Overview

The *XView/NeWS library* is intended as cleaner/simpler method for *NeWS* canvas handling in *XView*. It is not intended to be a full C client interface for for PostScript rendering. Canvases are created using the familiar `xv_create()` call. Access to canvases not created as `PScanvases` is available through `ps_token_from_xid()`.

Repaint and resize procs can be provided by the developer, as they can on *XView* `Canvases`. The environment is set up for each of these calls so that the rendering appears on the correct canvas (`setcanvas`) and the graphics state is not effected (`gsave/grestore`). Integration of the X damage information is provided through `PSCANVAS_CLIPRECTS`.

At the present time, all of the Red Book PostScript operators are not automatically defined by the *CPS* include files. Offerings in the future may include a more complete set of definitions, but at this time, it will still be necessary to use *CPS* to define many necessary operations.

Below is the *CPS* routine needed for use with the C routine above. This routine flips the coordinate system over, so that the X co-ordinate system is used, and the image appears with the correct side up. The canvas is then filled with a color, and the text is displayed centered, at an angle

simple.cps

```
cdef simple_setup()
          clippath pathbbox                % x y w h
          2 copy 0 exch translate pop      % goto bottom
          1 -1 scale                       % fllip view
          1 fillcanvas                     % clear canvas
          .8 setgray
          2 div exch 2 div exch moveto     % goto center
          pop pop 22 rotate                % 22 degrees
```

XView/NeWS Library: Examples

Below is a sample program that uses the X*View/NeWS library* to create a PSCANVAS and display the word “NeWS” using *NeWS* PostScript rendering.

simple.c

```
/*
 *
 * simple.c -- a short example using the XView/NeWS library
 *
 * A PSCANVAS is created, then NeWS PostScript rendering is
 * used to display the word “NeWS” at an angle.
 * The text remains centered when the window is resized.
 *
 */
#include <xview/xview.h>
#include <xvps/pscanvas.h>
#include “simple.h”
Frame frame;
void pscanvas_repaint_proc(pscanvas, newstoken, dpy, xid,
                    xrects)
 PScanvas           pscanvas;
 NeWStoken          newstoken;
 Display            *dpy;
 Window             xid;
 Xv_xrectlist *xrects;
{
 xv_set(pscanvas, PSCANVAS_CLIPRECTS, xrects, NULL);
 simple_setup();
 ps_findfont(“Bembo-Bold”);
 ps_scalefont(97);
 ps_setfont();
 ps_show(“NeWS”);
}
main(argc, argv)
                    int argc;
                    char **argv;
{
 xv_init(XV_INIT_ARGS, argc, argv, 0);
 frame=xv_create(NULL, FRAME, NULL);
 xv_create(frame, PSCANVAS,
                     FRAME_LABEL, “XView/NeWS Library
                    Example”,
                     PSCANVAS_REPAINT_PROC,
                    pscanvas_repaint_proc,
                     PSCANVAS_SYNC, 5, NULL);
 window_fit(frame);
 xv_main_loop(frame);
}
```

ps_token_from_framebuffer(newstoken)

`newstoken` is set to a *NeWS* usertoken that points to the *NeWS* framebuffer (or root window).

```
void
ps_token_from_framebuffer(newstoken)
        NeWStoken     *newstoken;
```

ps_clear_cliprects()

Sets the clipping for the current canvas back to `NULL`, or the original shape of the window. This is equivalent to `xv_set(pscanvas, PSCANVAS_CLIPRECTS, NULL);` for the appropriate PScanvas. But this can be used for routines that are used on more than one PScanvas.

```
void
ps_clear_cliprects()
```

Additional *CPS* Macros

This appendix lists CPS macros defined for use with the XView/NeWS library. These macros supplement the existing set of *CPS* macros defined in `$OPENWINHOME/include/NeWS/psmacros.h` and covered in Chapter 7 of the *NeWS Programmer's Guide*.

ps_setcanvas()

Sets the current *NeWS* canvas to be the canvas pointed to by newstoken. Note: all repaint and resize procedures are bracketted with a `ps_gsave`, the appropriate `ps_setcanvas`, and a `ps_grestore`, so it is not necessary to do any of these during normal circumstances.

```
void
ps_setcanvas()
```

ps_token_from_xid(xid, newstoken)

`xid` is the X11 window identifier for a window. This macro retrieves a *NeWS* usertoken identifier for this window suitable for use with `ps_setcanvas`. If the value given is invalid, -1 is returned in the second argument. Note that *CPS* macros do not return values, instead they change values given on the argument list, so the appropriate address must be passed in.

```
void
ps_token_from_xid(xid, newstoken)
       Window        xid;
       NeWStoken     *newstoken;
```

ps_to_token(postscript_code, newstoken)

`newstoken` is set to a *NeWS* usertoken for the PostScript object passed as the first argument (a character string). The PostScript is checked first to see if it is a single object, if it is, `newstoken` is set to a usertoken pointing at it, if not, `newstoken` is set to -1. This is useful for making tokens for canvases created by user-defined *CPS* procedures: `ps_to_token("MyCanvas",&mycanvas)`

```
void
ps_to_token(postscript_code, newstoken)
       char          *postscript_code;
       NeWStoken     *newstoken;
```

Summary of Procedures and Macros

This appendix lists the procedures and macros in alphabetical order and describes each of them briefly. The description includes the call that uses the procedure or macro.

pscanvas_repaint_proc()

Client-supplied callback procedure called when getting a `WIN_REPAINT` event. Function is installed using the attribute `PSCANVAS_REPAINT_PROC`.

```
void
pscanvas_repaint_proc(pscanvas, newstoken, display, xid,
                    xrects)
      PScanvas      pscanvas;
      NeWStoken     newstoken;
      Display       *display;
      Window        xid;
      Xv_rectlist   *xrects;
```

`newstoken` is a *NeWS* usertoken identifier for the pscanvas window. `display` is the display handle to the X11 server connection. `xid` is the X11 window identifier for the pscanvas. The `xrects` structure contains an array of `XRectangles` and a count that specifies the repaint area for the pscanvas. This structure can be used to set the attribute `PSCANVAS_CLIPRECTS` to use the X11 damage as a clipping region. See Attribute Summary.

pscanvas_resize_proc()

Procedure called when pscanvas width or height changes. This function is installed using the attibute `PSCANVAS_RESIZE_PROC`.

```
void
pscanvas_resize_proc(pscanvas, width, height)
      PScanvas      pscanvas;
      int           width;
      int           height;
```

pscanvas_is_token()

Macro to determine whether a *NeWS* token contains a valid usertoken

```
void
pscanvas_is_token(newstoken)
      NeWStoken     newstoken;
```

Attribute Summary

PSCANVAS_CLIPRECTS

Sets the clipping for an *XView* PostScript canvas. Any subsquent PostScript rendering operations will be clipped to this region. Input is an *XView* rectangle list, with the special value `NULL` meaning no extra clipping is to be done.

Default:	`NULL`
Type:	`Xv_rectlist`
Procs:	`create, set`

PSCANVAS_NEWSTOKEN

Returns the unique *NeWS* identifier for the given PSCANVAS. See `pscanvas_is_token()`, `ps_setcanvas()` in *Summary of Procedures and Macros*

Type:	`NeWStoken`
Procs:	`get`

PSCANVAS_REPAINT_PROC

Names a procedure called when then PSCANVAS has been damaged and must be repaired. See `pscanvas_repaint_proc()` in *Summary of Procedures and Macros*

Default:	`NULL`
Procs:	`create, get, set`

PSCANVAS_RESIZE_PROC

Names a procedure called when the PSCANVAS width or height changes.
See `pscanvas_repaint_proc()` in *Summary of Procedures and Macros*

Default:	`NULL`
Procs:	`create, get, set`

PSCANVAS_SYNC

Specifies a CPS tag to be used when synchronizing the *NeWS* server connection.

Default:	`NULL`
Type:	`int`
Procs:	`create, get, set`

Data Types

PScanvas	Handle to an opaque structure that describes a PostScript canvas.
NeWStoken	Handle to a *NeWS* object residing on the server, in *NeWS* terminology, a `usertoken`.

Overview

The *XView/NeWS library* is intended as cleaner/simpler method for *NeWS* canvas handling in *XView*. It is not intended to be a full C client interface for for PostScript rendering. Canvases are created using the familiar `xv_create()` call. Access to canvases not created as `PScanvases` is available through `ps_token_from_xid()`.

Repaint and resize procs can be provided by the developer, as they can on *XView* `Canvases`. The environment is set up for each of these calls so that the rendering appears on the correct canvas (`setcanvas`) and the graphics state is not effected (`gsave/grestore`). Integration of the X damage information is provided through `PSCANVAS_CLIPRECTS`.

At the present time, all of the Red Book PostScript operators are not automatically defined by the *CPS* include files. Offerings in the future may include a more complete set of definitions, but at this time, it will still be necessary to use *CPS* to define many necessary operations.

Below is the *CPS* routine needed for use with the C routine above. This routine flips the coordinate system over, so that the X co-ordinate system is used, and the image appears with the correct side up. The canvas is then filled with a color, and the text is displayed centered, at an angle

simple.cps

```
cdef simple_setup()
          clippath pathbbox                % x y w h
          2 copy 0 exch translate pop      % goto bottom
          1 -1 scale                       % fllip view
          1 fillcanvas                     % clear canvas
          .8 setgray
          2 div exch 2 div exch moveto     % goto center
          pop pop 22 rotate                % 22 degrees
```

XView/NeWS Library: Examples

Below is a sample program that uses the X*View/NeWS library* to create a PSCANVAS and display the word “NeWS” using *NeWS* PostScript rendering.

simple.c

```
/*
 *
 * simple.c -- a short example using the XView/NeWS library
 *
 * A PSCANVAS is created, then NeWS PostScript rendering is
 * used to display the word “NeWS” at an angle.
 * The text remains centered when the window is resized.
 *
 */
#include <xview/xview.h>
#include <xvps/pscanvas.h>
#include “simple.h”
Frame frame;
void pscanvas_repaint_proc(pscanvas, newstoken, dpy, xid,
                     xrects)
 PScanvas            pscanvas;
 NeWStoken           newstoken;
 Display             *dpy;
 Window              xid;
 Xv_xrectlist *xrects;
{
 xv_set(pscanvas, PSCANVAS_CLIPRECTS, xrects, NULL);
 simple_setup();
 ps_findfont(“Bembo-Bold”);
 ps_scalefont(97);
 ps_setfont();
 ps_show(“NeWS”);
}
main(argc, argv)
                     int argc;
                     char **argv;
{
 xv_init(XV_INIT_ARGS, argc, argv, 0);
 frame=xv_create(NULL, FRAME, NULL);
 xv_create(frame, PSCANVAS,
                      FRAME_LABEL, “XView/NeWS Library
                     Example”,
                      PSCANVAS_REPAINT_PROC,
                     pscanvas_repaint_proc,
                      PSCANVAS_SYNC, 5, NULL);
 window_fit(frame);
 xv_main_loop(frame);
}
```

ps_token_from_framebuffer(newstoken)

`newstoken` is set to a *NeWS* usertoken that points to the *NeWS* framebuffer (or root window).

```
void
ps_token_from_framebuffer(newstoken)
      NeWStoken     *newstoken;
```

ps_clear_cliprects()

Sets the clipping for the current canvas back to `NULL`, or the original shape of the window. This is equivalent to `xv_set(pscanvas, PSCANVAS_CLIPRECTS, NULL);` for the appropriate PScanvas. But this can be used for routines that are used on more than one PScanvas.

```
void
ps_clear_cliprects()
```

Additional *CPS* Macros

This appendix lists CPS macros defined for use with the XView/NeWS library. These macros supplement the existing set of *CPS* macros defined in `$OPENWINHOME/include/NeWS/psmacros.h` and covered in Chapter 7 of the *NeWS Programmer's Guide*.

ps_setcanvas()

Sets the current *NeWS* canvas to be the canvas pointed to by newstoken. Note: all repaint and resize procedures are bracketted with a `ps_gsave`, the appropriate `ps_setcanvas`, and a `ps_grestore`, so it is not necessary to do any of these during normal circumstances.

```
void
ps_setcanvas()
```

ps_token_from_xid(xid, newstoken)

`xid` is the X11 window identifier for a window. This macro retrieves a *NeWS* usertoken identifier for this window suitable for use with `ps_setcanvas`. If the value given is invalid, -1 is returned in the second argument. Note that *CPS* macros do not return values, instead they change values given on the argument list, so the appropriate address must be passed in.

```
void
ps_token_from_xid(xid, newstoken)
       Window        xid;
       NeWStoken     *newstoken;
```

ps_to_token(postscript_code, newstoken)

`newstoken` is set to a *NeWS* usertoken for the PostScript object passed as the first argument (a character string). The PostScript is checked first to see if it is a single object, if it is, `newstoken` is set to a usertoken pointing at it, if not, `newstoken` is set to -1. This is useful for making tokens for canvases created by user-defined *CPS* procedures: `ps_to_token("MyCanvas",&mycanvas)`

```
void
ps_to_token(postscript_code, newstoken)
       char          *postscript_code;
       NeWStoken     *newstoken;
```

Summary of Procedures and Macros

This appendix lists the procedures and macros in alphabetical order and describes each of them briefly. The description includes the call that uses the procedure or macro.

pscanvas_repaint_proc()

Client-supplied callback procedure called when getting a `WIN_REPAINT` event. Function is installed using the attribute `PSCANVAS_REPAINT_PROC`.

```
void
pscanvas_repaint_proc(pscanvas, newstoken, display, xid,
                     xrects)
      PScanvas      pscanvas;
      NeWStoken     newstoken;
      Display       *display;
      Window        xid;
      Xv_rectlist   *xrects;
```

`newstoken` is a *NeWS* usertoken identifier for the pscanvas window. `display` is the display handle to the X11 server connection. `xid` is the X11 window identifier for the pscanvas. The `xrects` structure contains an array of `XRectangles` and a count that specifies the repaint area for the pscanvas. This structure can be used to set the attribute `PSCANVAS_CLIPRECTS` to use the X11 damage as a clipping region. See Attribute Summary.

pscanvas_resize_proc()

Procedure called when pscanvas width or height changes. This function is installed using the attibute `PSCANVAS_RESIZE_PROC`.

```
void
pscanvas_resize_proc(pscanvas, width, height)
      PScanvas      pscanvas;
      int           width;
      int           height;
```

pscanvas_is_token()

Macro to determine whether a *NeWS* token contains a valid usertoken

```
void
pscanvas_is_token(newstoken)
      NeWStoken     newstoken;
```

Attribute Summary

PSCANVAS_CLIPRECTS

Sets the clipping for an *XView* PostScript canvas. Any subsquent PostScript rendering operations will be clipped to this region. Input is an *XView* rectangle list, with the special value `NULL` meaning no extra clipping is to be done.

Default:	`NULL`
Type:	`Xv_rectlist`
Procs:	`create, set`

PSCANVAS_NEWSTOKEN

Returns the unique *NeWS* identifier for the given PSCANVAS. See `pscanvas_is_token()`, `ps_setcanvas()` in *Summary of Procedures and Macros*

Type:	`NeWStoken`
Procs:	`get`

PSCANVAS_REPAINT_PROC

Names a procedure called when then PSCANVAS has been damaged and must be repaired. See `pscanvas_repaint_proc()` in *Summary of Procedures and Macros*

Default:	`NULL`
Procs:	`create, get, set`

PSCANVAS_RESIZE_PROC

Names a procedure called when the PSCANVAS width or height changes.
See `pscanvas_repaint_proc()` in *Summary of Procedures and Macros*

Default:	`NULL`
Procs:	`create, get, set`

PSCANVAS_SYNC

Specifies a CPS tag to be used when synchronizing the *NeWS* server connection.

Default:	`NULL`
Type:	`int`
Procs:	`create, get, set`

Data Types

PScanvas	Handle to an opaque structure that describes a PostScript canvas.
NeWStoken	Handle to a *NeWS* object residing on the server, in *NeWS* terminology, a `usertoken`.