, gdp_data )
Ppoint_list    *point_list;    array of points
Pint           gdp_id;         gdp function identifier
Pgdp_data      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp ( N, PXA, PYA, PRIMID, LDR, DATREC )
INTEGER          N                  number of points (>=0)
REAL             PXA(*), PYA(*)     coordinates of points (MC)
INTEGER          PRIMID             GDP identifier
INTEGER          LDR                dimension of data record array
CHARACTER*80     DATREC(LDR)        data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

The GENERALIZED DRAWING PRIMITIVE -18 subroutine puts a structure element containing a two-dimensional mono-encoded annotation text primitive into the currently-open structure.

The mono-encoded annotation text primitive is used by the application when mixed character sets are required or when a single character set other than ASCII is required. The mono-encoded annotation text primitive is an I-String, which is a series of substrings, each using a single distinct character set.

When the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE -18 subroutine is inserted in the open structure after the element pointed to by the current *element pointer*. When the current edit mode is REPLACE, the new GENERALIZED DRAWING PRIMITIVE -18 element replaces the element in the structure pointed to by the element pointer. In either case, the element pointer is updated to point to the new GENERALIZED DRAWING PRIMITIVE -18 element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_data
A pointer to a Pgdp_data union, Pgdp_u18, containing the information needed to perform the function specified by *gdp_id*. Pgdp_u18 is defined in phigs.h as:

```
typedef union {
        struct Pgdp_u18 {
                Ppoint          ref_pt;         /* reference point */
                Pvec            anno_offset;    /* annotation offset */
                Phigs_i_string  i_string;       /* annotation I-String */
        } gdp_u18;
} Pgdp_data;
```

ref_pt A Ppoint structure containing the *x* and *y* coordinates in MC that locate the mono-encoded text. Ppoint is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
} Ppoint;
```

anno_offset
A Pvec structure containing the *x* and *y* coordinates of the offset of the I-String from the transformed reference point. The annotation offset specifies an offset in NPC. Pvec is defined in phigs.h as follows:

```
typedef struct {
        Pfloat   delta_x;     /* x magnitude */
        Pfloat   delta_y;     /* y magnitude */
} Pvec;
```

i_string Contains a list of mono-encodings. Phigs_i_string is defined in phigs.h as follows:

```
typedef struct {
        Pint                   num_encodings;    /* number of
                                                    encodings */
        Phigs_mono_encoding    *mono_encodings;  /* list of
                                                    encodings */
} Phigs_i_string;
```

Each mono-encoding (substring) contains a character set, an encoding state, and a string. Phigs_mono_encoding is defined in phigs.h as follows:

```
typedef struct {
        Pint    char_set;          /* character set */
        char    encoding_state;    /* encoding state */
        char    *string;           /* character string */
} Phigs_mono_encoding;
```

char_set
Valid values for char_set are defined in phigs.h as follows:

PCS_ASCII	0
PCS_GREEK	–1
PCS_SYMBOL	–2
PCS_CARTOGRAPHIC	–3
PCS_KANJI	–4
PCS_KATAKANA	–5

encoding_state
Is used to encode within a mono-encoding a character encoding method (that is, EUC, CT). This parameter is not actually used by the SunPHIGS library. However, it can be used in conjunction with the INQUIRE ELEMENT CONTENT function to restore mono-encoded text to its original encoding method.

string A pointer to the character string.

gdp_id The function identifier for this generalized drawing primitive is PUGDP_MONO_ENC_ATXT_REL, which is defined in phigs.h to be -18.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPMONOATXTR, which is defined in phigs77.h to be -18.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.
The arguments passed to PACK DATA RECORD when PRIMID = PUGDPMONOATXTR should be as follows:

IL The number of integers = (SL $\times$ 2).

IA Contains array of IL integer values. For each substring, there is the character set and an encoding state.

Valid values for the character set are defined in phigs77.h as follows:

PCSASCII	0
PCSGREEK	–1
PCSSYMBOL	–2
PCSCARTOGRAPHIC	–3
PCSKANJI	–4

RL The number of real values = 4.

RA Contains array of RL real values.

Element 1 is the *x* coordinate of the point locating the annotation I-String in MC.

Element 2 is the *y* coordinate of the point locating the annotation I-String in MC.

Element 3 is the *delta_x* offset in NPC locating the I-String relative to Element 1.

Element 4 is the *delta_y* offset in NPC locating the I-String relative to Element 2.

SL	The number of mono-encoded substrings.
LSTR	Array of SL integers specifying the lengths of the mono-encoded strings.
SA	Contains the array of SL substrings.

Execution

When the structure is traversed, the GENERALIZED DRAWING PRIMITIVE -18 element draws the specified I-String on the plane in NPC defined by the reference point and the annotation offset. These parameters define a Text Local Coordinate (TLC) system in NPC. The annotation offset added to the transformed reference point defines the origin of the TLC system. The *x* and *y* axes of the TLC system are parallel to and have the same direction as the *x* and *y* axes of NPC. The precise position of the annotation I-String is defined in relation to this plane by the current values of the text primitive attributes CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT.

Other aspects of the appearance of the I-String are controlled by the attributes TEXT FONT, TEXT PRECISION, CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER EXPANSION FACTOR, CHARACTER SPACING and TEXT COLOUR INDEX. These attributes are applied to the entire I-string. They cannot be changed per substring.

The reference point is subject to the current transformations in the transformation pipeline from the MC system to the workstation display. The I-String itself is only subject to the transformations in the transformation pipeline from the NPC system to the workstation display.

The array of points passed to this subroutine in addition to the generalized drawing primitive data record is not used by the mono-encoded annotation text generalized drawing primitive.

C Example

To render an I-String consisting of four ASCII characters, followed by four Greek characters, you could do the following:

```
extern char *ascii_chars, *greek_chars;
draw_text(x, y)
Pfloat  x, y;
{
        Pgdp_data       text_gdp;
        Phigs_mono_encoding     substrings[2];

        substrings[0] = ascii_chars;

        substrings[1] = greek_chars;
```

```
                    text_gdp.gdp_u18.ref_pt.x = x;
                    text_gdp.gdp_u18.ref_pt.y = y;
                    text_gdp.ugdp18_.anno_offset.delta_x = 0.0;
                    text_gdp.ugdp18_.anno_offset.delta_y = 0.0;
                    text_gdp.ugdp18_.i_string.num_encodings = 2;
                    text_gdp.ugdp18_.i_string.mono_encodings = substrings;
                    pgdp((Ppoint_list *)NULL, PUGDP_MONO_ENC_ATXT_REL, &text_gdp);
          }
```

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE -18 structure element. The Aspect Source Flags (ASF)s describe where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

text font	text font ASF
text precision	text precision ASF
character expansion factor	character expansion factor ASF
character spacing	character spacing ASF
text colour	text colour index ASF
annotation character height	
annotation character up vector	
annotation text path	
annotation text alignment	
text index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
GENERALIZED DRAWING PRIMITIVE 3 -18 (3P)
ESCAPE -10 (3P)
ESCAPE -11 (3P)
ESCAPE -12 (3P)
ANNOTATION TEXT RELATIVE (3P)
INTRO INTERNATIONALIZATION (7P)
See *Text Attributes* in Index

NAME

GENERALIZED DRAWING PRIMITIVE 3 – create 3D GDP elements such as circle 3, ellipse 3, or rectangular grid 3

SYNOPSIS

C Syntax

void
pgdp3 (point_list, gdp3_id, gdp_data)

Ppoint_list3	*point_list;	*array of points*
Pint	gdp3_id;	*gdp function identifier*
Pgdp_data3	*gdp_data;	*gdp data record*

FORTRAN Syntax

SUBROUTINE pgdp3 (N, PXA, PYA, PZA, PRIMID, LDR, DATREC)

INTEGER	N	*number of points (>= 0)*
REAL	PXA(*), PYA(*), PZA(*)	*coordinates of points (MC)*
INTEGER	PRIMID	*GDP 3 identifier*
INTEGER	LDR	*dimension of data record array*
CHARACTER*80	DATREC(LDR)	*data record*

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 creates the following GDP 3 structure elements:

- circle 3
- circular arc 3
- circular arc close 3
- annotation circle 3
- annotation circular arc 3
- annotation circular arc close 3
- ellipse 3
- elliptical arc 3
- elliptical arc close 3
- annotation ellipse 3
- annotation elliptical arc 3
- annotation elliptical arc close 3
- image 3
- rectangular grid 3
- radial grid 3
- mono-encoded text 3
- mono-encoded annotation text relative 3
- hinted NURBS surface

If the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE 3 function is inserted into the open structure after the element pointed to by the element pointer. If the current edit mode is REPLACE, the GENERALIZED DRAWING PRIMITIVE 3 element replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new structure element.

Note: Support for GDPs is implementation-dependent and workstation-dependent. All current SunPHIGS GDPs are implemented on all supported workstations.

Each generalized drawing primitive has its own page in the reference manual. The name of the page is generalized drawing primitive 3 -xx, where xx is replaced by the GDP identifiers; for example, GENERALIZED DRAWING PRIMITIVE 3 -2.

Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id An integer specifying the GENERALIZED DRAWING PRIMITIVE 3 to be performed. Predefined values as defined in phigs.h are:

–2	PUGDP_CIRCLE3	*Circle 3*
–3	PUGDP_CIRC_ARC3	*Circular arc 3*
–4	PUGDP_CIRC_ARC_CLOSE3	*Circular arc close 3*
–5	PUGDP_ANNOT_CIRCLE3	*Annotation circle 3*
–6	PUGDP_ANNOT_CIRC_ARC3	*Annotation circular arc 3*
–7	PUGDP_ANNOT_CIRC_ARC_CLOSE3	*Annotation circular arc close 3*
–8	PUGDP_ELLIPSE3	*Ellipse 3*
–9	PUGDP_ELLP_ARC3	*Elliptical arc 3*
–10	PUGDP_ELLP_ARC_CLOSE3	*Elliptical arc close 3*
–11	PUGDP_ANNOT_ELLIPSE3	*Annotation ellipse 3*
–12	PUGDP_ANNOT_ELLP_ARC3	*Annotation elliptical arc 3*
–13	PUGDP_ANNOT_ELLP_ARC_CLOSE3	*Annotation elliptical arc close 3*
–14	PUGDP_IMAGE3	*Image 3*
–15	PUGDP_RECT_GRID3	*Rectangular grid 3*
–16	PUGDP_RAD_GRID3	*Radial grid 3*
–17	PUGDP_MONO_ENC_TXT3	*Mono-encoded text 3*
–18	PUGDP_MONO_ENC_ATXT_REL3	*Mono-encoded annotation text relative 3*
–19	PUGDP_HINTED_NURBS_SURF3	*Hinted NURBS surface*

gdp_data
A pointer to a Pgdp_data3 union containing the information needed to perform the function specified by *gdp3_id*. Pgdp_data3 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_r1 {
                Pint            unused;
        } gdp3_r1;
        Pdata   unsupp;         /* unsupported GDP data record  */
                                /* implementation dependent     */
```

```
struct Pgdp3_u2 {
        Ppoint3         center;         /∗ center point ∗/
        Pvec3           dir[2];         /∗ direction vectors ∗/
        Pfloat          radius;         /∗ radius ∗/
} gdp3_u2;                      /∗ circle 3 ∗/
struct Pgdp3_u3 {
        Ppoint3         center;         /∗ center point ∗/
        Pvec3           dir[2];         /∗ direction vectors ∗/
        Pfloat          radius;         /∗ radius ∗/
        Pfloat          st_ang;         /∗ start angle ∗/
        Pfloat          end_ang;        /∗ end angle ∗/
} gdp3_u3;                      /∗ circular arc 3 ∗/
struct Pgdp3_u4 {
        Ppoint3         center;         /∗ center point ∗/
        Pvec3           dir[2];         /∗ direction vectors ∗/
        Pfloat          radius;         /∗ radius ∗/
        Pfloat          st_ang;         /∗ start angle in radian∗/
        Pfloat          end_ang;        /∗ end angle in radian ∗/
        Pint            cls_type;       /∗ close type ∗/
} gdp3_u4;                      /∗ circular arc close 3 ∗/
struct Pgdp3_u5 {
        Ppoint3         center;         /∗ center point ∗/
        Pfloat          radius;         /∗ radius ∗/
} gdp3_u5;                      /∗ annotation circle 3 ∗/
struct Pgdp3_u6 {
        Ppoint3         center;         /∗ center point ∗/
        Pfloat          radius;         /∗ radius ∗/
        Pfloat          st_ang;         /∗ start angle ∗/
        Pfloat          end_ang;        /∗ end angle ∗/
} gdp3_u6;                      /∗ annotation circular arc 3 ∗/
struct Pgdp3_u7 {
        Ppoint3         center;         /∗ center point ∗/
        Pfloat          radius;         /∗ radius ∗/
        Pfloat          st_ang;         /∗ start angle in radian ∗/
        Pfloat          end_ang;        /∗ end angle in radian ∗/
        Pint            cls_type;       /∗ close type ∗/
} gdp3_u7;                      /∗ annotation circular arc close 3 ∗/
struct Pgdp3_u8 {
        Ppoint3         center;         /∗ center point ∗/
        Pvec3           major;          /∗ major axis ∗/
        Pvec3           minor;          /∗ minor axis ∗/
} gdp3_u8;                      /∗ ellipse 3 ∗/
struct Pgdp3_u9 {
        Ppoint3       center;         /∗ center point ∗/
        Pvec3         major;          /∗ major axis ∗/
        Pvec3         minor;          /∗ minor axis ∗/
        Pfloat        st_ang;         /∗ start angle ∗/
        Pfloat        end_ang;        /∗ end angle ∗/
} gdp3_u9;            /∗ elliptical arc 3 ∗/
struct Pgdp3_u10 {
        Ppoint3       center;         /∗ center point ∗/
        Pvec3         major;          /∗ major axis ∗/
        Pvec3         minor;          /∗ minor axis ∗/
        Pfloat        st_ang;         /∗ start angle ∗/
        Pfloat        end_ang;        /∗ end angle ∗/
        Pint          cls_type;       /∗ close type ∗/
} gdp3_u10;           /∗ elliptical arc close 3 ∗/
struct Pgdp3_u11 {
        Ppoint3       center;         /∗ center point ∗/
        Pvec          major;          /∗ major axis ∗/
        Pvec          minor;          /∗ minor axis ∗/
} gdp3_u11;           /∗ annotation ellipse 3 ∗/
struct Pgdp3_u12 {
        Ppoint3       center;         /∗ center point ∗/
        Pvec          major;          /∗ major axis ∗/
        Pvec          minor;          /∗ minor axis ∗/
        Pfloat        st_ang;         /∗ start angle ∗/
        Pfloat        end_ang;        /∗ end angle ∗/
} gdp3_u12;           /∗ annotation elliptical arc 3 ∗/
struct Pgdp3_u13 {
        Ppoint3       center;         /∗ center point ∗/
        Pvec          major;          /∗ major axis ∗/
        Pvec          minor;          /∗ minor axis ∗/
        Pfloat        st_ang;         /∗ start angle ∗/
        Pfloat        end_ang;        /∗ end angle ∗/
        Pint          cls_type;       /∗ close type ∗/
} gdp3_u13;           /∗ annotation elliptical arc close 3 ∗/
struct Pgdp3_u14 {
        Ppoint3       ref_pt;         /∗ reference point in MC ∗/
        Pint_size     dim;            /∗ device space dimensions ∗/
                                      /∗ of image ∗/
        Pint          colr_type;      /∗ PINDIRECT, PRGB, PCIE, ∗/
                                      /∗ PHSV, PHLS, and so on ∗/
        union {
        char          ∗chars;         /∗ 8-bit indices ∗/
        Pint          ∗ints;          /∗ 32-bit XBGR ∗/
```

```
                        } colrs;
                } gdp3_u14;
                struct Pgdp3_u15 {
                        Ppoint3         origin;         /∗ center point ∗/
                        Pfloat          dx;             /∗ spacing along the x axis ∗/
                        Pfloat          dy;             /∗ spacing along the y axis ∗/
                } gdp3_u15;
                struct Pgdp3_u16 {
                        Ppoint3         center;         /∗ center point ∗/
                        Pfloat          d_angle;        /∗ spacing angle between
                                                           rays ∗/
                        Pfloat          d_dist;         /∗ spacing between points
                                                           on a ray ∗/
                } gdp3_u16;
                struct Pgdp3_u17 {
                        Ppoint3         text_pt;        /∗ text point ∗/
                        Pvec3           dir[2];         /∗ direction vectors ∗/
                        Phigs_i_string  i_string;       /∗ I-String ∗/
                } gdp3_u17;
                struct Pgdp3_u18 {
                        Ppoint3         ref_pt;         /∗ reference point ∗/
                        Pvec3           anno_offset;    /∗ annotation offset ∗/
                        Phigs_i_string  i_string;       /∗ annotation I-String ∗/
                } gdp3_u18;
                struct Pgdp3_u19 {
                        Pint            uorder;         /∗ U spline order ∗/
                        Pint            vorder;         /∗ V spline order ∗/
                        Pfloat_list     ∗uknots;        /∗ U knots ∗/
                        Pfloat_list     ∗vknots;        /∗ V knots ∗/
                        Prational       rationality;    /∗ rationality selector ∗/
                        Ppoint_grid34   ∗grid;          /∗ grid of 3D or 4D control points ∗/
                        Pint            nloops;         /∗ number of trim curve loops ∗/
                        Ptrimcurve_list ∗tloops;        /∗ trim curve loops ∗/
                        Pnurb_surf_hints                ∗hints;   /∗ surface shape hints ∗/
                } gdp3_u19;
        } Pgdp_data3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID

An integer specifying the GENERALIZED DRAWING PRIMITIVE 3 to be performed. Predefined values defined in phigs77.h are:

–2	PUGDPCIRC3	*Circle 3*
–3	PUGDPCIRCARC3	*Circular arc 3*
–4	PUGDPCIRCARCCL3	*Circular arc close 3*
–5	PUGDPACIRC3	*Annotation circle 3*
–6	PUGDPACIRCARC3	*Annotation circular arc 3*
–7	PUGDPACIRCARCCL3	*Annotation circular arc close 3*
–8	PUGDPELLP3	*Ellipse 3*
–9	PUGDPELLPARC3	*Elliptical arc 3*
–10	PUGDPELLPARCCL3	*Elliptical arc close 3*
–11	PUGDPAELLP3	*Annotation ellipse 3*
–12	PUGDPAELLPARC3	*Annotation elliptical arc 3*
–13	PUGDPAELLPARCCL3	*Annotation elliptical arc close 3*
–14	PUGDPIMG3	*Image 3*
–15	PUGDPRECTGRID3	*Rectangular grid 3*
–16	PUGDPRADGRID3	*Radial grid 3*
–17	PUGDPMONOENCTXT3	*Mono-encoded text 3*
–18	PUGDPMONOENCATXTREL3	*Mono-encoded annotation text relative 3*
–19	PUGDPHINTEDNURBSURF3	*Hinted NURBS surface*

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by *PRIMID*.

Execution

Circle, Circular Arc, Circular Arc Close, Ellipse, Elliptical Arc, and Elliptical Arc Close are rendered according to type 1 curve approximation criteria (Constant Parametric Subdivision). A default curve approximation criteria type 1, and approximation value of 36.0 will be used if curve approximation criteria other than type 1 or curve approximation value of less than 36.0 is specified.

Image, mono-encoded text, and mono-encoded annotation text relative are not affected by curve approximation criteria.

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

PACK DATA RECORD (3P)
GENERALIZED DRAWING PRIMITIVE 3 -2 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -3 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -4 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -5 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -6 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -7 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -8 (3P)

GENERALIZED DRAWING PRIMITIVE 3 -9 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -10 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -11 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -12 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -13 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -14 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -15 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -16 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -17 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -18 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -19 (3P)
GENERALIZED DRAWING PRIMITIVE (3P)

NAME

GENERALIZED DRAWING PRIMITIVE 3 -2 – create 3D GDP circle 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -2 creates the 3D circle 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_CIRCLE3, which is defined in phigs.h to be -2.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u2, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u2 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u2 {
                Ppoint3   center;    /* center point */
                Pvec3     dir[2];    /* direction vectors */
                Pfloat    radius;    /* radius */
        } gdp3_u2;          /* circle */
Pgdp3_data
```

The center component specifies the center of the circle 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
        Pfloat      z;      /∗ z coordinate ∗/
} Ppoint3;
```

The *dir* component is an array of two Pvec3 structures containing direction vectors. The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /∗ x magnitude ∗/
        Pfloat   delta_y;   /∗ y magnitude ∗/
        Pfloat   delta_z;   /∗ z magnitude ∗/
} Pvec3;
```

The radius component specifies the radius of the circle 3 in MC.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC2.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRC3, which is defined in phigs77.h to be -2.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 10.

RA Contains array of RL real values.
Element 1 is the *x* coordinate of the center of the circle 3 in MC.
Element 2 is the *y* coordinate of the center of the circle 3 in MC.
Element 3 is the *z* coordinate of the center of the circle 3 in MC.
Element 4 is the *x* component of the first direction vector in MC.
Element 5 is the *y* component of the first direction vector in MC.
Element 6 is the *z* component of the first direction vector in MC.
Element 7 is the *x* component of the second direction vector in MC.
Element 8 is the *y* component of the second direction vector in MC.

Element 9 is the *z* component of the second direction vector in MC.

Element 10 is the radius of the circle in MC.

SL The number of strings = 0.

Execution

A circle 3 is a planar area defined by a center point and a radius. The Circle Local Coordinate (CLC) system is located at the center point. Let P be the center point defined in MC, and U and V be the direction vectors. The plane of the circle is a plane that contains P and is perpendicular U x V. The horizontal and vertical axes of CLC are defined by dir[0] and dir[1]. The positive *x* axis of the CLC system is parallel to dir[0]. Let T be a vector perpendicular to dir[0] in the circle 3 plane. If T.dir[1] is positive, then the positive *y* axis is parallel to T; otherwise, the positive *y* axis is opposite to T.

The center point and radius are specified in MC. At traversal, the circle 3 in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted.

The current values of all fill area set attributes will be applied to the circle 3 when it is drawn.

The circle 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circle 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -2 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF

edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Interior Attributes* in *Index*
See *Edge Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -3 – create 3D GDP circular arc 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -3 creates the 3D circular arc 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_CIRC_ARC3, which is defined in phigs.h to be -3.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u3, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u3 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u3 {
                Ppoint3    center;      /* center point */
                Pvec3      dir[2];      /* direction vectors */
                Pfloat     radius;      /* radius */
                Pfloat     st_ang;      /* start angle in radians */
                Pfloat     end_ang;     /* end angle in radians */
        } gdp3_u3;          /* circular arc 3 */
} Pgdp3_data;
```

The center component specifies the center of the circular arc 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The dir component is an array of two Pvec3 structures containing direction vectors. The radius component specifies the radius of the circular arc 3 in MC. The st_ang component specifies the start angle of the circular arc 3 in radians. The end_ang component specifies the end angle of the circular arc 3 in radians. The start and end angles are specified in a counterclockwise direction with the MC positive *x* axis as the origin.

The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
        Pfloat   delta_z;   /* z magnitude */
} Pvec3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRCARC3, which is defined in phigs77.h to be -3.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRCARC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 12.

RA Contains array of RL real values.
Element 1 is the *x* coordinate of the center of the circular arc 3 in MC.
Element 2 is the *y* coordinate of the center of the circular arc 3 in MC.
Element 3 is the *z* coordinate of the center of the circular arc 3 in MC.

Element 4 is the *x* component of the first direction vector in MC.
Element 5 is the *y* component of the first direction vector in MC.
Element 6 is the *z* component of the first direction vector in MC.
Element 7 is the *x* component of the second direction vector in MC.
Element 8 is the *y* component of the second direction vector in MC.
Element 9 is the *z* component of the second direction vector in MC.
Element 10 is the radius of the circular arc 3 in MC.
Element 11 is the start angle of the circular arc 3 in radians.
Element 12 is the end angle of the circular arc 3 in radians.

SL The number of strings = 0.

Execution

A circular arc 3 is a curve defined by a center point, a radius, a start angle, and an end angle. Let P be the center point defined in the MC, and U and V be the direction vectors, respectively. The plane that the circular arc 3 lies on is a plane that contains P and is perpendicular to U X V. The Circle Local Coordinate (CLC) system is located at the center point. The horizontal and vertical axes of CLC are defined by dir[0] and dir[1]. The positive *x* axis of the CLC system is parallel to dir[0]. Let T be a vector perpendicular to dir[0] in the circle 3 plane. If T.dir[1] is positive, then the positive *y* axis is parallel to T; otherwise, the positive *y* axis is opposite to T.

The center point and radius are specified in MC. The start and end angles are specified in radians in the counterclockwise direction with the CLC positive *x* axis as the origin. At traversal, the circular arc 3 in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. A circular arc 3 greater than 360 degrees is drawn as a hollow circle 3.

The current values of all polyline attributes will be applied to the circular arc 3 when it is drawn.

The circular arc 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circular arc 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -3 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline shading method	polyline shading method ASF
curve approximation criteria	curve approximation criteria ASF
polyline index	
depth cue index	

name set

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -4 – create 3D GDP circular arc close 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                      number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*) coordinates of points (MC)
INTEGER          PRIMID                 GDP 3 identifier
INTEGER          LDR                    dimension of data record array
CHARACTER*80     DATREC(LDR)            data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -4 creates the 3D circular arc close 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_CIRC_ARC_CLOSE3, which is defined in phigs.h to be -4.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u4, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u4 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u4 {
                Ppoint3   center;     /* center point */
                Pvec3     dir[2];     /* direction vectors */
                Pfloat    radius;     /* radius */
                Pfloat    st_ang;     /* start angle in radians */
                Pfloat    end_ang;    /* end angle in radians */
                Pint      cls_type;   /* close type */
        } gdp3_u4;          /* circular arc close 3 */
} Pgdp_data3;
```

The center component specifies the center of the circular arc close 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The *dir* component is an array of two Pvec3 structures containing direction vectors. The radius component specifies the radius of the circular arc close 3 in MC. The st_ang component specifies the start angle of the circular arc close 3 in radians. The end_ang component specifies the end angle of the circular arc close 3 in radians. The start and end angles are specified in a counterclockwise direction with the MC positive *x* axis as the origin. The cls_type component is either PARC_CLOSE_PIE or PARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the circular arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the circular arc and the chord from the starting point to the ending point.

The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
        Pfloat   delta_z;   /* z magnitude */
} Pvec3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPCIRCARCCL3, which is defined in phigs77.h to be -4.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPCIRCARCCL3 should be as follows:

IL	The number of integers = 1.
IA	Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).
RL	The number of real values = 12.
RA	Contains array of RL real values. Element 1 is the *x* coordinate of the center of the circular arc close 3 in MC. Element 2 is the *y* coordinate of the center of the circular arc close 3 in MC. Element 3 is the *z* coordinate of the center of the circular arc close 3 in MC. Element 4 is the *x* component of the first direction vector in MC. Element 5 is the *y* component of the first direction vector in MC. Element 6 is the *z* component of the first direction vector in MC. Element 7 is the *x* component of the second direction vector in MC. Element 8 is the *y* component of the second direction vector in MC. Element 9 is the *z* component of the second direction vector in MC. Element 10 is the radius of the circular arc close 3 in MC. Element 11 is the start angle of the circular arc close 3 in radians. Element 12 is the end angle of the circular arc close 3 in radians.
SL	The number of strings = 0.

Execution

A circular arc close 3 is a planar area defined by a center point, a radius, a start angle, and an end angle. Let P be center point defined in the MC, and U and V be the direction vectors respectively. The plane which the circular arc close 3 lies on is a plane that contains P and is perpendicular to U x V. The Circle Local Coordinate (CLC) system is located at the center point. The horizontal and vertical axes of CLC are defined by dir[0] and dir[1]. The positive *x* axis of the CLC system is parallel to dir[0]. Let T be a vector perpendicular to dir[0] in the circle 3 plane. If T.dir[1] is positive, then the positive *y* axis is parallel to T; otherwise, the positive *y* axis is opposite to T.

The center point and radius are specified in MC. The start and end angles are specified in radians in the counterclockwise direction, with the CLC positive *x* axis as the origin. At traversal, the circular arc close in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. A circular arc close 3 greater than 360 degrees is drawn as a filled circle 3.

The current values of all fill area set attributes will be applied to the circular arc close 3 when it is drawn.

The circular arc close 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the circular arc close 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -4 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Interior Attributes* in *Index*
See *Edge Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -5 – create 3D GDP annotation circle 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                       number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)  coordinates of points (MC)
INTEGER          PRIMID                  GDP 3 identifier
INTEGER          LDR                     dimension of data record array
CHARACTER*80     DATREC(LDR)             data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -5 creates the 3D annotation circle 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRCLE3, which is defined in phigs.h to be -5.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u5, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u5 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u5 {
                Ppoint3    center;     /* center point */
                Pfloat     radius;     /* radius */
        } gdp3_u5;             /* annotation circle 3 */
} Pgdp_data3;
```

The center component specifies the center of the annotation circle 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat     x;     /* x coordinate */
        Pfloat     y;     /* y coordinate */
        Pfloat     z;     /* z coordinate */
} Ppoint3;
```

The radius component specifies the radius of the annotation circle 3 in NPC.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRC3, which is defined in phigs77.h to be -5.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 4.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the center of the annotation circle 3 in MC.
Element 2 is the *y* coordinate of the center of the annotation circle 3 in MC.
Element 3 is the *z* coordinate of the center of the annotation circle 3 in MC.
Element 4 is the radius of the annotation circle in NPC.

SL The number of strings = 0.

Execution

An annotation circle 3 is a planar area defined by a center point and a radius. The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes.

The center point of the annotation circle 3 is specified in MC. The radius of the annotation circle 3 is specified in CLC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation circle 3.

The current values of all fill area set attributes will be applied to the annotation circle 3 when it is drawn.

Annotation circle is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circle 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -5 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -6 – create 3D GDP annotation circular arc 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                         number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)    coordinates of points (MC)
INTEGER          PRIMID                    GDP 3 identifier
INTEGER          LDR                       dimension of data record array
CHARACTER*80     DATREC(LDR)               data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -6 creates the 3D annotation circular arc 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRC_ARC3, which is defined in phigs.h to be -6.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u6, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u6 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u6 {
                Ppoint3    center;      /* center point */
                Pfloat     radius;      /* radius */
                Pfloat     st_ang;      /* start angle in radians */
                Pfloat     end_ang;     /* end angle in radians */
        } gdp3_u6;          /* annotation circular arc 3 */
} Pgdp_data3;    /* annotation circular arc 3 */
```

The center component specifies the center of the annotation circular arc 3. It is a

Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The radius component specifies the radius of the annotation circular arc 3 in NPC. The st_ang component specifies the start angle of the annotation circular arc 3 in radians. The end_ang component specifies the end angle of the annotation circular arc 3 in radians. The start and end angles are specified in a counterclockwise direction with the MC positive *x* axis as the origin.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRCARC3, which is defined in phigs77.h to be -6.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRCARC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 6.

RA Contains an array of RL real values.
Element 1 is the *x* coordinate of the center of the annotation circular arc 3 in MC.

Element 2 is the *y* coordinate of the center of the annotation circular arc 3 in MC.

Element 3 is the *z* coordinate of the center of the annotation circular arc 3 in MC.

Element 4 is the radius of the annotation circular arc 3 in NPC.

Element 5 is the start angle of the annotation circular arc 3 in radians.

Element 6 is the end angle of the annotation circular arc 3 in radians.

SL The number of strings = 0.

Execution

An annotation circular arc 3 is a curve defined by a center point, a radius, a start angle, and an end angle.

The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes.

The center point of the annotation circular arc 3 is specified in MC. The radius of the annotation circular arc 3 is specified in CLC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation circular arc 3. An annotation circular arc 3 greater than 360 degrees is drawn as a hollow annotation circle 3.

The current values of all polyline attributes will be applied to the annotation circular arc 3 when it is drawn.

Annotation circular arc 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circular arc 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -6 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -7 – create 3D GDP annotation circular arc close 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER         N                       number of points (>= 0)
REAL            PXA(*), PYA(*), PZA(*)  coordinates of points (MC)
INTEGER         PRIMID                  GDP 3 identifier
INTEGER         LDR                     dimension of data record array
CHARACTER*80    DATREC(LDR)             data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -7 creates the 3D annotation circular arc close 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_CIRC_ARC_CLOSE3, which is defined in phigs.h to be -7.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u7, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u7 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u7 {
                Ppoint3   center;     /* center point */
                Pfloat    radius;     /* radius */
                Pfloat    st_ang;     /* start angle in radians */
                Pfloat    end_ang;    /* end angle in radians */
                Pint      cls_type;   /* close type */
        } gdp3_u7;              /* annotation circular arc close 3 */
} Pgdp_data3;
```

The center component specifies the center of the annotation circular arc close 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;        /* x coordinate */
        Pfloat      y;        /* y coordinate */
        Pfloat      z;        /* z coordinate */
} Ppoint3;
```

The radius component specifies the radius of the annotation circular arc close 3 in NPC. The st_ang component specifies the start angle of the annotation circular arc close 3 in radians. The end_ang component specifies the end angle of the annotation circular arc close 3 in radians. The start and end angles are specified in a counterclockwise direction with the positive *x* axis as the origin. The cls_type component is either PARC_CLOSE_PIE or PARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the circular arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the circular arc and the chord from starting point to the ending point.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPACIRCARCCL3, which is defined in phigs77.h to be -7.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPACIRCARCCL3 should be as follows:

IL The number of integers = 1.

IA Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).

RL The number of real values = 6.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation circular arc close 3 in MC.

Element 2 is the *y* coordinate of the center of the annotation circular arc close 3 in MC.

Element 3 is the *z* coordinate of the center of the annotation circular arc close 3 in MC.

Element 4 is the radius of the annotation circular arc close 3 in NPC.

Element 5 is the start angle of the annotation circular arc close in radians.

Element 6 is the end angle of the annotation circular arc close in radians.

SL The number of strings = 0.

Execution

An annotation circular arc close 3 is a planar area defined by a center point, a radius, a start angle, and an end angle. The Circle Local Coordinate (CLC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes.

The center point of the annotation circular arc close 3 is specified in MC. The radius of the annotation circular arc close 3 is specified in CLC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation circular arc close 3. An annotation circular arc close 3 greater than 360 degrees is drawn as a filled annotation circle 3.

The current values of all fill area set attributes will be applied to the annotation circular arc close 3 when it is drawn.

Annotation circular arc close 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation circular arc close 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -7 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	

depth cue index
name set

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -8 – create 3D GDP ellipse 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -8 creates the 3D ellipse 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ELLIPSE3, which is defined in phigs.h to be -8.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u8, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u8 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u8 {
                Ppoint3  center;    /* center point */
                Pvec3    major;     /* major axis */
                Pvec3    minor;     /* minor axis */
        } gdp3_u8;          /* ellipse 3 */
} Pgdp_data3;
```

The center component specifies the center of the ellipse 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;       /* x coordinate */
        Pfloat      y;       /* y coordinate */
        Pfloat      z;       /* z coordinate */
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. It is a Pvec3 structure defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
        Pfloat   delta_z;   /* z magnitude */
} Pvec3;
```

The minor component specifies the minor axis of the ellipse 3. The *x* and *y* magnitudes of the major and minor axes of the ellipse 3 are specified in MC. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLP3, which is defined in phigs77.h to be -8.

LDR The dimension of *DATREC* (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPELLP3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 9.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the ellipse 3 in MC.

Element 2 is the *y* coordinate of the center of the ellipse 3 in MC.

Element 3 is the *z* coordinate of the center of the ellipse 3 in MC.

Element 4 is the *x* component of the ellipse 3 major axis in MC.

Element 5 is the *y* component of the ellipse 3 major axis in MC.

Element 6 is the *z* component of the ellipse 3 major axis in MC.

Element 7 is the *x* component of the ellipse 3 minor axis in MC.

Element 8 is the *y* component of the ellipse 3 minor axis in MC.

Element 9 is the *z* component of the ellipse 3 minor axis in MC.

SL The number of strings = 0.

Execution

An ellipse 3 is a planar area defined by a center point, a major axis, and a minor axis. Let P be the center point defined in the MC, and U and V be the major and minor axes, respectively. The plane on which the ellipse 3 lies is a plane that contains P and is perpendicular to U × V. The origin of the ELC system is located at the center point. The horizontal and vertical axes of the ELC system are defined by the major and minor axes. The positive *x* axis of the ELC system is parallel to U. Let T be a vector perpendicular to U in the ellipse 3 plane. If T.V is positive, then the positive *y* axis is parallel to T; otherwise, the positive *y* axis is opposite to T. In the event that the magnitudes of both axes are zero, the ellipse 3 is displayed as a dot (.) at the center point. In the event that the magnitude of only one of the axes is zero, the ellipse 3 is displayed as a line of correct magnitude along the other axis. If the major and minor axes fail to define the local coordinate system (that is, the vectors are parallel and non-zero), then a line, which is twice the length of the major axis, parallel to the major axis, and centered at the origin of the ellipse, is used to display the ellipse.

The center point and major and minor axes are specified in MC. At traversal, the ellipse in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted.

The current values of all fill area set attributes will be applied to the ellipse 3 when it is drawn.

The ellipse 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the ellipse 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -8 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF

interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -9 – create 3D GDP elliptical arc 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -9 creates the 3D elliptical arc 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ELLP_ARC3, which is defined in phigs.h to be -9.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u9, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u9 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u9 {
                Ppoint3 center;      /* center point */
                Pvec3   major;       /* major axis */
                Pvec3   minor;       /* minor axis */
                Pfloat  st_ang;      /* start angle in radians */
                Pfloat  end_ang;     /* end angle in radians */
        } gdp3_u9;          /* elliptical arc 3 */
} Pgdp_data3;
```

The center component specifies the center of the elliptical arc 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;       /∗ x coordinate ∗/
        Pfloat      y;       /∗ y coordinate ∗/
        Pfloat      z;       /∗ z coordinate ∗/
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. The minor component specifies the minor axis. The st_ang component specifies the start angle of the elliptical arc 3 in radians. The end_ang component specifies the end angle of the elliptical arc 3 in radians. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes. The angles are specified in radians in a counterclockwise direction with the positive *x* axis as the origin.

The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /∗ x magnitude ∗/
        Pfloat   delta_y;   /∗ y magnitude ∗/
        Pfloat   delta_z;   /∗ z magnitude ∗/
} Pvec3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLPARC3, which is defined in phigs77.h to be -9.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when *PRIMID* = PUGDPELLPARC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 11.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the elliptical arc 3 in MC.

Element 2 is the *y* coordinate of the center of the elliptical arc 3 in MC.

	Element 3 is the *z* coordinate of the center of the elliptical arc 3 in MC.
	Element 4 is the *x* component of the elliptical arc 3 major axis in MC.
	Element 5 is the *y* component of the elliptical arc 3 major axis in MC.
	Element 6 is the *z* component of the elliptical arc 3 major axis in MC.
	Element 7 is the *x* component of the elliptical arc 3 minor axis in MC.
	Element 8 is the *y* component of the elliptical arc 3 minor axis in MC.
	Element 9 is the *z* component of the elliptical arc 3 minor axis in MC.
	Element 10 is the start angle of the elliptical arc 3 in radians.
	Element 11 is the end angle of the elliptical arc 3 in radians.
SL	The number of strings = 0.

Execution

An elliptical arc 3 is a curve defined by a center point, a major axis, a minor axis, a start angle, and an end angle. Let P be the center point defined in the MC, and U and V be the major and minor axes respectively. The plane on which the elliptical arc 3 lies is a plane that contains P and is perpendicular to U × V. The origin of the ELC system is located at the center point. The horizontal and vertical axes of the ELC system are defined by the major and minor axes respectively.

The center point and major and minor axes of the elliptical arc 3 are specified in MC. The start and end angles are specified in radians in a counterclockwise direction with the ELC positive *x* axis as the origin. At traversal, the circular arc in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. An elliptical arc 3 greater than 360 degrees is drawn as a hollow ellipse 3.

The current values of all polyline attributes will be applied to the elliptical arc 3 when it is drawn.

The elliptical arc 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the elliptical arc 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -9 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF
linetype	linetype ASF
polyline shading method	polyline shading method ASF

curve approximation criteria	curve approximation criteria ASF
polyline index	

depth cue index
name set

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -10 – create 3D GDP elliptical arc close 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                      number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*) coordinates of points (MC)
INTEGER          PRIMID                 GDP 3 identifier
INTEGER          LDR                    dimension of data record array
CHARACTER*80     DATREC(LDR)            data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -10 creates the 3D elliptical arc close 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ELLP_ARC_CLOSE3, which is defined in phigs.h to be -10.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u10, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u10 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u10 {
                Ppoint3  center;     /* center point */
                Pvec3    major;      /* major axis */
                Pvec3    minor;      /* minor axis */
                Pfloat   st_ang;     /* start angle in radians */
                Pfloat   end_ang;    /* end angle in radians */
                Pint     cls_type;   /* close type */
        } gdp3_u10;      /* elliptical arc close 3 */
} Pgdp_data3;
```

The center component specifies the center of the elliptical arc close 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. The minor component specifies the minor axis. The st_ang component specifies the start angle of the elliptical arc close 3 in radians. The end_ang component specifies the end angle of the elliptical arc close 3 in radians. The horizontal and vertical axes of the Ellipse Local Coordinate (ELC) system are defined by the major and minor axes. The angles are specified in radians in a counterclockwise direction with the ELC positive *x* axis as the origin. The cls_type component is either PARC_CLOSE_PIE or PARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the elliptical arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the elliptical arc and the chord from the starting point to the ending point.

The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
        Pfloat   delta_z;   /* z magnitude */
} Pvec3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPELLPARCCL3, which is defined in phigs77.h to be -10.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPELLPARCCL3 should be as follows:

IL	The number of integers = 1.
IA	Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).
RL	The number of real values = 11.
RA	Contains an array of RL real values.
	Element 1 is the *x* coordinate of the center of the elliptical arc close 3 in MC.
	Element 2 is the *y* coordinate of the center of the elliptical arc close 3 in MC.
	Element 3 is the *z* coordinate of the center of the elliptical arc close 3 in MC.
	Element 4 is the *x* component of the elliptical arc close 3 major axis in MC.
	Element 5 is the *y* component of the elliptical arc close 3 major axis in MC.
	Element 6 is the *z* component of the elliptical arc close 3 major axis in MC.
	Element 7 is the *x* component of the elliptical arc close 3 minor axis in MC.
	Element 8 is the *y* component of the elliptical arc close 3 minor axis in MC.
	Element 9 is the *z* component of the elliptical arc close 3 minor axis in MC.
	Element 10 is the start angle of the elliptical arc close 3 in radians.
	Element 11 is the end angle of the elliptical arc close 3 in radians.
SL	The number of strings = 0.

Execution

An elliptical arc close 3 is a planar area defined by a center point, a major axis, a minor axis, a start angle, and an end angle. Let P be the center point defined in the MC, and U and V be the major and minor axes respectively. The plane on which the elliptical arc close 3 lies is a plane that contains P and is perpendicular to U x V. The origin of the ELC system is located at the center point. The horizontal and vertical axes of the ELC system are defined by the major and minor axes respectively.

The center point and major and minor axes of the elliptical arc close 3 are specified in MC. The start and end angles are specified in radians in a counterclockwise direction with the ELC positive *x* axis as the origin. At traversal, the circular arc in MC is transformed by the current local and global modelling transformations, the view representation selected by the *current view index*, and the workstation transformation current on the workstation to which the structure is posted. An elliptical arc close 3 greater than 360 degrees is drawn as a filled ellipse 3.

The current values of all fill area attributes will be applied to the elliptical arc close 3 when it is drawn.

The elliptical arc close 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the elliptical arc close 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -10 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance equation	interior reflectance equation ASF
back interior reflectance equation	back interior reflectance equation ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
curve approximation criteria	curve approximation criteria ASF
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
face distinguishing mode	
face culling mode	
depth cue index	
light source state	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET CURVE APPROXIMATION CRITERIA (3PP)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -11 – create 3D GDP annotation ellipse 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -11 creates the 3D annotation ellipse 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLIPSE3, which is defined in phigs.h to be -11.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u11, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u11 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u11 {
                Ppoint3    center;    /* center point */
                Pvec       major;     /* major axis */
                Pvec       minor;     /* minor axis */
        } gdp3_u11;           /* annotation ellipse 3 */
} Pgdp_data3;
```

The center component specifies the center of the annotation ellipse 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. The minor component specifies the minor axis of the annotation ellipse 3. The *x* and *y* magnitudes of the major and minor axes of the annotation ellipse 3 are specified in NPC.

The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPAELLP3, which is defined in phigs77.h to be -11.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPAELLP3 should be as follows:
The number of integers = 0.

RL The number of real values = 7.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation ellipse 3 in MC.

Element 2 is the *y* coordinate of the center of the annotation ellipse 3 in MC.

Element 3 is the *z* coordinate of the center of the annotation ellipse 3 in MC.

Element 4 is the *x* component of the annotation ellipse 3 major axis in NPC.

Element 5 is the *y* component of the annotation ellipse 3 major axis in NPC.

Element 6 is the *x* component of the annotation ellipse 3 minor axis in NPC.

Element 7 is the *y* component of the annotation ellipse 3 minor axis in NPC.

SL The number of strings = 0.

Execution

An annotation ellipse 3 is a planar area defined by a center point, a major axis, and a minor axis. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes. The horizontal and vertical axes of the ELC system are defined by the major and minor axes, respectively.

The center point of the annotation ellipse 3 is specified in MC. The major and minor axes of the annotation ellipse 3 are specified in ELC. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation ellipse.

The current values of all fill area set attributes will be applied to the annotation ellipse 3 when it is drawn.

Annotation ellipse 3 rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation ellipse 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -11 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -12 – create 3D GDP annotation elliptical arc 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER          N                        number of points (>= 0)
REAL             PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER          PRIMID                   GDP 3 identifier
INTEGER          LDR                      dimension of data record array
CHARACTER*80     DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -12 creates the 3D annotation elliptical arc 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLP_ARC3, which is defined in phigs.h to be -12.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u12, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u12 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u12 {
                Ppoint3   center;     /* center point */
                Pvec      major;      /* major axis */
                Pvec      minor;      /* minor axis */
                Pfloat    st_ang;     /* start angle in radians */
                Pfloat    end_ang;    /* end angle in radians */
        } gdp3_u12;           /* annotation elliptical arc 3 */
} Pgdp_data3;
```

The center component specifies the center of the annotation elliptical arc 3. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /∗ x coordinate ∗/
        Pfloat      y;      /∗ y coordinate ∗/
        Pfloat      z;      /∗ z coordinate ∗/
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. The minor component specifies the minor axis. The major and minor axes of the annotation ellipse are specified in NPC. The st_ang component specifies the start angle of the elliptical arc. The end_ang component specifies the end angle of the elliptical arc. The angles are specified in radians in a counterclockwise direction with the Ellipse Local Coordinate (ELC) system positive *x* axis as the origin.

The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /∗ x magnitude ∗/
        Pfloat   delta_y;   /∗ y magnitude ∗/
} Pvec;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPAELLPARC3, which is defined in phigs77.h to be -12.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when *PRIMID* = PUGDPAELLPARC3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 9.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the annotation elliptical arc 3 in MC.

Element 2 is the *y* coordinate of the center of the annotation elliptical arc 3 in MC.

Element 3 is the *z* coordinate of the center of the annotation elliptical arc 3 in MC.

Element 4 is the *x* component of the annotation elliptical arc 3 major axis in NPC.

Element 5 is the *y* component of the annotation elliptical arc 3 major axis in NPC.

Element 6 is the *x* component of the annotation elliptical arc 3 minor axis in NPC.

Element 7 is the *y* component of the annotation elliptical arc 3 minor axis in NPC.

Element 8 is the start angle of the annotation elliptical arc 3 in radians.

Element 9 is the end angle of the annotation elliptical arc 3 in radians.

SL The number of strings = 0.

Execution

An annotation elliptical arc 3 is a curve defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The horizontal and vertical axes of the ELC system are defined by the major and minor axes, respectively. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes.

The center point of the annotation elliptical arc 3 is specified in MC. The major and minor axes of the annotation elliptical arc 3 are specified in ELC. The start and end angles are specified in radians with the ELC positive *x* axis as the origin. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation elliptical arc 3. An annotation elliptical arc 3 greater than 360 degrees is drawn as a hollow annotation ellipse 3.

The current values of all polyline attributes will be applied to the annotation elliptical arc 3 when it is drawn.

Annotation elliptical arc 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation elliptical arc 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -12 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

polyline colour	polyline colour index ASF
linewidth scale factor	linewidth scale factor ASF

linetype	linetype ASF
polyline index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Line Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -13 – create 3D GDP annotation elliptical arc close 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER         N                         number of points (>= 0)
REAL            PXA(*), PYA(*), PZA(*)    coordinates of points (MC)
INTEGER         PRIMID                    GDP 3 identifier
INTEGER         LDR                       dimension of data record array
CHARACTER*80    DATREC(LDR)               data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -13 creates the 3D annotation elliptical arc close 3 GDP structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_ANNOT_ELLP_ARC_CLOSE3, which is defined in phigs.h to be -13.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u13, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u13 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u13 {
                Ppoint3   center;     /* center point */
                Pvec      major;      /* major axis */
                Pvec      minor;      /* minor axis */
                Pfloat    st_ang;     /* start angle in radians */
                Pfloat    end_ang;    /* end angle in radians */
                Pint      cls_type;   /* close type */
        } gdp3_u13;          /* annotation elliptical arc close 3 */
    } Pgdp_data3;
```

The center component specifies the center of the annotation elliptical arc close 3. It is a Ppoint3 structure containing *x* and *y* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

The major component specifies the length and direction of the major axis relative to the center point. The minor component specifies the minor axis. The major and minor axes of the annotation ellipse are specified in NPC. The st_ang component specifies the start angle of the elliptical arc close. The end_ang component specifies the end angle of the elliptical arc close. The angles are specified in radians in a counterclockwise direction with the Ellipse Local Coordinate (ELC) system's positive *x* axis as the origin. The cls_type component is either PARC_CLOSE_PIE or PARC_CLOSE_CHORD. If the close type is *pie*, the pie sector is defined by the elliptical arc, the center point, the starting point, and the ending point. If the close type is *chord*, the segment is defined by the elliptical arc and the chord from the starting point to the ending point.

The Pvec structure is defined in phigs.h as:

```
typedef struct {
        Pfloat  delta_x;    /* x magnitude */
        Pfloat  delta_y;    /* y magnitude */
} Pvec;
```

FORTRAN Input Parameters

N　Number of points passed in the PXA, PYA, and PZA arrays.

PXA　An array of N real values containing *x* coordinates in MC.

PYA　An array of N real values containing *y* coordinates in MC.

PZA　An array of N real values containing *z* coordinates in MC.

PRIMID
　The function identifier for this generalized drawing primitive is PUGDPAELLPARCCL3, which is defined in phigs77.h to be -13.

LDR　The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
　A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

　The arguments passed to PACK DATA RECORD when *PRIMID* = PUGDPAELLPARCCL3 should be as follows:

IL	The number of integers = 1.
IA	Contains the arc close type, which is either 0 (PARCCLPIE) or 1 (PARCCLCHORD).
RL	The number of real values = 9.
RA	Contains an array of RL real values. Element 1 is the *x* coordinate of the center of the annotation elliptical arc close 3 in MC. Element 2 is the *y* coordinate of the center of the annotation elliptical arc close 3 in MC. Element 3 is the *z* coordinate of the center of the annotation elliptical arc close 3 in MC. Element 4 is the *x* component of the elliptical arc close 3 major axis in NPC. Element 5 is the *y* component of the elliptical arc close 3 major axis in NPC. Element 6 is the *x* component of the elliptical arc close 3 minor axis in NPC. Element 7 is the *y* component of the elliptical arc close 3 minor axis in NPC. Element 8 is the start angle of the annotation elliptical arc close 3 in radians. Element 9 is the end angle of the annotation elliptical arc close 3 in radians.
SL	The number of strings = 0.

Execution

An annotation elliptical arc close 3 is a planar area defined by a center point, a major axis, a minor axis, a start angle, and an end angle. The Ellipse Local Coordinate (ELC) system has the same unit size as the NPC system. It is defined with origin at the center point and *x* and *y* axes parallel to, and with the same direction as, the NPC *x* and *y* axes. The horizontal and vertical axes of the ELC system are defined by the major and minor axes, respectively.

The center point of the annotation elliptical arc 3 is specified in MC. The major and minor axes of the annotation elliptical arc close 3 are specified in ELC. The start and end angles are specified in radians with the ELC positive *x* axis as the origin. At traversal, all transformations affect the center point, but only the workstation transformation affects the other parameters of the annotation elliptical arc close 3. An annotation elliptical arc close 3 greater than 360 degrees is drawn as a filled annotation ellipse 3.

The current values of all fill area set attributes will be applied to the annotation elliptical arc close 3 when it is drawn.

Annotation elliptical close 3 is rendered according to the *current curve approximation criteria*.

The array of points passed to this subroutine in addition to the GDP data record is not used by the annotation elliptical arc close 3 GDP.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -13 structure element. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
interior style	interior style ASF
interior style index	interior style index ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
edge index	
depth cue index	
name set	

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
See *Edge Attributes* in *Index*
See *Interior Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -14 – create 3D GDP image

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3   *point_list;   array of points
Pint           gdp3_id;       gdp function identifier
Pgdp_data3     *gdp_data;     data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER         N                        number of points (>= 0)
REAL            PXA(*), PYA(*), PZA(*)   coordinates of points (MC)
INTEGER         PRIMID                   GDP identifier
INTEGER         LDR                      dimension of data record array
CHARACTER*80    DATREC(LDR)              data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -14 creates the image structure element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs. the *x*, *y*, and *z* values in

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_IMAGE3, which is defined in phigs.h to be -14.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u14, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u14 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u14 {
                Ppoint3      ref_pt;       /* reference point in MC */
                Pint_size    dim;          /* dimensions of array */
                Pint         colr_type;    /* indirect, RGB, CIE, HSV, HLS */
                union {
                             char          *chars;   /* 8-bit indices */
                             Pint          *ints;    /* 32-bit integers */
                } colrs;
        } gdp3_u14;          /* image */
} Pgdp_data3;   /* image */
```

ref_pt A component that specifies the top left corner of the image. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

dim A pointer to a Pint_size structure specifying the dimensions of the image in pixels. The Pint_size structure is defined in phigs.h as follows:

```
typedef struct {
        Pint    size_x;    /* dimension along x */
        Pint    size_y;    /* dimension along y */
} Pint_size;
```

colr_type
Specifies the colour type used by the input data.

0	PINDIRECT	*Colour index specified*
1	PMODEL_RGB	*Red, green, and blue*
2	PMODEL_CIELUV	CIE *colour type*
3	PMODEL_HSV	*Hue, saturation, and value*
4	PMODEL_HLS	*Hue, lightness, and saturation*

colrs *colrs* is a union of pointers to the appropriate image buffer, depending on *colr_type*. The image colour data must be organized scanline by scanline. If *colr_type* is PINDIRECT, *colrs*→*chars* points to the 8-bit indexed colour data. In this case, each scanline must be padded to a 32-bit boundary. If *colr_type* is PMODEL_RGB, *colrs*→*ints* points to the 32-bit XBGR colour data. True colour 32-bit integers are packed with the true colour triplet in the lower 24 bits; that is, XBGR. Each of B, G, and R are an 8-bit index in the range of 0 to 255.

Example For example, to take the image from a 24-bit memory pixrect and display it in a PHIGS workstation, you could do the following:

```
Pixrect *mem_pr;
Pgdp_data3    image_gdp3;

load_image ( x, y, z, struct_id )
Pfloat x, y, z;
Pint struct_id;
{
```

```
                    image_gdp3.gdp3_u14.ref_pt.x = x;
                    image_gdp3.gdp3_u14.ref_pt.y = y;
                    image_gdp3.gdp3_u14.ref_pt.z = z;

                    image_gdp3.gdp3_u14.dim.size_x = img_pr->pr_size.x;
                    image_gdp3.gdp3_u14.dim.size_y = img_pr->pr_size.y;

                    image_gdp3.gdp3_u14.colr_type = PMODEL_RGB;

                    image_gdp3.gdp3_u14.colrs.ints =
                        (Pint *)(((struct mpr_data *)img_pr->pr_data)->md_image);

                    popen_struct(struct_id); {
                        pgdp3((Ppoint_list *)0, PUGDP_IMAGE3, &image_gdp3);
                    }
                    pclose_struct();
        }
```

Note: This passes a direct pointer to the pixrect's image data. When using a device pixrect, the data would need to be memory mapped in order to directly access the data.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPIMG3, which is defined in phigs77.h to be –14.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A generalized drawing primitive 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID is PUGDPIMG3 should be as follows:

IL The number of integers.

IA An array of integers.

The first three elements of IA are always constant.

Element 1 is the *x* dimension of the image in pixels.

Element 2 is the *y* dimension of the image in pixels.

Element 3 is the colour type. This specifies the colour type for the true colour array.

0	PINDIR	*Colour index specified*
1	PRGB	*Red, green, and blue*
2	PCIE	*CIE colour type*
3	PHSV	*Hue, saturation, and value*
4	PHLS	*Hue, lightness, and saturation*

RL	The number of reals.
RA	An array of reals. The first three elements of RA are always constant. Element 1 is the *x* coordinate of the top left corner of the image in MC. Element 2 is the *y* coordinate of the top left corner of the image in MC. Element 3 is the *z* coordinate of the top left corner of the image in MC.
SL	The number of strings.
LSTR	The length of strings.
STR	The character data contained in the specified elements.

Execution

An image is defined by a reference point, *x* and *y* dimensions, and a series of colours. The top left corner of the image is specified by the reference point in MC. The *x* and *y* dimensions specify the width and height of the image in pixels. At traversal, all transformations only affect the reference point.

The array of points passed to this subroutine in addition to the generalized drawing primitive data record is not used by the image generalized drawing primitive.

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
GENERALIZED DRAWING PRIMITIVE 3 (3P)
GENERALIZED DRAWING PRIMITIVE -14 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE 3 -15 – create 3D GDP rectangular grid 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( point_list, gdp3_id, gdp_data )
Ppoint_list3    *point_list;    array of points
Pint            gdp3_id;        gdp function identifier
Pgdp_data3      *gdp_data;      data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER         N                       number of points (>= 0)
REAL            PXA(*), PYA(*), PZA(*)  coordinates of points (MC)
INTEGER         PRIMID                  GDP 3 identifier
INTEGER         LDR                     dimension of data record array
CHARACTER*80    DATREC(LDR)             data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -15 creates the 3D rectangular grid 3 structure element. A rectangular grid 3 is a set of polymarkers uniformly spaced in the *x* and *y* dimensions, with *z* equal to the *z* component of the origin of the grid.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_RECT_GRID3, which is defined in phigs.h to be -15.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u15, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u15 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u15 {
                Ppoint3   origin;   /* center point */
                Pfloat    dx;       /* spacing alongs the x axis */
                Pfloat    dy;       /* spacing alongs the y axis */
        } gdp3_u15;         /* rectangular grid */
} Pgdp_data3;
```

The origin component specifies the center of the rectangular grid. It is a Ppoint3 structure containing *x*, *y*, and *z* coordinates specified in MC. The *dx* and *dy* components specify the uniform spacing in the *x* and *y* dimensions respectively in MC. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPRECTGRID3, which is defined in phigs77.h to be -15.

LDR The dimension of *DATREC* (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by *PRIMID*.

The arguments passed to PACK DATA RECORD when *PRIMID* = PUGDPRECTGRID3 should be as follows:

IL The number of integers = 0.

RL The number of real values = 5.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the origin of the rectangular grid in MC.

Element 2 is the *y* coordinate of the origin of the rectangular grid in MC.

Element 3 is the *z* coordinate of the origin of the rectangular grid in MC.

Element 4 is the *x* component of the rectangular grid spacing in MC.

Element 5 is the *y* component of the rectangular grid spacing in MC.

SL The number of strings = 0.

Execution

A 3D rectangular grid is a planar area defined by an origin and spacings in both *x* and *y* dimensions. The Grid Local Coordinate (GLC) system is located at the origin point with its *x* and *y* axes in parallel to the *x* and *y* axes of the MC.

When the structure is traversed, starting from the origin, the rectangular grid GDP 3 element places a marker at each of the GLC coordinate locations in specified by the *dx* and *dy*.

The coordinates used to specify the GLC are MCs. These can be any coordinate units that are convenient to the application. At traversal, these coordinate values are transformed by the current local and global modelling transformations, the view representation selected by the current view index, and the workstation transformation current on the workstation to which the structure is posted. However, since the polymarker has no geometric attributes, transformations affect only the grid position but not the appearance of the marker glyph in which the grid is constructed.

The current values of all polymarker attributes are applied to the 3D rectangular grid.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -15 structure element. The Aspect Source Flags (ASFs) describe where to access the output display attributes. These attributes can come directly from the traversal state list or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

marker type	marker type ASF
marker size scale factor	marker size scale factor ASF
polymarker colour index	polymarker colour index ASF
polymarker index	
polymarker colour	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

POLYMARKER (3P)
POLYMARKER 3 (3P)
GENERALIZED DRAWING PRIMITIVE -15 (3P)
GENERALIZED DRAWING PRIMITIVE -16 (3P)
GENERALIZED DRAWING PRIMITIVE 3 -16 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE 3 -16 – create a 3D GDP radial grid 3

SYNOPSIS

C Syntax

```
void
pgdp3 ( points, gdp3_id, gdp_data )
Ppoint_list3     *point_list;     array of points
Pint             gdp3_id;         gdp function identifier
Pgdprec3         *gdp_data;       data record pointer
```

FORTRAN Syntax

```
SUBROUTINE pgdp3 ( N, PXA, PYA, PZA, PRIMID, LDR, DATREC )
INTEGER           N                          number of points (>= 0)
REAL              PXA(*), PYA(*), PZA(*)     coordinates of points (MC)
INTEGER           PRIMID                     GDP 3 identifier
INTEGER           LDR                        dimension of data record array
CHARACTER*80      DATREC(LDR)                data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -16 creates the radial grid 3 structure element. A radial grid 3 is a set of polymarkers of uniform radial and angular distance from each other in the *x* and *y* dimensions with *z* equal to the *z* component of the center of the grid.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_RAD_GRID3, which is defined in phigs.h to be -16.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u16, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u16 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_16 {
                Ppoint3     center;     /* center point */
                Pfloat      d_angle;    /* spacing angle between rays */
                Pfloat      d_dist;     /* spacing between points on a ray */
        } gdp3_u16;         /* radial grid */
} Pgdp_data3;
```

The center component specifies the center of the radial grid. It is a Ppoint3 structure containing *x*, *y* and *z* coordinates specified in MC. Ppoint3 is defined in phigs.h as:

```
typedef struct {
        Pfloat   x;    /∗ x coordinate ∗/
        Pfloat   y;    /∗ y coordinate ∗/
        Pfloat   z;    /∗ z coordinate ∗/
} Ppoint3;
```

The dangle specifies the angle between rays of the grid. The ddist specifies the radial distance between points on a given ray in MC.

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPRADGRID3, which is defined in phigs77.h to be -16.

LDR The dimension of *DATREC (returned by the PACK DATA RECORD subroutine).*

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when *PRIMID = PUGDPRADGRID3 should be as follows:*

IL The number of integers = 0.

RL The number of real values = 5.

RA Contains an array of RL real values.

Element 1 is the *x* coordinate of the center of the radial grid in MC.

Element 2 is the *y* coordinate of the center of the radial grid in MC.

Element 3 is the *z* coordinate of the center of the radial grid in MC.

Element 4 is the angle between rays.

Element 5 is the radial distance between points on a given ray in MC.

SL The number of strings = 0.

Execution

A 3D radial grid 3 is a planar area defined by a center, a radius, and the spacing along the ray. The Grid Local Coordinate (GLC) system is located at the center point on the plane with its *x* and *y* axes in parallel to the *x* and *y* axes of the MC.

The radial grid uses the positive *x*-axis of the GLC as the base ray. Polymarkers are uniformly spaced with specified *ddist* along this axis. By using the specified center point as center, it sweeps the base ray in a counterclockwise direction, for every multiple of the specified *dangle*, thus defining a ray for each one until the remainder angle is less than or equal to the *dangle*. (The number of rays for the radial grid is always less than or equal to 2∗PI / *dangle*.)

A full or an arc section of the radial grid is drawn when the structure is traversed, depending on whether or not the center is inside the view volume.

The coordinates used to specify the GLC are MCs. These can be any coordinate units that are convenient to the application. At traversal, these coordinate values are transformed by the current local and global modelling transformations, the view representation selected by the current view index, and the workstation transformation current on the workstation to which the structure is posted. However, since the polymarker has no geometric attributes, transformations affect only the grid position, but not the appearance, of the marker glyph in which the grid is constructed.

The current values of all polymarker attributes will be applied to the 3D radial grid 3.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -16 structure element. The Aspect Source Flags (ASFs) describe where to access the output display attributes. These attributes can come directly from the traversal state list or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

marker type	marker type ASF
marker size scale factor	marker size scale factor ASF
polymarker colour index	polymarker colour index ASF
polymarker index	
polymarker colour	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

POLYMARKER (3P)
POLYMARKER 3 (3P)
GENERALIZED DRAWING PRIMITIVE -15 (3P)
GENERALIZED DRAWING PRIMITIVE -16 (3P)
GENERALIZED DRAWING PRIMITIVE -15 (3P)

NAME

GENERALIZED DRAWING PRIMITIVE 3 -17 – create structure element specifying 3D mono-encoded text primitive

SYNOPSIS

C Syntax

void
pgdp3 (point_list, gdp3_id, gdp_data)

Ppoint_list3	∗point_list;	*array of points*
Pint	gdp3_id;	*gdp function identifier*
Pgdp_data3	∗gdp_data;	*data record pointer*

FORTRAN Syntax

SUBROUTINE pgdp3 (N, PXA, PYA, PZA, PRIMID, LDR, DATREC)

INTEGER	N	*number of points (>=0)*
REAL	PXA(∗), PYA(∗), PZA(∗)	*coordinates of points (MC)*
INTEGER	PRIMID	*GDP 3 identifier*
INTEGER	LDR	*dimension of data record array*
CHARACTER∗80	DATREC(LDR)	*data record*

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

The GENERALIZED DRAWING PRIMITIVE 3 -17 subroutine puts a structure element containing a three dimensional mono-encoded text primitive into the currently open structure.

The application uses the mono-encoded text primitive when mixed character sets are required or when a single character set other than ASCII is required. The mono-encoded text primitive is an I-String, which is a series of substrings, each using a single distinct character set.

When the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE 3 -17 subroutine is inserted in the open structure after the element pointed to by the current *element pointer*. When the current edit mode is REPLACE, the new GENERALIZED DRAWING PRIMITIVE 3 -17 element replaces the element in the structure pointed to by the element pointer. In either case, the element pointer is updated to point to the new GENERALIZED DRAWING PRIMITIVE 3 -17 element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u17, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u17 is defined in phigs.h as:

```
typedef union {
```

```
            struct Pgdp3_u17 {
                    Ppoint3             text_pt;     /* text point */
                    Pvec3               dir[2];      /* direction vectors*/
                    Phigs_i_string      i_string;    /* I-String */
            } gdp3_u17;
    } Pgdp_data3;
```

text_pt A Ppoint3 structure containing the *x*, *y*, and *z* coordinates that locate the mono-encoded text. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;      /* x coordinate */
        Pfloat      y;      /* y coordinate */
        Pfloat      z;      /* z coordinate */
} Ppoint3;
```

dir An array of two Pvec3 structures containing direction vectors. The Pvec3 structure is defined in phigs.h as:

```
typedef struct {
        Pfloat   delta_x;   /* x magnitude */
        Pfloat   delta_y;   /* y magnitude */
        Pfloat   delta_z;   /* z magnitude */
} Pvec3;
```

i_string contains a list of mono-encodings. Phigs_i_string is defined in phigs.h as follows:

```
typedef struct {
   Pint                   num_encodings;     /* number of
                                                encodings */
   Phigs_mono_encoding    *mono_encodings;   /* list of
                                                encodings */
} Phigs_i_string;
```

Each *mono_encoding* (substring) contains a character set, an encoding state, and a string. Phigs_mono_encoding is defined in phigs.h as follows:

```
typedef struct {
        Pint    char_set;           /* character set */
        char    encoding_state;     /* encoding state */
        char    *string;            /* character string */
} Phigs_mono_encoding;
```

char_set
Valid values for *char_set* are defined in phigs.h as follows:

PCS_ASCII	0
PCS_GREEK	−1
PCS_SYMBOL	−2
PCS_CARTOGRAPHIC	−3
PCS_KANJI	−4
PCS_KATAKANA	−5

encoding_state
Is used to encode a character encoding method (for example, EUC, CT) within a mono-encoding. This parameter is not actually used by the SunPHIGS library. However, it can be used in conjunction with the INQUIRE ELEMENT CONTENT function to restore mono-encoded text to its original encoding method.

string A pointer to the character string, char_count characters long.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_MONO_ENC_TXT3, which is defined in phigs.h to be -17.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPMONOENCTXT3, which is defined in phigs77.h to be -17.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID = PUGDPMONOENCTXT should be as follows:

IL The number of integers = (SL × 2).

IA Contains array of IL integer values. For each substring, there is the character set and an encoding state.

Valid values for the character set are defined in phigs77.h as follows:

PCSASCII	0
PCSGREEK	−1
PCSSYMBOL	−2
PCSCARTOGRAPHIC	−3
PCSKANJI	−4

PCSKATAKANA	–5

RL	The number of real values = 9.
RA	Contains array of RL real values.
	Elements 1, 2, and 3 are the *x*, *y*, and *z* coordinates of the point locating the I-String in MC.
	Elements 4, 5, and 6 are the *x*, *y*, and *z* components of the direction vector defining the text's *x* axis.
	Elements 7, 8, and 9 are the *x*, *y*, and *z* components of the direction vector defining the text's *y* axis.
SL	The number of mono-encoded substrings.
LSTR	Array of length SL containing lengths of strings in STR.
STR	Contains array of SL substrings.

Execution

When the structure is traversed, the GENERALIZED DRAWING PRIMITIVE 3 -17 element draws the specified I-String on the plane in the Modelling Coordinate system (MC) that is defined by the text point and the two direction vectors. The text point defines the origin of the Text Local Coordinate (TLC) system, the first direction vector defines the positive *x* axis, and the second direction vector defines the positive *y* axis. Only the directions, not the lengths, of these vectors are relevant.

The precise position of the I-String is defined in relation to this plane by the current values of the text primitive attributes CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT.

The text point is specified in MC. The mono-encoded text 3 primitive is subject to the current transformations in the transformation pipeline from the MC system to the workstation display.

Other aspects of the appearance of the I-String are controlled by the attributes TEXT FONT, TEXT PRECISION, CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, and TEXT COLOUR INDEX. These attributes are applied to the entire I-string. They cannot be changed per substring.

The array of points that is passed to this subroutine in addition to the generalized drawing primitive data record is not used by the mono-encoded text generalized drawing primitive.

C Example

To render an I-String consisting of four ASCII characters, followed by four Greek characters, you could do the following:

```
extern char *ascii_chars, *greek_chars;
draw_text(x, y, z)
Pfloat  x, y z;
{
        Pgdp_data3      text_gdp;
        Phigs_mono_encoding     substrings[2];
```

```
                substrings[0] = ascii_chars;
                substrings[1] = greek_chars;
                text_gdp.gdp3_u17.text_pt.x = x;
                text_gdp.gdp3_u17.text_pt.y = y;
                text_gdp.gdp3_u17.text_pt.z = z;
                text_gdp.gdp3_u17.dir[0].delta_x = 1.0;
                text_gdp.gdp3_u17.dir[0].delta_y = 0.0;
                text_gdp.gdp3_u17.dir[0].delta_z = 0.0;
                text_gdp.gdp3_u17.dir[1].delta_x = 0.0;
                text_gdp.gdp3_u17.dir[1].delta_y = 1.0;
                text_gdp.gdp3_u17.dir[1].delta_z = 0.0;
                text_gdp.gdp3_u17.i_string.num_encodings = 2;
                text_gdp.gdp3_u17.i_string.mono_encodings = substrings;
                pgdp3((Ppoint_list3 *)NULL, PUGDP_MONO_ENC_TXT3, &text_gdp);
        }
```

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -17 structure element. The Aspect Source Flags (ASF)s describe where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

text font	text font ASF
text precision	text precision ASF
character expansion factor	character expansion factor ASF
character spacing	character spacing ASF
text colour	text colour index ASF
character height	
character up vector	
text path	
text alignment	
text index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
GENERALIZED DRAWING PRIMITIVE -17 (3P)
ESCAPE -10 (3P)
ESCAPE -11 (3P)
ESCAPE -12 (3P)
TEXT 3 (3P)

INTRO INTERNATIONALIZATION (7P)
See *Text Attributes* in *Index*

NAME

GENERALIZED DRAWING PRIMITIVE 3 -18 – create structure element specifying 3D mono-encoded annotation text relative primitive

SYNOPSIS

C Syntax

void
pgdp3 (point_list, gdp3_id, gdp_data)

Ppoint_list3	∗point_list;	*array of points*
Pint	gdp3_id;	*gdp function identifier*
Pgdp_data3	∗gdp_data;	*data record pointer*

FORTRAN Syntax

SUBROUTINE pgdp3 (N, PXA, PYA, PYZ, PRIMID, LDR, DATREC)

INTEGER	N	*number of points (>=0)*
REAL	PXA(∗), PYA(∗), PZA(∗)	*coordinates of points (MC)*
INTEGER	PRIMID	*GDP 3 identifier*
INTEGER	LDR	*dimension of data record array*
CHARACTER∗80	DATREC(LDR)	*data record*

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

The GENERALIZED DRAWING PRIMITIVE 3 -18 subroutine puts a structure element containing a three-dimensional mono-encoded annotation text relative primitive into the currently open structure.

The mono-encoded annotation text 3 primitive is used by the application when mixed character sets are required or when a single character set other than ASCII is required. The mono-encoded annotation text 3 primitive is an I-String, which is a series of substrings, each using a single distinct character set.

When the current edit mode is INSERT, the structure element created by the GENERALIZED DRAWING PRIMITIVE 3 -18 subroutine is inserted in the open structure after the element pointed to by the current *element pointer*. When the current edit mode is REPLACE, the new GENERALIZED DRAWING PRIMITIVE 3 -18 element replaces the element in the structure pointed to by the element pointer. In either case, the element pointer is updated to point to the new GENERALIZED DRAWING PRIMITIVE 3 -18 element.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_MONO_ENC_ATXT_REL3, which is defined in phigs.h to be -18.

gdp_data

A pointer to a Pgdp_data3 union, Pgdp3_u18, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u18 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u18 {
                Ppoint3          ref_pt;           /∗ reference point ∗/
                Pvec3            anno_offset;      /∗ annotation offset ∗/
                Phigs_i_string   i_string;         /∗ annotation I-String ∗/
        } gdp3_u18;
} Pgdp_data3;
```

ref_pt The *x* and *y* coordinates in MC that locate the mono-encoded text. Ppoint3 is defined in phigs.h as follows:

```
struct {
        Pfloat      x;       /∗ x coordinate ∗/
        Pfloat      y;       /∗ y coordinate ∗/
        Pfloat      z;       /∗ z coordinate ∗/
} Ppoint3;
```

anno_offset
A Pvec3 structure containing the *x*, *y*, and *z* coordinates of the offset of the I-String from the transformed reference point. The annotation offset specifies an offset in NPC. Pvec3 is defined in phigs.h as follows:

```
typedef struct {
        Pfloat   delta_x;  /∗ x magnitude ∗/
        Pfloat   delta_y;  /∗ y magnitude ∗/
        Pfloat   delta_z;  /∗ z magnitude ∗/
} Pvec3;
```

i_string Contains a list of mono-encodings. Phigs_i_string is defined in phigs.h as follows:

```
typedef struct {
    Pint                     num_encodings;      /∗ number of
                                                    encodings ∗/
    Phigs_mono_encoding      ∗mono_encodings;    /∗ list of
                                                    encodings ∗/
} Phigs_i_string;
```

Each *mono_encodings* (substring) contains a character set, an encoding state, and a string. Phings_mono_encoding is defined in phigs.h as follows:

```
typedef struct {
        Pint    char_set;           /∗ character set ∗/
        char    encoding_state;     /∗ encoding state ∗/
        char    *string;                /* character string */
} Phigs_mono_encoding;
```

char_set

Valid values for *char_set* are defined in phigs.h as follows:

PCS_ASCII	0
PCS_GREEK	–1
PCS_SYMBOL	–2
PCS_CARTOGRAPHIC	–3
PCS_KANJI	–4
PCS_KATAKANA	–5

encoding_state

Is used to encode a character encoding method (that is, EUC, CT) within a mono-encoding. This parameter is not actually used by the SunPHIGS library. However, it can be used in conjunction with the INQUIRE ELEMENT CONTENT function to restore mono-encoded text to its original encoding method.

i_string A pointer to the character string, char_count characters long.

FORTRAN Input Parameters

N Number of points passed in the PXA and PYA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PRIMID

The function identifier for this generalized drawing primitive is PUGDPMONOATXTR3, which is defined in phigs77.h to be -18.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC

A GDP data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when *PRIMID* = PUGDPMONOENCATXT should be as follows:

IL The number of integers = (SL × 2).

IA Contains array of IL integer values. For each substring, there is the character set and an encoding state.

Valid values for the character set are defined in phigs77.h as follows:

PCSASCII	0
PCSGREEK	–1
PCSSYMBOL	–2
PCSCARTOGRAPHIC	–3
PCSKANJI	–4
PCSKATAKANA	–5

RL	The number of real values = 6.
RA	Contains array of RL real values. Element 1 is the *x* coordinate of the point locating the annotation I-String in MC. Element 2 is the *y* coordinate of the point locating the annotation I-String in MC. Element 3 is the *z* coordinate of the point locating the annotation I-String in MC. Element 4 is the *x* offset in NPC locating the I-String relative to Element 1. Element 5 is the *y* offset in NPC locating the I-String relative to Element 2. Element 6 is the *z* offset in NPC locating the I-String relative to Element 3.
SL	The number of mono-encoded substrings.
LSTR	Array of SL integers specifying the lengths of the mono-encoded strings.
SA	Contains array of SL substrings.

Execution

When the structure is traversed, the GENERALIZED DRAWING PRIMITIVE 3 -18 element draws the specified I-String on the plane in NPC defined by the reference point and the annotation offset. These parameters define a Text Local Coordinate (TLC) system in NPC. The annotation offset added to the transformed reference point defines the origin of the TLC system. The *x* and *y* axes of the TLC system are parallel to and have the same direction as the *x* and *y* axes of NPC. The precise position of the annotation I-String is defined in relation to this plane by the current values of the text primitive attributes CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT.

Other aspects of the appearance of the I-String are controlled by the attributes TEXT FONT, TEXT PRECISION, CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER EXPANSION FACTOR, CHARACTER SPACING and TEXT COLOUR INDEX. These attributes are applied to the entire I-string. They cannot be changed per substring.

The reference point is subject to the current transformations in the transformation pipeline from the MC system to the workstation display. The I-String itself is only subject to the transformations in the transformation pipeline from the NPC system to the workstation display.

The array of points passed to this subroutine in addition to the generalized drawing primitive data record is not used by the mono-encoded annotation text 3 generalized drawing primitive.

C Example

To render an I-String consisting of four ASCII characters, followed by four Greek characters, you could do the following:

```
        extern char *ascii_chars, *greek_chars;
        draw_text(x, y)
        Pfloat  x, y;
        {
                Pgdp_data3          text_gdp;
                Phigs_mono_encoding     substrings[2];
                substrings[0] = ascii_chars;
                substrings[1] = greek_chars;
                text_gdp.gdp3_u18.ref_pt.x = x;
                text_gdp.gdp3_u18.ref_pt.y = y;
                text_gdp.gdp3_u18.ref_pt.z = z;
                text_gdp.gdp3_u18.anno_offset.delta_x = 0.0;
                text_gdp.gdp3_u18.anno_offset.delta_y = 0.0;
                text_gdp.gdp3_u18.anno_offset.delta_z = 0.0;
                text_gdp.gdp3_u18.i_string.num_encodings = 2;
                text_gdp.gdp3_u18.i_string.mono_encodings = substrings;
                pgdp3((Ppoint_list3 *)NULL, PUGDP_MONO_ENC_ATXT_REL, &text_gdp);
        }
```

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -18 structure element. The Aspect Source Flags (ASF)s describe where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly by using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

text font	text font ASF
text precision	text precision ASF
character expansion factor	character expansion factor ASF
character spacing	character spacing ASF
text colour	text colour index ASF
annotation character height	
annotation character up vector	
annotation text path	
annotation text alignment	
text index	
depth cue index	
name set	

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED DRAWING PRIMITIVE 3 (3P)
GENERALIZED DRAWING PRIMITIVE -18 (3P)
ESCAPE -10 (3P)
ESCAPE -11 (3P)
ESCAPE -12 (3P)
ANNOTATION TEXT RELATIVE 3 (3P)
INTRO INTERNATIONALIZATION (7P)
See *Text Attributes* in Index

NAME

GENERALIZED DRAWING PRIMITIVE 3 -19 – create 3D GDP hinted NURBS surface

SYNOPSIS

C Syntax

void
pgdp3 (point_list, gdp3_id, gdp_data)

```
Ppoint_list3   *point_list;   array of points
Pint           gdp3_id;       gdp function identifier
Pgdp_data3     *gdp_data;     data record pointer
```

FORTRAN Syntax

SUBROUTINE pgdp3 (N, PXA, PYA, PZA, PRIMID, LDR, DATREC)

```
INTEGER        N                      number of points (>= 0)
REAL           PXA(*), PYA(*), PZA(*) coordinates of points (MC)
INTEGER        PRIMID                 GDP identifier
INTEGER        LDR                    dimension of data record array
CHARACTER*80   DATREC(LDR)            data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED DRAWING PRIMITIVE 3 -19 creates a structure element containing the definition of a hinted non-uniform B-spline surface. This element is the same as that created by NON-UNIFORM B-SPLINE SURFACE (3PP) except that hints may be specified which can dramatically improve the rendering performance for simple NURBS surfaces. The simple surfaces supported by this generalized drawing primitive include planes, spheres, cones, and cylinders. The surface can be rational or non-rational, and can include trimming curves.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

point_list
A pointer to a Ppoint_list3 structure. Currently, this parameter is ignored for all SunPHIGS-supported GDPs.

gdp3_id The function identifier for this generalized drawing primitive is PUGDP_HINTED_NURBS_SURF3, which is defined in phigs.h to be -19.

gdp_data
A pointer to a Pgdp_data3 union, Pgdp3_u19, containing the information needed to perform the function specified by *gdp3_id*. Pgdp3_u19 is defined in phigs.h as:

```
typedef union {
        struct Pgdp3_u19 {
                Pint            uorder;     /* U spline order */
                Pint            vorder;     /* V spline order */
                Pfloat_list     *uknots;    /* U knots */
                Pfloat_list          *vknots;       /* V knots */
                Prational            rationality;   /* rationality selector */
urf_hints       Ppoint_grid34        *grid;         /* grid of 3D or 4D
                                                        control points */
                Pint                 nloops;        /* number of trim curve
                                                        loops */
                Ptrimcurve_list      *tloops;       /* trim curve loops */
                Pnurb_surf_hints     *hints         /* surface shape hints */
        } gdp3_u19;                  /* hinted NURBS */
} Pgdp_data3;
```

uorder/vorder
Order of the surface in the *u* and *v* dimensions, respectively.

uknots/vknots
Pointers to Pfloat_list structures listing the knots for the *u* and *v* dimensions of the surface. Pfloat_list is defined in phigs.h as follows:

```
typedef struct {
        Pint            num_floats;     /* number of Pfloats in list */
        Pfloat          *floats;        /* list of floats */
} Pfloat_list;
```

rationality
The rationality selector. Prational is defined in phigs.h as follows:

```
typedef enum {
        PNON_RATIONAL = 0,
        PRATIONAL = 1
} Prational;
```

grid Pointer to a Ppoint_grid34 structure containing the surface control points. Ppoint_grid34 is defined in phigs.h as follows:

```
typedef struct {                                /* grid of 3D or 4D points,
                                                    [v_dim] [u_dim] */
   Ppcs_dims     num_points;                    /* number of points in each
                                                    dimension */
   union {
        Ppoint3             *point3d;           /* array of 3D points */
        Ppoint4             *point4d;           /* array of 4D points */
   } points;
} Ppoint_grid34;
```

Ppcs_dims is defined in phigs.h as:

```
typedef struct {
```

```
          Pint    u_dim;    /* dimension (number of divisions) along u */
          Pint    v_dim;    /* dimension (number of divisions) along v */
} Ppcs_dims;
```

Ppoint3 is defined in phigs.h as:

```
typedef struct {
          Pfloat    x;    /* x coordinate */
          Pfloat    y;    /* y coordinate */
          Pfloat    z;    /* z coordinate */
} Ppoint3;
```

Ppoint4 is defined in phigs.h as:

```
typedef struct {
          Pfloat    x;    /* x coordinate */
          Pfloat    y;    /* y coordinate */
          Pfloat    z;    /* z coordinate */
          Pfloat    w;    /* w coordinate */
} Ppoint4;
```

nloops Number of trimming curve loops.

tloops A pointer to an array of Ptrimcurve_list structures that specifies the trimming curve loops. Each of these structures contains the specification of an individual trimming curve loop (set of trimming curves). Ptrimcurve_list is defined in phigs.h as:

```
typedef struct {
          Pint            num_curves;    /* number of trimming curves
                                            in list */
          Ptrimcurve      *curves;       /* list of curves */
} Ptrimcurve_list;
```

Ptrimcurve is defined in phigs.h as:

```
typedef struct {
          Pedge_flag      visible;        /* curve visibility flag */
          Prational       rationality;    /* rationality */
          Pint            order;          /* curve order */
          Pint            approx_type     /* approximation type */
          Pfloat          approx_val      /* approximation value */
          Pfloat_list     knots;          /* curve knot vector */
          Pfloat          tmin, tmax;     /* curve parameter range */
          Ppoint_list23   cpts;           /* control points */
} Ptrimcurve;
```

Pedge_flag is defined in phigs.h as:

```
typedef enum {
        PEDGE_OFF,
        PEDGE_ON
} Pedge_flag;
```

Prational is defined in phigs.h as:

```
typedef enum {
        PNON_RATIONAL = 0,
        PRATIONAL = 1
} Prational;
```

Pfloat_list is defined in phigs.h as:

```
typedef struct {
        Pint      num_floats;    /* number of Pfloats in list */
        Pfloat    *floats;       /* list of floats */
} Pfloat_list;
```

Ppoint_list23 is defined in phigs.h as:

```
typedef struct {                                   /* list of 2D or 3D
                                                      points */
        Pint                    num_points;     /* number of points */
        union {
                Ppoint      *point2d;          /* array of 2D points */
                Ppoint3     *point3d;          /* array of 3D points */
        } points;
} Ppoint_list23;
```

The *point3d* member of the points union is used if the rationality member of Ptrimcurve is PRATIONAL; otherwise, the *point2d* member is used. Ppoint is defined in phigs.h as:

```
typedef struct {
        Pfloat    x;    /* x coordinate */
        Pfloat    y;    /* y coordinate */
} Ppoint;
```

Ppoint3 is defined in phigs.h as:

```
typedef struct {
        Pfloat    x;    /* x coordinate */
        Pfloat    y;    /* y coordinate */
        Pfloat    z;    /* z coordinate */
} Ppoint3;
```

hints If the surface is planar, spherical, cylindrical, or conical, hints about the shape of the surface may be supplied to optimize the rendering of the primitive. Pnurb_surf_hints is defined in phigs.h as follows:

```
typedef struct {
        Pnurb_surf_type    surf_type;    /∗ simple nurb surface
                                            type ∗/
        Pnurb_surf_geom    geom_desc;    /∗ simple nurb surface
                                            geom. ∗/
} Pnurb_surf_hints;
```

Pnurb_surf_type is defined in phigs.h as:

```
typedef enum {
        PHIGS_SURF_NURBS,
        PHIGS_SURF_PLANAR,
        PHIGS_SURF_CYLINDRICAL,
        PHIGS_SURF_CONICAL,
        PHIGS_SURF_SPHERICAL
} Pnurb_surf_type;
```

Pnurb_surf_geom is defined in phigs.h as:

```
typedef union {
        Pptnorm3  planar;    /∗ plane point and normal ∗/
        struct {
                Ppoint3      axial_pt;    /∗ point on cylinder axis ∗/
                Pvec3        axis_dir;    /∗ cylinder axis direction ∗/
                Pfloat       radius;      /∗ radius of cylinder ∗/
                Pnorm_flag   norm_flag;   /∗ surface normal
                                             direction ∗/
        } cylindrical;
        struct {
                Ppoint3      apex;        /∗ apex of cone ∗/
                Pvec3        axis_dir;    /∗ cone axis direction ∗/
                Pfloat       cone_angle;  /∗ angle between axis &
                                             side ∗/
                Pnorm_flag   norm_flag;   /∗ surface normal
                                             direction ∗/
        } conical;
        struct {
                Ppoint3      center;      /∗ center of sphere ∗/
                Pfloat       radius;      /∗ radius of sphere ∗/
                Pnorm_flag   norm_flag;   /∗ surface normal
                                             direction ∗/
        } spherical;
} Pnurb_surf_geom;
```

Pptnorm3 is defined in phigs.h as:

```
typedef struct {
```

```
                Ppoint3    point;    /∗ point coordinates ∗/
                Pvec3      norm;     /∗ normal ∗/
        } Pptnorm3;
```

Ppoint3 is defined in phigs.h as:

```
        typedef struct {
                Pfloat    x;     /∗ x coordinate ∗/
                Pfloat    y;     /∗ y coordinate ∗/
                Pfloat    z;     /∗ z coordinate ∗/
        } Ppoint3;
```

Pvec3 is defined in phigs.h as:

```
        typedef struct {
                Pfloat    delta_x;    /∗ x magnitude ∗/
                Pfloat    delta_y;    /∗ y magnitude ∗/
                Pfloat    delta_z;    /∗ z magnitude ∗/
        } Pvec3;
```

Pnorm_flag is defined in phigs.h as:

```
        typedef enum {
                PNORM_INWARDS,
                PNORM_OUTWARDS
        } Pnorm_flag;
```

FORTRAN Input Parameters

N Number of points passed in the PXA, PYA, and PZA arrays.

PXA An array of N real values containing *x* coordinates in MC.

PYA An array of N real values containing *y* coordinates in MC.

PZA An array of N real values containing *z* coordinates in MC.

PRIMID
The function identifier for this generalized drawing primitive is PUGDPHINTEDNURBSURF3, which is defined in phigs77.h to be –19.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GDP 3 data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by PRIMID.

The arguments passed to PACK DATA RECORD when PRIMID is PUGDPHINTEDNURBSURF3 should be as follows:

IL The number of integers in array IA = 25 + variable size of trimming curve data.

IA An array of IL integer values.

Element 1 is the surface *u* order.

Element 2 is the surface *v* order.

Element 3 is the rationality selector, which is either rational (PRAT) or nonrational (PNRAT).

Element 4 is the number of *u* knots.

Element 5 is the number of *v* knots.

Element 6 is the number of control points in the *u* dimension.

Element 7 is the number of control points in the *v* dimension.

Element 8 is the number of trimming curve loops.

Element 9 is the index of the first surface *u* knot in the real array RA.

Element 10 is the index of the first surface *v* knot in the real array RA.

Element 11 is the index of the first control point in the real array RA.

Element 12 is the starting index in IA of the list of the number of trimming curves per loop.

Element 13 is the starting index in IA of the list of trimming curve visibilities.

Element 14 is the starting index in IA of the list of trimming curve orders.

Element 15 is the starting index in IA of the list of trimming curve approximation types.

Element 16 is the starting index in IA of the list of trimming curve rationalities.

Element 17 is the starting index in IA of the list of trimming curve knot counts.

Element 18 is the starting index in IA of the list of trimming curve knot starting indices in the real array RA.

Element 19 is the starting index in IA of the list of trimming curve control point counts.

Element 20 is the starting index in IA of the list of trimming curve control point starting indices in the real array RA. The trimming curve control points are contiguously stored in the RA array, in the *u*, *v*, *w* order. The *w* coordinate is ignored if the rationality of that particular curve is PNRAT. Note that these starting indices address the one-dimensional RA array, unlike the indices in the TCSTRT argument of the PNUBSS function, which index into a two-dimensional array.

Element 21 is the starting index in RA of the list of trimming curve approximation values.

Element 22 is the starting index of the list of trimming curve lower parameter limits in the real array RA.

Element 23 is the starting index of the list of trimming curve upper parameter limits in the real array RA.

Element 24 is the type of the NURBS surface. Valid values are:

0	PSTNURBS	*Non-simple NURBS surface*
1	PSTPLANE	*Planar surface*
2	PSTCYLINDER	*Cylindrical surface*
3	PSTCONE	*Conical surface*
4	PSTSPHERE	*Spherical surface*

Element 25 is the direction of the surface normals which is either inwards (PNORMIN) or outwards (PNORMOUT).

Element 26 and onward are the variable length surface and trimming curve integer data.

RL The number of real numbers in array RA = 7 + IA(4) + IA(5) + 4∗IA(6)∗IA(7) + variable size of the trimming curve data.

RA The array of RL real values.

Element 1 is the *x* coordinate in MC of the point on the simple NURBS surface.

Element 2 is the *y* coordinate in MC of the point on the simple NURBS surface.

Element 3 is the *z* coordinate in MC of the point on the simple NURBS surface.

Element 4 is the *x* component in MC of the direction vector of the simple NURBS surface.

Element 5 is the *y* component in MC of the direction vector of the simple NURBS surface.

Element 6 is the *z* component in MC of the direction vector of the simple NURBS surface.

Element 7 is the angle (for a cone) or radius (for a sphere or cylinder) in MC of the simple NURBS surface.

Elements (IA(9)) through (IA(9) + IA(4)–1) are the *u* knot values.

Elements (IA(10)) through (IA(10) + IA(5)–1) are the *v* knot values.

Elements (IA(11)) through (IA(11) + 4∗IA(6)∗IA(7)–1) are the surface control points, in *x*, *y*, *z*, *w* order. The *w* coordinate is ignored if the surface is non-rational; that is, if IA[3] = PNRAT.

Element (IA(11)) is the *x* coordinate of the first control point.

Element (IA(11) + 1) is the *y* coordinate of the first control point, and so on.

The rest of RA contains the variable-length trimming curve data. The indices packed in the IA array must reflect the structure of this data.

SL	The number of strings (0).
LSTR	The array of string lengths (empty).
STR	The array of characters in the strings (empty).

Execution

A hinted NURBS is defined by two knot vectors, a grid of control points, an optional list of trimming curves, and hints about the shape of the surface. The surface *u* and *v* orders must be positive integers. Surfaces of unsupported orders in either dimension are displayed by drawing the control grid, that is, the corresponding surface of order 2 in both dimensions.

The *u* and *v* knots must each form a non-decreasing sequence of numbers.

The *rationality selector* parameter may have the enumerated value Rational or Nonrational. When Rational is specified, the control points must be specified as 4D homogeneous modelling coordinates. When Nonrational is specified, the control points are 3D PHIGS modelling coordinates.

The number of control points in the *u* and *v* directions must be at least as large as the corresponding order. The number of control points in each direction plus the corresponding spline order must be equal to the corresponding number of knots.

At structure traversal time, a GENERALIZED DRAWING PRIMITIVE 3 -19 primitive is drawn using the attributes that apply to FILL AREA SET 3 primitives, plus the extended interior attributes. See SET INTERIOR REPRESENTATION PLUS for a listing of these attributes.

Trimming Curves

The surface parameter area to be rendered may be defined by trimming curves. Trimming curves override the specified surface parameter range values, and the parameter range values are ignored during traversal if trimming curves are specified.

The trimming definition of a surface consists of a list of loops. Each of these loops is a list of one or more trimming curves. Each trimming curve is a non-uniform rational B-spline curve defined in the 2D parameter space of the surface. The curves of a loop connect in a head-to-tail fashion. Each loop must be explicitly closed. Closure is assumed and not checked.

The restrictions on trimming curves are the same as those for non-uniform B-spline curves, and are specified in the function NON-UNIFORM B-SPLINE CURVE. Only the trimming curve's control polygon is used when it is of unsupported order or of order 1.

Trimming loops may not go outside the parameter space of the surface. The trimming curves associated with a surface may touch each other only at their end points. No other pair of trimming curves may connect at that same point. A trimming curve may not be self-touching or self-intersecting.

If trimming curves are specified, then they define the edges of the surface. Each trimming curve has a flag that controls its visibility. These visibility flags have lower precedence than the edge flag attribute and are ignored if the edge flag attribute is set to OFF.

SunPHIGS Extensions restricts trimming loops of a single surface primitive to a certain canonical form. This form must redundantly identify the interior parameter area in two different ways: according to the odd winding and the curve handedness rules.

Hints

For simple NURBS surfaces such as planes and cylinders, a major performance improvement can be expected in lighting and shading if hints are supplied about the geometry of the simple surface. A planar surface is defined by a point on the plane and a normal. A cylindrical surface is defined by an axis, a point on the axis, and a radius. A conical surface is defined by its apex, axis, and an angle between the axis and the side. Finally, a spherical surface is defined by its center and a radius. The norm_flag for cylindrical, conical, and spherical surfaces indicates whether normals to the surface point outwards (PNORM_OUTWARDS) or inwards (PNORM_INWARDS).

All the NURBS parameters to this GDP must always be provided even though geometry hints are specified. It is the application's responsibility to ensure that provided hints do not conflict with the geometry defined by the NURBS parameters. If this is not the case, lighting may be incorrect.

Attributes Applied

The attributes listed below are used to display the GENERALIZED DRAWING PRIMITIVE 3 -19 primitive when the structure is traversed. The Aspect Source Flags (ASFs) tell where to access the output display attributes. These attributes can come directly from the traversal state list, or they can be accessed indirectly, using the appropriate index in the traversal state list and the corresponding bundled representation in the workstation state list.

interior colour	interior colour index ASF
back interior colour	back interior colour ASF
interior style	interior style ASF
back interior style	back interior style ASF
interior style index	interior style index ASF
back interior style index	back interior style index ASF
interior shading method	interior shading method ASF
back interior shading method	back interior shading method ASF
interior reflectance characteristics	interior reflectance characteristics ASF
back interior reflectance characteristics	back interior reflectance characteristics ASF
reflectance properties	reflectance properties ASF
back reflectance properties	back reflectance properties ASF
interior index	
edge colour	edge colour index ASF
edge flag	edge flag ASF
edgetype	edgetype ASF
edgewidth scale factor	edgewidth scale factor ASF
surface approximation criteria	surface approximation criteria ASF
edge index	
face distinguishing mode	

face culling mode
depth cue index
light source state
name set

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

600　Ignoring function, not enough control points for specified order

601　Ignoring function, knot sequence is not nondecreasing

602　Ignoring function, order is inconsistent with number of knots and control points specified

615　Ignoring function, parameter range is inconsistent with knots

SEE ALSO

GENERALIZED DRAWING PRIMITIVE (3P)
GENERALIZED DRAWING PRIMITIVE 3 (3P)
SET SURFACE APPROXIMATION CRITERIA (3PP)
NON-UNIFORM B-SPLINE CURVE (3PP)
NON-UNIFORM B-SPLINE SURFACE (3PP)
SET INTERIOR REPRESENTATION PLUS (3PP)
INQUIRE CURVE AND SURFACE FACILITIES (3PP)
INTRO PHIGS PLUS (3PP)

NAME

GENERALIZED STRUCTURE ELEMENT – create generalized structure element

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint        id;      gse identifier
Pgse_data   *gse;    gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER         GSEID          GSE identifier
INTEGER         LDR            dimension of data record array
CHARACTER*80    DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT creates a structure element that can be used to:

- Set the highlight colour index
- Set the slant angle for text
- Set the slant angle for annotation text
- Set the polygon type (self-intersecting or not)
- Control width-1 stroke anti-aliasing
- Control the width of vector strokes used to draw vector text
- Control the appearance of wide stroke end-caps
- Control the appearance of wide stroke joints
- Set the silhouette edge flag

Each generalized structure element has its own page in the reference manual. The name of the page is generalized structure element-xx, where xx is replaced by the GSE identifiers; for example, GENERALIZED STRUCTURE ELEMENT -1.

Note: Support for GSEs is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The identifier of the generalized structure element to insert. Recognized identifiers are defined in phigs.h as:

−1	PUGSE_HIGHLIGHT_COLOUR_INDEX	*Set highlight color index*
−4	PUGSE_TEXT_SLANT_ANGLE	*Set text slant angle*
−5	PUGSE_ATEXT_SLANT_ANGLE	*Set annotation text slant angle*
−6	PUGSE_PGON_TYPE	*Set current polygon type*
−7	PUGSE_SET_STROKE_AA_ID	*Set stroke anti-aliasing ID*
−8	PUGSE_SET_TEXT_STROKE_WIDTH_ SCALE_FACTOR	*Set text stroke width scale factor*

–9	PUGES_SET_STROKE_END_CAP	*Set stroke end cap*
–10	PUGSE_SET_STROKE_JOIN	*Set stroke join*
–11	PUGSE_SIL_EDGE_FLAG	*Set current silhouette edge flag*

gse A pointer to a Pgse_data union containing the information needed to perform the function specified by *id*. Pgse_data is defined in phigs.h as:

```
typedef union {
        struct Pgse_r1 {
                Pint                unused;
        } gse_r1;
        Pdata                       unsupp;     /∗ unsupp GSE data record ∗/
                                                /∗ implementation dependent ∗/
        struct Pgse_u1 {
                Pint                highl_colr;
        } gse_u1;
        struct Pgse_u4 {
                Pfloat              slant_angle;
        } gse_u4;
        struct Pgse_u5 {
                Pfloat              slant_angle;
        } gse_u5;
        struct Pgse_u6 {
                Phigs_pgon_type     pgon_type;
        } gse_u6;
        struct Pgse_u7 {
                Pattrs              primitive;  /∗ attribute set whose AA ID
                                                    to set∗/
                Phigs_aalias_id     aa_id;      /∗ anti-aliasing identifier ∗/
        } gse_u7;
        struct Pgse_u8 {
                Pfloat              scale;      /∗ text stroke width scale factor ∗/
        } gse_u8;
        struct Pgse_u9 {
                Pattrs              primitive;  /∗ attribute set’s line end-cap style
                                                   to set ∗/
                Phigs_end_cap       style;      /∗ style for wide-stroke
                                                   end-caps ∗/
        } gse_u9;
        struct Pgse_u10 {
                Pattrs              primitive;  /∗ attribute’s wide-stroke join style
                                                   to set ∗/
                Phigs_join_rec      join;       /∗ data record for wide-stroke
```

```
                                                    joins */
          } gse_u10;
     } Pgse_data;
```

FORTRAN Input Parameters

GSEID The identifier of the generalized structure element to insert. Recognized identifiers are defined in phigs77.h as:

–1	PUGSEHLCI	*Set highlight colour index*
–4	PUGSESETTSA	*Set text slant angle*
–5	PUGSESETATSA	*Set annotation text slant angle*
–6	PUGSEPGONTYPE	*Set polygon type*
–7	PUGSESETSAAID	*Set stroke anti-aliasing flag*
–8	PUGSESETTXSWSF	*Set text stroke width scale factor*
–9	PUGSESETSTEC	*Set stroke end cap*
–10	PUGSESETSTJ	*Set stroke join*
–11	PUGSESILEDGEF	*Set silhouette edge flag*

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A GSE data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted in the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

PACK DATA RECORD (3P)
GENERALIZED STRUCTURE ELEMENT –1 (3P)
GENERALIZED STRUCTURE ELEMENT –4 (3P)
GENERALIZED STRUCTURE ELEMENT –5 (3P)
GENERALIZED STRUCTURE ELEMENT –6 (3P)
GENERALIZED STRUCTURE ELEMENT –7 (3P)
GENERALIZED STRUCTURE ELEMENT –8 (3P)
GENERALIZED STRUCTURE ELEMENT –9 (3P)
GENERALIZED STRUCTURE ELEMENT –10 (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -1 – create structure element to set highlight colour index

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint            id;        gse identifier
Pgse_data       *gse;      gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER          GSEID          GSE identifier
INTEGER          LDR            dimension of data record array
CHARACTER*80     DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -1 creates a structure element that can be used to set the highlight colour index.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_HIGHLIGHT_COLOUR_INDEX, which is defined in phigs.h to be -1.

gse A pointer to a Pgse_data union, Pgse_u1, containing the information needed to perform the function specified by *id*. Pgse_u1 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u1 {
                Pint    highl_colr;
        } gse_u1;
} Pgse_data;
```

The *highl_colr* component specifies the new highlighting colour index to use when highlighting output primitives.

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSEHLCI, which is defined in phigs77.h to be -1.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSEHLCI should be as follows:

IL	The number of integers = 1.
IA	Contains one integer value, specifying the highlight colour index.
RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

The *Set Highlight Colour Index* GSE is used to set a *highlighting colour index* entry in the PHIGS traversal state list. This entry will be used when highlighting subsequent output primitives, overriding the colour index from other attributes. If the specified colour index is negative, an error is generated and the element is not created.

The default highlighting colour index is the maximum colour index value on the workstation. The default highlighting on a monochrome workstation yields no visual distinction for primitives with colour index 1 (the default).

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

PACK DATA RECORD (3P)
SET HIGHLIGHTING FILTER (3P)
GENERALIZED STRUCTURE ELEMENT (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -4 – create structure element to set slant angle for text

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint         id;       gse identifier
Pgse_data    *gse;     gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER           GSEID           GSE identifier
INTEGER           LDR             dimension of data record array
CHARACTER*80      DATREC(LDR)     data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -4 creates a structure element that can be used to set the slant angle for text.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_TEXT_SLANT_ANGLE, which is defined in phigs.h to be -4.

gse A pointer to a Pgse_data union, Pgse_u4, containing the information needed to perform the function specified by *id*. Pgse_u4 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u4 {
                Pfloat    slant_angle;    /* slant angle (in radians) of text */
        } gse_u4;
} Pgse_data;
```

The *slant_angle* field of this record specifies the angle of inclination, in radians, of the text characters from the character up vector. The default value for normal text is 0 radians. The *slant_angle* can take values between -π/2 and +π/2.

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETTSA, which is defined in phigs77.h to be -4.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETTSA should be as follows:

IL	The number of integers = 0.
RL	The number of real values = 1.
IA	Contains one real value. It specifies the slant angle for text. The default value for normal text is 0 radians.
SL	The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

This GSE is used to set the *text slant angle* entry in the PHIGS traversal state list. This entry will alter the appearance of subsequent text primitives. The vertical component of the text character is slanted from the character up vector by the specified angle. The height of the characters is constant for varying slant angles. (This can lead to elongated characters for very large slant angles.)

The slant angle, in radians, can take values between $-\pi/2$ and $+\pi/2$. A slant angle between $-\pi/2$ and 0 will slant the characters in the backward direction. A slant angle between 0 and $\pi/2$ will slant the characters in the forward direction. The default value for the slant angle is 0 radians, which results in upright text. This GSE does not affect TEXT ALIGNMENT and cannot be used with INQUIRE TEXT EXTENT.

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

PACK DATA RECORD (3P)
GENERALIZED STRUCTURE ELEMENT (3P)
TEXT (3P)
TEXT 3 (3P)
SET CHARACTER UP VECTOR (3P)
SET TEXT PATH (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -5 – create structure element to set slant angle for annotation text

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint          id;       gse identifier
Pgse_data     *gse;     gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER          GSEID          GSE identifier
INTEGER          LDR            dimension of data record array
CHARACTER*80     DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -5 creates a structure element that can be used to set the slant angle for annotation text.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_ATEXT_SLANT_ANGLE, which is defined in phigs.h to be -5.

gse A pointer to a Pgse_data union, Pgse_u5, containing the information needed to perform the function specified by *id*. Pgse_u5 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u5 {
                Pfloat    slant_angle;    /* slant angle (in radians) of text */
        } gse_u5;
} Pgse_data;
```

The *slant_angle* field of this record specifies the angle of inclination, in radians, of the annotation text characters from the character up vector. The default value for normal annotation text is 0 radians. The *slant_angle* can take values between -π/2 and +π/2.

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETATSA, which is defined in phigs77.h to be -5.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the

information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETATSA should be as follows:

IL	The number of integers = 0.
RL	The number of real values = 1.
IA	Contains one real value. It specifies the slant angle for annotation text. The default value for normal text is 0 radians.
SL	The number of strings = 0.

Execution If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

This GSE is used to set the *annotation text slant angle* entry in the PHIGS traversal state list. This entry will alter the appearance of subsequent annotation text primitives. The vertical component of the text character is slanted from the character up vector by the specified angle. The height of the characters is constant for varying slant angles. (This can lead to elongated characters for very large slant angles.)

The slant angle, in radians, can take values between -π/2 and +π/2. A slant angle between -π/2 and 0 will slant the characters in the backward direction. A slant angle between 0 and π/2 will slant the characters in the forward direction. The default value for the slant angle is 0 radians, which results in upright text. This GSE does not affect ANNOTATION TEXT ALIGNMENT and cannot be used with INQUIRE TEXT EXTENT.

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

PACK DATA RECORD (3P)
GENERALIZED STRUCTURE ELEMENT (3P)
ANNOTATION TEXT RELATIVE (3P)
ANNOTATION TEXT RELATIVE 3 (3P)
SET ANNOTATION TEXT CHARACTER UP VECTOR (3P)
SET ANNOTATION TEXT PATH (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -6 – create structure element to set current solid polygon type

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint          id;       gse identifier
Pgse_data     *gse;     gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER          GSEID          GSE identifier
INTEGER          LDR            dimension of data record array
CHARACTER*80     DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -6 creates a structure element that can be used to set the current solid polygon type.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_PGON_TYPE, which is defined in phigs.h to be -6.

gse A pointer to a Pgse_data union, Pgse_u6, containing the information needed to perform the function specified by *id*. Pgse_u6 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u6 {
                Phigs_pgon_type     pgon_type;     /* polygon type */
        } gse_u6;
} Pgse_data;
```

The *pgon_type* of this record specifies the type of the current polygon. It is defined in phigs.h as:

```
typedef enum {
        PHIGS_PGON_TYPE_COMPLEX = 0,
        PHIGS_PGON_TYPE_NSI
} Phigs_pgon_type;
```

The PHIGS_PGON_TYPE_COMPLEX enumeration value specifies that the current solid polygon may be a self-intersecting polygon. The PHIGS_PGON_TYPE_NSI enumeration value specifies that the current solid polygon is a nonself-intersecting polygon.

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSEPGONTYPE, which is defined in phigs77.h to be -6.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSEGONTYPE should be as follows:

IL The number of integers = 1.

IA Contains one integer value specifying the type of the current polygon. Valid values are defined in phigs77.h as follows:

0	PPGONTYPECOMPLEX
1	PPGONTYPENSI

The PPGONTYPECOMPLEX specifies that the current solid polygon can be self-intersecting. PPGONTYPENSI specifies that the current solid polygon is nonself-intersecting.

RL The number of real values = 0.

SL The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

The Set Polygon Type GSE is used to set the *current polygon type* entry in the PHIGS traversal state list. This is used for polygon decomposition performance. This should have no effect on devices that do not take advantage of the triangulated geometries.

The default polygon type is PPGONTYPECOMPLEX.

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED STRUCTURE ELEMENT (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -7 – create structure element to control width-1 stroke anti-aliasing

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint        id;      gse identifier
Pgse_data   *gse;    gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER        GSEID          GSE identifier
INTEGER        LDR            dimension of data record array
CHARACTER*80   DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -7 creates a structure element that can be used to set or reset a primitive’s anti-aliasing identifier during traversal. The anti-aliasing identifier, together with the workstation anti-aliasing mode, controls stroke anti-aliasing. It is used on some devices for polylines, markers, text, HOLLOW interior fill, or fill-area-set edges, to improve stroke (vector) appearance.

On a GT true colour workstation, stroke anti-aliasing improves the appearance of a one-pixel wide stroke (vector) by distributing it over three pixels. On a GT workstation, the rendering speed with anti-aliasing is about 40% slower. Other devices may ignore this GSE during traversal.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_SET_STROKE_AA_ID, which is defined as -7 in phigs.h.

gse A pointer to a Pgse_data union, Pgse_u7, containing the information needed to perform the function specified by *id*. Pgse_u7 is defined in phigs.h as:

```
typedef union {
    struct Pgse_u7 {
        Pattrs           primitive;   /* primitive whose AA flag
                                         is to be set*/
        Phigs_aalias_id  aa_id;       /* anti-aliasing identifier */
    } gse_u7;
} Pgse_data;
```

primitive The primitive whose anti-aliasing flag is being set. It is defined by the

Pattrs enumeration:

```
typedef enum {
        PATTR_LINE,
        PATTR_MARKER,
        PATTR_TEXT,
        PATTR_INT,
        PATTR_EDGE
} Pattrs;
```

Note that PATTR_INT refers to HOLLOW interior style.

aa_id Specifies whether the anti-aliasing identifier is set as defined in the Phigs_aalias_id enumeration:

```
typedef enum {
        PHIGS_AA_ID_NONE,              /* do not anti-alias (the
                                          default) */
        PHIGS_AA_ID_TO_CONST_BG,       /* constant background
                                          colour 0 */
        PHIGS_AA_ID_TO_ARBIT_BG        /* arbitrary background
                                          (each pixel) */
} Phigs_aalias_id;
```

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETAAID, which is defined as –7 in phigs77.h.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETAAID should be as follows:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is an enumerated value specifying the primitive whose anti-aliasing flag is being set:

0	PPOLYLINE
1	PPOLYMARKER
2	PTEXT
3	PINTERIOR
4	PEDGE

Note that PINTERIOR refers to HOLLOW interior style.

Element 2 is an enumerated value specifying the anti-aliasing identifier. Valid values are defined by phigs77.h.

0	PHIGSAAIDNONE
1	PHIGSAAIDCONST
2	PHIGSAAIDARBIT

RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

The *Set Stroke Anti-aliasing Identifier* GSE is used to set the *stroke anti-aliasing identifier* entry in the SunPHIGS traversal state list for the attribute set specified. This entry is used to control anti-aliasing of those width-1 strokes when the workstation anti-aliasing mode is Stroke. See ESCAPE -40.

When the anti-aliasing identifier is None, no anti-aliasing is performed. This results in maximum performance.

Anti-aliasing to a Constant Background Colour assumes that the strokes are on top of the workstation's background colour (workstation colour table entry 0), or on top of other strokes of the same colour. Anti-aliasing strokes on top of non-background colours may result in the display of other colours; strokes of the background colour are not visible when anti-aliased. RGB Colours and intensities are added when strokes overlap; where red and green strokes overlap, the result is yellow.

Anti-aliasing to Arbitrary Backgrounds blends the strokes' colours into whatever colour is found at the time the stroke is written, introducing order-dependencies. This is appropriate for edges applied to filled areas of a different colour. Use ESCAPE -40 to set the workstation anti-aliasing mode to Stroke with 2-pass traversal to reduce order-dependent artifacts.

The default for all attributes' anti-aliasing ID is None.

ERRORS

005 Ignoring function, function requires state (PHOP, ∗, STOP, ∗)

SEE ALSO

GENERALIZED STRUCTURE ELEMENT (3P)
PACK DATA RECORD (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -8 – create structure element to control width of vector strokes used to draw vector text.

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint          id;       gse identifier
Pgse_data     *gse;     gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER          GSEID           GSE identifier
INTEGER          LDR             dimension of data record array
CHARACTER*80     DATREC(LDR)     data record
```

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -8 creates a structure element that sets the width of the vector strokes used to draw vector text. This attribute also applies to annotation text. On a GT workstation, the rendering speed with the text stroke width set wider than 1-pixel is about one-half as fast.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element, which is defined as –8 in phigs.h, is PUGSE_SET_TEXT_STROKE_WIDTH_SCALE_FACTOR.

gse A pointer to a Pgse_data union, Pgse_u8, containing the information needed to perform the function specified by *id*. Pgse_u8 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u8 {
                Pfloat     scale;      /* text stroke width scale factor */
        } gse_u8;
} Pgse_data;
```

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETTXSWSF, which is defined as –8 in phigs77.h.X

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETTXSWSF should be as follows:

IL	The number of integers = 0.
RL	The number of real values = 1.
RA	Contains one real value. It specifies the text stroke width scale factor.
SL	The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

The *Text Stroke Width Value* GSE is used to set the *Text Stroke Width Scale Factor* entry in the SunPHIGS traversal state list. This entry is used to render vector text with wider stroke widths.

The default Text Stroke Width Scale Factor is 1.0.

The scale factor is multiplied by the normal text stroke width to give the effective stroke width. The SunPHIGS nominal stroke width is 1.0, which is also the default and the minimum stroke width for this function. The maximum stroke width is 10.0. A floating-point value less than 1.0 sets the width to 1.0. Values greater than 1.0 are truncated to the integer number of pixels.

ERRORS

005　Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED STRUCTURE ELEMENT (3P)
GENERALIZED STRUCTURE ELEMENT -9 (3P)
GENERALIZED STRUCTURE ELEMENT -10 (3P)
PACK DATA RECORD (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -9 – create structure element to control appearance of wide stroke end-caps.

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint          id;      gse identifier
Pgse_data     *gse;    gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER          GSEID           GSE identifier
INTEGER          LDR             dimension of data record array
CHARACTER*80     DATREC(LDR)     data record
```

Required PHIGS Operating States

(PHOP, ∗, STOP, ∗)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -9 creates a structure element that may be used to set the type of end-cap for wide strokes.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element, which is defined as -9 in phigs.h, is PUGSE_SET_STROKE_END_CAP.

gse A pointer to a Pgse_data union, Pgse_u9, containing the information needed to perform the function specified by *id*. Pgse_u9 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u9 {
                Pattrs            primitive;    /* attribute sets line end-cap
                                                     style to set */
                Phigs_end_cap     style;        /* style for wide-stroke
                                                     end-caps */
        } gse_u9;
} Pgse_data;
```

primitive May be PATTR_LINE, PATTR_MARKER, PATTR_TEXT, PATTR_INT, or PATTR_EDGE as defined by the Pattrs enumeration.

Note that PATTR_INTERIOR refers to HOLLOW interior style.

Note that polymarker, text, and HOLLOW interior style strokes are always width 1.

style Specifies the end-cap style as defined by the Phigs_end_cap enumeration:

```
typedef enum {
        PHIGS_END_BUTT,
        PHIGS_END_SQUARE,
        PHIGS_END_ROUND
} Phigs_end_cap;
```

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETSTEC, which is defined as -9 in phigs77.h.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETSTEC should be as follows:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is an enumerated value specifying the primitive to which wide vector end-caps are being applied:

0	PPOLYLINE
1	PPOLYMARKER
2	PTEXT
3	PINTERIOR
4	PEDGE

Note that PINTERIOR refers to HOLLOW interior style.

Element 2 is an enumerated value specifying the end-cap style:

0	PHIGSENDBUTT
1	PHIGSENDSQUARE
2	PHIGSENDROUND

RL The number of real values = 0.

SL The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

This function is used to set the *Stroke End Cap Style* entry in the PHIGS traversal state list. This entry is used to render end-caps in the specified style (butt, square, or rounded).

The end-cap choices include rounded, squared, and butt, which is the default for SunPHIGS workstations.

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

ESCAPE -42 (3P)
GENERALIZED STRUCTURE ELEMENT (3P)
GENERALIZED STRUCTURE ELEMENT -8 (3P)
GENERALIZED STRUCTURE ELEMENT -10 (3P)
PACK DATA RECORD (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -10 – create structure element to control appearance of wide-stroke joints.

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint          id;       gse identifier
Pgse_data     *gse;     gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER         GSEID          GSE identifier
INTEGER         LDR            dimension of data record array
CHARACTER*80    DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -10 creates a structure element that can be used to set the type of joint for wide strokes.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element, which is defined as -10 in phigs.h, is PUGSE_SET_STROKE_JOIN.

gse A pointer to a Pgse_data union, Pgse_u10, containing the information needed to perform the function specified by *id*. Pgse_u10 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u10 {
                Pattrs            primitive;    /* attribute's wide-stroke
                                                   join style to set */
                Phigs_join_rec    join;         /* data record for
                                                   wide-stroke joins */
        } gse_u10;
} Pgse_data;
```

primitive May be PATTR_LINE, PATTR_MARKER, PATTR_TEXT, PATTR_INT, or PATTR_EDGE as defined by the Pattrs enumeration.

```
typedef enum {
    PPOLYLINE,
    PPOLYMARKER,
    PTEXT,
    PINTERIOR,
    PEDGE
} Pattrs;
```

Note that PATTR_INT refers to HOLLOW interior style.

Note that polymarker, text, and HOLLOW interior style strokes are always width 1.

join Specifies the style of joint as defined by the Phigs_join_rec structure:

```
typedef struct {
        Phigs_join_style    style;        /* style for wide-line
                                             joins */
        Pfloat              threshold;
} Phigs_join_rec;
```

The *style* component is defined by the Phigs_join_style enumeration:

```
typedef enum {
        PHIGS_JOIN_BUTT,
        PHIGS_JOIN_BEVELED,
        PHIGS_JOIN_MITRED,
        PHIGS_JOIN_ROUND
} Phigs_join_style;
```

The threshold component causes line segments that are almost parallel to fall back from mitred join to beveled join. Sufficiently small angles would result in a very long miter, and so a threshold allows a fall-back to beveled instead. The threshold is the cosine of the minimum angle at which mitering should occur. Larger cosines (smaller angles) result in a beveled line join instead of a mitered line join. The angle at which two lines meet is considered to be between 0 and 180 degrees, and so the cosine threshold must be between +1 and -1, respectively.

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESETSTJ, which is defined as -10 in phigs77.h.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC
A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESETSTJ should be as follows:

IL The number of integers = 2.

IA Contains two integer values.

Element 1 is an enumerated value specifying the primitive that wide vector end-caps are being applied to:

0 PPOLYLINE

1	PPOLYMARKER
2	PTEXT
3	PINTERIOR
4	PEDGE

Note that PINTERIOR refers to HOLLOW interior style.

Element 2 is an enumerated value specifying the join style:

0	PHIGSJOINBUTT
1	PHIGSJOINBEVELED
2	PHIGSJOINMITRED
3	PHIGSJOINROUND

RL The number of real values = 1.

The real element is the threshold. The threshold component causes line segments that are almost parallel to fall back from mitred join to beveled join. Sufficiently small angles would result in a very long miter, and so a threshold allows a fall-back to beveled instead. The threshold is the cosine of the minimum angle at which mitering should occur. Larger cosines (smaller angles) result in a beveled line join instead of a mitered line join. The angle at which two lines meet is considered to be between 0 and 180 degrees, and so the cosine threshold must be between +1 and -1, respectively.

RL The number of real values = 0.

SL The number of strings = 0.

Execution If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

This function is used to set the *Stroke Join Style* entry information in the PHIGS traversal state list. This information is used to render joins in the specified style (butted, beveled, mitred, or rounded).

The joint choices include beveled, mitred, rounded, and butt, which is the default for SunPHIGS workstations.

Certain characters (for example, C) apply the join style. However, other characters (L, V, and M in the default character set and font) are defined with independent vectors instead of polylines. These characters do not have a stroke join to which the join style can be applied.

ERRORS 005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

ESCAPE -42 (3P)
GENERALIZED STRUCTURE ELEMENT (3P)
GENERALIZED STRUCTURE ELEMENT -8 (3P)
GENERALIZED STRUCTURE ELEMENT -9 (3P)
PACK DATA RECORD (3P)

NAME

GENERALIZED STRUCTURE ELEMENT -11 – create a structure element to set the current silhouette edge flag

SYNOPSIS

C Syntax

```
void
pgse ( id, gse )
Pint        id;     gse identifier
Pgse_data   *gse;   gse data record
```

FORTRAN Syntax

```
SUBROUTINE pgse ( GSEID, LDR, DATREC )
INTEGER         GSEID          GSE identifier
INTEGER         LDR            dimension of data record array
CHARACTER*80    DATREC(LDR)    data record
```

Required PHIGS Operating States

(PHOP, *, STOP, *)

DESCRIPTION

Purpose

GENERALIZED STRUCTURE ELEMENT -11 creates a structure element containing a value for the silhouette edge flag.

Note: Support for this function is implementation-dependent. It may not be supported when a workstation is running in remote PEX mode.

C Input Parameters

id The function identifier for this generalized structure element is PUGSE_SIL_EDGE_FLAG, which is defined in phigs.h to be -11.

gse A pointer to a Pgse_data union, Pgse_u11, containing the information needed to perform the function specified by *id*. Pgse_u11 is defined in phigs.h as:

```
typedef union {
        struct Pgse_u11 {
                Phigs_sil_edgef    sil_edgef;    /* silhouette edge flag */
        } Pgse_u11;
} Pgse_data;
```

The *sil_edgef* of this record specifies the value for the current silhouette edge flag. It is defined in phigs.h as:

```
typedef enum {
        PHIGS_SIL_EDGEF_OFF = 0,     No silhouette edges
        PHIGS_SIL_EDGEF_ON = 1,      Enable silhouette edges
} Phigs_sil_edgef;
```

FORTRAN Input Parameters

GSEID The function identifier for this generalized structure element is PUGSESILEDGEF, which is defined in phigs77.h to be -11.

LDR The dimension of DATREC (returned by the PACK DATA RECORD subroutine).

DATREC

A data record packed by the PACK DATA RECORD subroutine containing the information needed to perform the function specified by GSEID.

The arguments passed to PACK DATA RECORD when GSEID = PUGSESILEDGEF should be as follows:

IL	The number of integers = 1.
RL	The number of real values = 0.
IA	Contains one integer value specifying the silhouette edge flag. Valid values are defined in phigs77.h as follows:

0	PHIGSSILEDGEFOFF	*No silhouette edges*
1	PHIGSSILEDGEFON	*Enable silhouette edges*

RL	The number of real values = 0.
SL	The number of strings = 0.

Execution

If the current edit mode is INSERT, then GENERALIZED STRUCTURE ELEMENT is inserted into the currently open structure after the element currently pointed to by the element pointer. If the edit mode is REPLACE, then GENERALIZED STRUCTURE ELEMENT replaces the element pointed to by the element pointer. In either case, the element pointer is updated to point to the new element.

The *Set Silhouette Edge Flag* GSE is used to set the *Silhouette Edge Flag* entry in the PHIGS traversal state list.

During traversal, when the workstation's current *Silhouette Edge Mode* entry is on and the *Silhouette Edge Flag* in the traversal state list is on, SunPHIGS will attempt to render the silhouette edges, if any, implicit in area-filling primitives. See ESCAPE -34 (3P).

ERRORS

005 Ignoring function, function requires state (PHOP, *, STOP, *)

SEE ALSO

GENERALIZED STRUCTURE ELEMENT (3P)
ESCAPE -34 (3P)
ESCAPE -35 (3P)

NAME

WORKSTATION TYPE CREATE – create new workstation type — SunPHIGS extension

SYNOPSIS

C Syntax

Pint
phigs_ws_type_create (base, attributes)

```
Pint                  base;         base workstation type
Phigs_ws_type_attr    attributes;
```

FORTRAN Syntax

SUBROUTINE phigswstcreate (BASEWST, NEWWST)

```
INTEGER    BASEWST    base workstation type
INTEGER    NEWWST     new workstation type
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

WORKSTATION TYPE CREATE creates a new workstation type. The new type is a copy of an existing type and has the same characteristics. WORKSTATION TYPE SET can operate on the new type to modify its workstation description table. For a description of the predefined workstation types and their characteristics, see OPEN WORKSTATION. The workstation description table associated with the new type is a copy of the one associated with the base type, passed as the first argument to this function. (This is true unless the table is changed by attributes specified in the C version of the call).

An application creates a workstation type to obtain one having workstation description table values different from those of an existing workstation type. WORKSTATION TYPE SET can change the workstation description table values.

The workstation type returned from this function can be used anywhere a workstation type is required; it is added to the list of available workstation types in the PHIGS description table.

This is a SunPHIGS addition and is not part of the PHIGS standard.

C Input Parameters

base An existing workstation type from which to build the new type.

attributes
A 0-terminated list of workstation type attributes and corresponding values. As described in WORKSTATION TYPE SET, the workstation description table fields of the new workstation type are modified according to the contents of this list.

C Output Parameters

If this function is successful, then new workstation type is returned as the value of the function.

FORTRAN Parameters

BASEWST
An existing workstation type from which to build the new type.

NEWWST
The new workstation type. WORKSTATION TYPE SET is used to change the workstation description table fields of this new type.

Execution

The third argument of OPEN WORKSTATION is a *workstation type*. Each workstation type has a *workstation description table* associated with it, which contains both the PHIGS-specified data describing workstation capabilities and some SunPHIGS-specific data describing workstation characteristics. An application can change SunPHIGS-specific data and some PHIGS data prior to opening a workstation. WORKSTATION TYPE SET describes the data that can be changed and how to do it.

SunPHIGS supplies a number of predefined workstation types. They are described in OPEN WORKSTATION. The WORKSTATION TYPE SET function cannot operate on predefined types; that is, their description table values cannot be changed. Instead, it is necessary to create a new workstation type based on a predefined type, which WORKSTATION TYPE SET can modify. Once created, the new workstation type can be used anywhere that PHIGS requires a workstation type, including the *base type* argument of this function. WORKSTATION TYPE SET can repeatedly edit the new type, and it can be passed to OPEN WORKSTATION to use to open workstations with only slightly different description table values. When the type is no longer needed, it should be destroyed by using WORKSTATION TYPE DESTROY.

If a call to this function is successful, then the new workstation type is added to the *list of available workstation types* in the PHIGS description table.

ERRORS

002 Ignoring function, function requires state (PHOP, *, *, *)

052 Ignoring function, workstation type not recognized by the implementation

SEE ALSO

WORKSTATION TYPE DESTROY (3P)
WORKSTATION TYPE GET (3P)
WORKSTATION TYPE SET (3P)
OPEN WORKSTATION (3P)
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES (3P)
INQUIRE WORKSTATION CONNECTION AND TYPE (3P)
PHIGS WORKSTATION DESCRIPTION TABLE (7P)

NAME

WORKSTATION TYPE DESTROY – destroy workstation type – SunPHIGS extension

SYNOPSIS

C Syntax

```
void
phigs_ws_type_destroy ( wst )
Pint      wst;      workstation type
```

FORTRAN Syntax

```
SUBROUTINE phigswstdestroy ( WST )
INTEGER     WST     workstation type
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

Destroy a workstation type created by the application. WORKSTATION TYPE DESTROY deallocates any memory used by the workstation type and removes it from the *list of available workstation types* in the PHIGS description table. After a workstation type is destroyed it can no longer be used. For a description of how and why to create a new workstation type, see WORKSTATION TYPE CREATE .

This is a SunPHIGS addition and is not part of the PHIGS standard.

C Input Parameter

wst The workstation type to destroy.

FORTRAN Input Parameter

WST The workstation type to destroy.

ERRORS

002	Ignoring function, function requires state (PHOP, *, *, *)
052	Ignoring function, workstation type not recognized by the implementation
-100	Ignoring function, workstation type is a default type or bound to a workstation and cannot be modified

SEE ALSO

WORKSTATION TYPE CREATE (3P)
WORKSTATION TYPE GET (3P)
WORKSTATION TYPE SET (3P)
OPEN WORKSTATION (3P)
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES (3P)

NAME

WORKSTATION TYPE GET – return value of specified workstation type field – SunPHIGS extension

SYNOPSIS

C Syntax

caddr_t
phigs_ws_type_get (wst, attr, arg)

```
Pint    wst;     workstation type
char    *attr;
char    *arg
```

FORTRAN Syntax

SUBROUTINE phigswstget (WST, ATTRIBUTE, VALUE1, VALUE2, VALUE3)

```
INTEGER    WST           workstation type
INTEGER    ATTRIBUTE     attribute
INTEGER    VALUE1        value
INTEGER    VALUE2        value
INTEGER    VALUE3        value
```

Required PHIGS Operating States

(PHOP, *, *, *)

DESCRIPTION

Purpose

WORKSTATION TYPE GET retrieves the values of certain fields in the workstation description table associated with a specified workstation type. The *attribute* parameter specifies the value to return. For more information, see WORKSTATION TYPE CREATE and WORKSTATION TYPE SET.

This is a SunPHIGS addition and is not part of the PHIGS standard.

C Input Parameters

wst The workstation type in question.

attr The attribute indicating the value to return. For a list of attributes defined, their descriptions, and the data types of their values, see WORKSTATION TYPE SET.

arg Provided for some attributes to specify additional information. If no additional information is needed, then NULL must be specified for this parameter.

C Output Parameters

This function returns the requested information as the value of the function. The return value must be cast to the appropriate type when accessing the returned information.

FORTRAN Input Parameters

WST The workstation type in question.

ATTRIBUTE
The attribute indicating the value(s) to return.

FORTRAN Output Parameters

VALUE1, VALUE2, VALUE3
The value, or values, of the attribute. This must be of the appropriate type for the return value (that is, INTEGER, REAL, LOGICAL, or CHARACTER(*)), depending on the

attribute. For a list of attributes defined, their description, and the data types of their values, see WORKSTATION TYPE SET.

Most attributes return one value, but some attributes return two or three values. The following attributes return two values. (The first is an INTEGER value specifying the string length; the second is CHARACTER(*), specifying the string itself.)

PHIGSTOOLLABEL
PHIGSTOOLICONFL
PHIGSTOOLICONLBL

The following attributes return three REAL values, specifying the red, blue, and green colour components.

PHIGSNMLCURSCLR
PHIGSTRVCURSCLR

ERRORS

002 Ignoring function, function requires state (PHOP, *, *, *)

052 Ignoring function, workstation type not recognized by the implementation

SEE ALSO

WORKSTATION TYPE SET (3P)
PHIGS WORKSTATION DESCRIPTION TABLE (7P)
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES (3P)
INQUIRE WORKSTATION CONNECTION AND TYPE (3P)

NAME

WORKSTATION TYPE SET – set workstation type attributes – SunPHIGS Extension

SYNOPSIS

C Syntax

```
caddr_t
phigs_ws_type_set ( wst, <attribute-value list> )
Pint                wst;           workstation type
Phigs_ws_type_attr  attributes;    attributes
```

FORTRAN Syntax

```
SUBROUTINE phigswstset ( WST, ATTRIBUTE, VALUE1, VALUE2, VALUE3 )
INTEGER   WST          workstation type
INTEGER   ATTRIBUTE    attribute
INTEGER   VALUE1       value
INTEGER   VALUE2       value
INTEGER   VALUE3       value
```

Required PHIGS Operating States

(PHOP, ∗, ∗, ∗)

DESCRIPTION

Purpose

WORKSTATION TYPE SET changes the values of fields in the workstation description table associated with a specified workstation type. These values control the characteristics of workstations of this type when they are opened. WORKSTATION TYPE SET operates only on workstation types created by WORKSTATION TYPE CREATE. The predefined types cannot be modified, nor can types returned by INQUIRE WORKSTATION CONNECTION AND TYPE. See OPEN WORKSTATION for a description of the predefined workstation types.

The workstation description table contains the PHIGS-defined fields and SunPHIGS-specific fields. All of the SunPHIGS-specific fields, as well as some of the PHIGS-specific fields, can be changed with this function. The fields that can be modified are listed below in the *Workstation Attributes* section.

This is a SunPHIGS addition, and is not part of the PHIGS standard.

C Input Parameters

wst The workstation type to modify. This cannot be a specific workstation type (one associated with an open workstation) or one of the predefined types.

<attributes>
A zero terminated list of workstation type attributes and corresponding values. The attributes are defined in phigs.h. The data type of the values varies according to the attribute. The list of attributes below specifies the value type required for each attribute. The attribute fields of the workstation type are modified according to the contents of this list.

The following is an example of an attribute list:

```
Pint  wst;

phigs_ws_type_set( wst,
                    PHIGS_TOOL_X, (char*)700,
                    PHIGS_TOOL_Y, (char*)100,
                    PHIGS_TOOL_WIDTH, (char*)200,
                    PHIGS_TOOL_HEIGHT, (char*)200,
                    PHIGS_TOOL_LABEL, "views",
                    (char*)0 );
```

FORTRAN Input Parameters

WST The workstation type to operate on. This cannot be a bound type or one of the predefined types.

ATTRIBUTE
The workstation description table field to set. The attributes are defined in PARAMETER statements in phigs77.h. The data type of the values varies according to the attribute. The list of attributes below specifies the value type required for each attribute.

VALUE1, VALUE2, VALUE3
The value, or values, of the attribute. The latter two are not used for most attributes. The attribute descriptions below specify which are used for a given attribute.

Execution

WORKSTATION TYPE SET modifies the fields of the workstation description table associated with the specified workstation type. The modifiable fields are listed below. The changes are specified with *attribute-value* pairs. The *attribute* specifies the field to change. The *value* is the value to assign to the field. The SunPHIGS-specific fields are different for different workstation types, so the attributes are listed according to the workstation type to which they apply.

Workstation Type Attributes

The workstation description table attributes are described here. Both the C and FORTRAN constants are listed along with a description of each attribute, the data type required and the default value. The C constants are defined in phigs.h. The FORTRAN constants are defined with PARAMETER statements in phigs77.h.

Common Attributes

PHIGS_BASE_NAME (C), *PHIGSBASENAME* (FORTRAN)
This argument to WORKSTATION TYPE GET retrieves the base type of the workstation type. It can not be specified as an argument to this function.

C enumeration: *PHIGS_X_TOOL, PHIGS_X_DRAWABLE, PHIGS_X_DRAWABLE_REGION, PHIGS_CGM_OUT*

FORTRAN enumeration: *PHIGSXTOOL, PHIGSXDRAWABLE, PHIGSXDRAWREG, PHIGSCGMOUT*

Note: PHIGS_X_DRAWABLE_REGION may not be supported when a workstation is running in remote PEX mode.

PHIGS_COLOUR_TABLE_SIZE (C), *PHIGSCLRTBLSIZE* (FORTRAN)
The size of the PHIGS colour table. If the size is not a power of two, then it will be rounded up to the nearest power of two. Maximum size is 256 on Indexed Color

workstations, and 4,096 on True Color workstations (see PHIGS_COLOUR_MODE, below).

Data type: *Pint* (C), *INTEGER* (FORTRAN)
Default: 32

PHIGS_WST_NUM_VIEW_REP (C), *PHIGSWSTNUMVWREP* (FORTRAN)
Sets the number of view representations available for the workstation type. Minimum is 6.

Data type: *Pint* (C), *INTEGER* (FORTRAN)
Default: 20

Common X Tool and X Drawable Attributes

PHIGS_ACCELERATOR_USAGE (C), *PHIGSACCUSAGE* (FORTRAN)
Specifies whether or not SunPHIGS should use hardware acceleration directly, if any exists. *None* disables direct use of any existing accelerator. *Required* specifies that if an accelerator is not available, OPEN WORKSTATION should fail and generate an appropriate error. *Preferred* (the default) indicates that an accelerator should be used if it is available, but to perform graphics output without acceleration if not.

Data type: *Phigs_acc_usage* (C), *INTEGER* (FORTRAN)
Default: *Preferred*

C enumeration: *PHIGS_USE_ACC_PREFERRED*, *PHIGS_USE_ACC_REQUIRED*, *PHIGS_USE_ACC_NONE*

FORTRAN enumeration: *PHIGSUSEACCPREF*, *PHIGSUSEACCREQ*, *PHIGSUSEACCNONE*

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

PHIGS_COLOUR_MODE (C), *PHIGSCLRMODE* (FORTRAN)

This attribute allows the application to select a model for the PHIGS colour pipeline and the selection of X visuals.

Selection of PHIGS' colour pipeline:

When the workstation colour mode is set to index colour, direct colour (that is, specifying colours by RGB triplets) is not supported, and direct colours from SunPHIGS Extension attributes and primitive data are drawn as colour type indirect, colour index one. Primitives that use indirect colour (including all PHIGS-standard primitives) will appear exactly as specified. Anti-aliasing and alpha transparency are not supported on index colour workstations.

When this attribute is set to true colour, both indirect and direct colour are supported. This means that colour can be specified either as indices or as RGB colours. All PHIGS-standard colour attributes are supported, and both indirect and direct specification of colour are available using SunPHIGS Extension attributes and primitives. Colormap double buffering is not supported on true colour workstations.

On true colour workstations, changes to the workstation colour table do not appear until the next traversal of the posted structures. The dynamic modification flag for SET COLOUR REPRESENTATION is IRG.

SunPHIGS simulates true colour on PseudoColor visuals if this attribute is set to true colour. This simulation may be slower than index colour mode on PseudoColor visuals, and the colour of primitives may be slightly different. The colour cube used to provide this simulation can be accessed through the PHIGS_X_CMAP_PROP_ATOM workstation type attribute. The application can overwrite the colour cube with a custom colour cube. See Colour(7P) under the subsection Sharing Colormap Resources for more details.

On TrueColor visuals, true colour mode includes all of the functionality of index colour mode; thus index colour mode is not supported on TrueColor visuals.

Selection of X visuals:

X Tool workstations will create a window with the best visual for the workstation colour mode from the list of visuals supported by the display.

On an indexed colour (only) display, the best visual is typically a PseudoColor visual, where pixel colours are specified as indices into a colour table. On a true colour (only) display, the best visual is TrueColor visual, where pixel colours are specified as red, green, and blue (RGB) triplets. Since the GS and GT support both 8-bit indexed and 24-bit true colour display modes, a PseudoColor visual is best if the colour mode is indexed, and a TrueColor visual is best if the colour mode is true colour.

For X Drawable (and X Drawable Region) workstations, the application has the responsibility of creating a window with the appropriate visual; its drawable is provided to SunPHIGS at OPEN WORKSTATION (3P). There is a mismatch between the default X Visual on the GS or GT, PseudoColor, and the default SunPHIGS workstation colour mode, true colour. Programmers must be careful to select a TrueColor visual when creating the drawable for a true colour workstation, or to select a PseudoColor visual for an indexed colour workstation. On a GS or GT, mismatching the workstation colour mode and the drawable's visual results in **unaccelerated** operation, and may raise a warning or error in a future release. The X Drawable example programs demonstrate careful visual selection from among the display's supported visual types.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Phigs_colr_mode* (C), *INTEGER* (FORTRAN)

Default: *True colour*

C enumeration: *PHIGS_INDEX_COLOUR, PHIGS_TRUE_COLOUR*

FORTRAN enumeration: *PHIGSINDEXCOLOUR, PHIGSTRUECOLOUR*

PHIGS_PRUNE_MODE (C), *PHIGSPRUNEMODE* (FORTRAN)

Indicates whether to use output primitive pruning. This checks each primitive's extent against the view volume; if the extent does notintersect the view volume, then the primitive is clipped, and further processing is skipped. Primitive pruning incurs a small overhead during traversal. If any primitives of the posted structures are clipped, then primitive pruning will probably improve performance.

Data type: *Phigs_prune_mode* (C), *INTEGER* (FORTRAN)
Default: *prune primitives*

C enumeration: *PHIGS_PRUNE_NONE, PHIGS_PRUNE_PRIMS*

FORTRAN enumeration: *PHIGSPRUNENONE, PHIGSPRUNEPRIMS*

PHIGS_X_CMAP_PROP_ATOM (C), *PHIGSXCMAPATOM* (FORTRAN)

Allows application to control colormap sharing. See *COLOUR (7P)* for more information.

Data type: *Atom (UNSIGNED LONG)* (C), *None (FORTRAN)*
Default:

PHIGS_DC_MODEL (C), *PHIGSDCMODEL* (FORTRAN)

Set the effect of window resizing on the workstation's DC limits. Valid values are defined in phigs.h.

Fixed, the default, causes the DC limits to be left as they are, even though the new window size is possibly smaller or larger than the limits. *Adjust to window* causes the DC limits to be adjusted to the new window size.

Data type: *Phigs_DC_model* (C), *INTEGER* (FORTRAN)
Default: *Fixed*

C enumeration: *PHIGS_DC_LIMITS_FIXED, PHIGS_DC_LIMITS_ADJUST_TO_WINDOW*

FORTRAN enumeration: *PHIGSDCLIMFIXED, PHIGSDCLIMADJUST*

PHIGS_X_BUF_MODE (C), *PHIGSXBUFMODE* (FORTRAN)

Specifies the method of double buffering to use. *Single* indicates that SunPHIGS should not use double buffering. *Double* indicates that hardware double buffering should be used if available. *Cmap* indicates that colourmap double buffering should be used. See COLOUR (7P) for more information.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Pint* (C), *INTEGER* (FORTRAN)

Default: *Single*

C enumeration: *PHIGS_BUF_SINGLE, PHIGS_BUF_DOUBLE, PHIGS_BUF_DOUBLE_CMAP*

FORTRAN enumeration: *PHIGSBUFSINGLE, PHIGSBUFDOUBLE, PHIGSBUFDBLCMAP*

PHIGS_STEREO (C), *PHIGSSTEREO* (FORTRAN)

Specifies whether left and right eye images can differ. This attribute is ignored if the monitor (the entire screen including all windows on the screen) is not in stereo mode.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Phigs_stereo_mode* (C), *LOGICAL* (FORTRAN)

Default: 0 / false

C enumeration: *FALSE, TRUE*

PHIGS_WS_CATEGORY (C), *PHIGSWSCATEGORY* (FORTRAN)

The category of the workstation. This attribute can be used to change the workstation category from OUTIN (output and input) to OUTPUT (output only).

Data type: *Pws_cat (PCAT_OUT or PCAT_OUTIN only)* (C), *INTEGER* (FORTRAN)

Default: *output-input* for X Tool, *output* for X Drawable, X Drawable Region

C enumeration: *PCAT_OUT, PCAT_IN, PCAT_OUTIN, _PCAT_MO, PCAT_MI*

FORTRAN enumeration: *POUTPT, PINPUT, POUTIN, PMO, PMI*

PHIGS_X_DISPLAY (C), *PHIGSXDISPLAY* (FORTRAN)

Modify the workstation type to reflect the capabilities of the specified display. The display is specified by a display pointer. Some of the workstation type attributes and workstation description table values are dependent on the capabilities of the PEX extension in the X server. This attribute causes such values to be modified to reflect those capabilities. This attribute can only be specified to WORKSTATION TYPE SET or WORKSTATION TYPE CREATE. It cannot be specified to WORKSTATION TYPE GET.

Data type: *Display** (C), *INTEGER* (FORTRAN)

Default: NULL

PHIGS_X_DISPLAY_NAME (C), *PHIGSXDISPNAME* (FORTRAN)

Modify the workstation type to reflect the capabilities of the specified display. The display is specified by a display name string. Some of the workstation type attributes and workstation description table values are dependent on the capabilities of the PEX extension in the X server. This attribute causes such values to be modified to reflect those capabilities.

Data type: *char∗* (C), *CHARACTER(∗)* (FORTRAN)
Default: NULL

PHIGS_X_DISPLAY_WINDOW (C), *PHIGSXDISPWINDOW* (FORTRAN)
Modify the workstation type to reflect the capabilities of the specified window. Some of the workstation type attributes and workstation description table values are dependent on the capabilities of the X window and PEX extension the PHIGS workstation will use. This attribute causes such values to be modified to reflect those capabilities. This attribute can only be specified to WORKSTATION TYPE SET or WORKSTATION TYPE CREATE; it cannot be specified to WORKSTATION TYPE GET.

Data type: *Display∗, Window* (C), *INTEGER, INTEGER* (FORTRAN)
Default: NULL, 0

X Tool Attributes

PHIGS_TOOL_BORDER_WIDTH (C), *PHIGSTOOLBORDER* (FORTRAN)
The border width of the window.

Data type: *Pint* (C), *INTEGER* (FORTRAN)
Default: 0

PHIGS_TOOL_ICON_FILE (C), *PHIGSTOOLICONFL* (FORTRAN)
The file that contains the icon definition to use when the frame is closed.

Data type: *char∗* (C), *CHARACTER(∗)* (FORTRAN)
Default: A path to shared_files / phigs.icon

The default is a full path to the run time shared file, phigs.icon. If this file is not present, the default value will be ‘‘’’ (none).

PHIGS_TOOL_ICON_LABEL (C), *PHIGSTOOLICONLBL* (FORTRAN)
The label to put in the icon when it is displayed.

Data type: *Pchar∗* (C), *CHARACTER(∗)* (FORTRAN)
Default: SunPHIGS

PHIGS_TOOL_LABEL (C), *PHIGSTOOLLABEL* (FORTRAN)
The label to put in the workstation frame banner.

Data type: *Pchar∗* (C), *CHARACTER(∗)* (FORTRAN)
Default: PHIGS Workstation

PHIGS_TOOL_X (C), *PHIGSTOOLX* (FORTRAN)
PHIGS_TOOL_Y (C), *PHIGSTOOLY* (FORTRAN)

The initial position of the workstation in screen coordinates.

Data type: *Pint* (C), *INTEGER* (FORTRAN)

Default: 50, 50

PHIGS_TOOL_WIDTH (C), *PHIGSTOOLWIDTH* (FORTRAN)
PHIGS_TOOL_HEIGHT (C), *PHIGSTOOLHEIGHT* (FORTRAN)

The initial size of the graphics window in screen coordinates.

Data type: *Pint* (C), *INTEGER* (FORTRAN)

Default: 600, 600

PHIGS_NORM_CURSOR_COLOUR (C), *PHIGSNMLCURSCLR* (FORTRAN)
The RGB colour of the normal cursor. The exact appearance of the cursor depends on the colour and the pixels under the cursor.

Data type: *Prgb** (C), *REAL*, *REAL*, *REAL* (FORTRAN)

Default: 1.0, 1.0, 1.0

PHIGS_NORM_CURSOR_TYPE (C), *PHIGSNMLCURSTYPE* (FORTRAN)
The cursor type to use as the *normal* cursor that is displayed when no input devices are active and a traversal is not in progress. See ESCAPE for a description of SunPHIGS cursor usage and control.

Data type: *Phigs_cursor_type* (C), *INTEGER* (FORTRAN)

Default: *Not Sign*

C enumeration: *PHIGS_CURSOR_NONE*, *PHIGS_CURSOR_DEFAULT*, *PHIGS_CURSOR_CROSS*, *PHIGS_CURSOR_NOT_SIGN*

FORTRAN enumeration: *PHIGSCURSNONE*, *PHIGSCURSDEFAULT*, *PHIGSCURSCROSS*, *PHIGSCURSNOTSIGN*

PHIGS_TRAV_CURSOR_COLOUR (C), *PHIGSTRVCURSCLR* (FORTRAN)
The RGB colour of the traversal cursor. The exact appearance of the cursor depends on the colour and the pixels under the cursor.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Prgb** (C), *REAL*, *REAL*, *REAL* (FORTRAN)

Default: 1.0, 1.0, 1.0

PHIGS_TRAV_CURSOR_TYPE (C), *PHIGSTRVCURSTYPE* (FORTRAN)
The cursor type to use as the *traversal* cursor that is displayed when a traversal is in progress. See ESCAPE for a description of cursor usage and control.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Phigs_cursor_type* (C), *INTEGER* (FORTRAN)

Default: *None*

X Drawable Region Attributes

PHIGS_REGION_BACKGROUND (C), *PHIGSRGNBACK* (FORTRAN)
Whether the background of the workstation is cleared before traversal.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Phigs_bg_clear* (C), *INTEGER* (FORTRAN)

Default: *Clear*

C enumeration: *PHIGS_BG_CLEAR*, *PHIGS_BG_NO_CLEAR*

FORTRAN enumeration: *PHIGSREGBGCLR*, *PHIGSREGBGNOCLR*

PHIGS_REGION_PRIORITY (C), *PHIGSRGNPRIOR* (FORTRAN)
The priority of the region workstation or workstations within the drawable. Values must be in the range of 0.0 to 1.0 (inclusive). When a collection of workstations on a drawable are updated together the workstations are updated from lowest to highest priority. The order in which region workstations with the same priority are updated is undefined.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Pfloat** (C), *REAL* (FORTRAN)

Default: 0.0

PHIGS_REGION_RECT (C), *PHIGSRGNRECT* (FORTRAN)
The *x* and *y* limits of the workstation's region of the drawable, expressed as the lower left and upper right corners of the region, in screen coordinates.

If a specified attribute would put the upper right corner below or to the left of the lower left corner, then the attribute is ignored.

Note: This workstation type attribute may not be supported when a workstation is running in remote PEX mode.

Data type: *Prect** (C), *REAL(4)* (FORTRAN)

Default: full window size

CGM Output Attributes

PHIGS_CGM_ENCODING (C), *PHIGSCGMENCODING* (FORTRAN)
The encoding to use in generating the CGM file. SunPHIGS will generate any of the three encodings defined by the CGM Standard (ANSI X3.122-1986).

Data type: *Phigs_cgm_encoding* (C), *INTEGER* (FORTRAN)

Default: *Binary*

C enumeration: *PHIGS_CGM_BINARY*, *PHIGS_CGM_CHARACTER*, *PHIGS_CGM_CLEAR_TEXT*

FORTRAN enumeration: *PHIGSCGMBINARY*, *PHIGSCGMCHAR*, *PHIGSCGMCLEAR*

PHIGS_CGM_METRIC_HEIGHT_PTR (C), *PHIGSCGMMTHEIGHT* (FORTRAN)

PHIGS_CGM_METRIC_WIDTH_PTR (C), *PHIGSCGMMTWIDTH* (FORTRAN)

The physical drawing size in meters. By default, SunPHIGS generates a metafile that is dimensionless and is correctly displayed at any size. If the height or width is set to a positive floating point value, then SunPHIGS generates a metafile with implied measure, and the picture is correctly displayed only at the indicated drawing size.

Note: The data type for the C binding is a pointer to a Pfloat; not simply a Pfloat.

Data type: *Pfloat∗* (C), *REAL* (FORTRAN)

Default: -1.0, -1.0 (Dimensionless)

PHIGS_CGM_VDC_PRECISION (C), *PHIGSCGMVDCPREC* (FORTRAN)

Sets the precision for CGM output to either low [0, 32767] or high [0, 2147483648].

Data type: *Phigs_cgm_vdc_precision* (C), *INTEGER* (FORTRAN)

Default: *Low*

C enumeration: *PHIGS_CGM_LOW_PREC*, *PHIGS_CGM_HIGH_PREC*

FORTRAN enumeration: *PCGMLOWPREC*, *PCGMHIGHPREC*

ERRORS

002	Ignoring function, function requires state (PHOP, ∗, ∗, ∗)
052	Ignoring function, workstation type not recognized by the implementation
–100	Ignoring function, workstation type is a default type or bound to a workstation and cannot be modified.

SEE ALSO

WORKSTATION TYPE CREATE (3P)
WORKSTATION TYPE DESTROY (3P)
WORKSTATION TYPE GET (3P)
OPEN WORKSTATION (3P)
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES (3P)
INQUIRE WORKSTATION CONNECTION AND TYPE (3P)
PHIGS WORKSTATION DESCRIPTION TABLE (7P)

Index