Appendix C: Drawarea Functions

drawarea_init_traverse()

This initialized a Display_list_arg structure used for traversing the display and returns the first Display_list_cmd (or NULL) on the list.

```
Display_list_cmd *drawarea_init_traverse(
        Drawarea            drawarea,
        Display_list_arg*   dl_arg);
```

display_list_append()

Append a new display list command onto the display list. This returns a pointer to the uninitialized allocated space (aligned to 8 byte boundary). Generally, this is used if you want to write application specific display list primitives.

```
Display_list_cmd *display_list_append(
        Drawarea            drawarea,
        Display_list_vec*   vec,
        int                 size);
```

drawarea_next_traverse()

Returns the next Display_list_cmd (or NULL) on the list.

```
Display_list_cmd *drawarea_next_traverse(
        Display_list_arg*   dl_arg);
```

DDrawArc()

Draws an arc fitting inside the rectangle at (x1, y1), width w, height h. The start_angle and end_angle are measured in 64th of a degree. (See XDrawArc.) Uses the configurable (double) coordinate system.

```
void DDrawArc(
        Drawarea     drawarea,
        double       x1,
        double       y1,
        double       w,
        double       h,
        int          start_angle,
        int          end_angle);
```

DDrawImage()

Draws a XView server image at (x1, y1). The image may be multicolored, but the mask should be mono. No scaling of the image is done during rendering. If mask is not NULL, only those bits that are on in the mask will be painted. Uses the configurable (double) coordinate system.

```
void DDrawImage(
        Drawarea     drawarea,
        double       x1,
        double       y1,
        Server_image image,
        Server_image mask);
```

DDrawLine()

Draws a line from (x1, y1) to (x2, y2). (See XDrawLine.) Uses the configurable (double) coordinate system.

```
void DDrawLine(
      Drawarea      drawarea,
      double        x1,
      double        y1,
      double        x2,
      double        y2);
```

DDrawLines()

Draws the lines as specified in points. The underlying Xlib routine is XDrawLines, with mode set to CoordModeOrigin. (See XDrawLines.) Uses the configurable (double) coordinate system.

```
void DDrawLines(
      Drawarea      drawarea,
      DPoint        *points,
      int           npoints);
```

DDrawPoly()

Draws the outline of a polygon as specified by points and closes the polygon. It has the same semantics as DDrawLines, except it closes the polygon by connecting the beginning and ending points. Uses the configurable (double) coordinate system.

```
void DDrawPoly(
      Drawarea      drawarea,
      DPoint        *points,
      int           npoints);
```

DDrawRectangle()

Draws a rectangle at (x1, y1) with width w, and height h. (See XDrawRectangle) Uses the configurable (double) coordinate system.

```
void DDrawRectangle(
      Drawarea      drawarea,
      double        x1,
      double        y1,
      double        w,
      double        h);
```

DDrawString()

Draws a string at x1, baseline at y1. (See XDrawImageString.) Uses the configurable (double) coordinate system.

```
void DDrawString(
      Drawarea      drawarea,
      double        x1,
      double        y1,
      char          *string,
      int           strlen);
```

DFillRectangle()

Fills a rectangle at (x1, y1) with width w, height h. (See XDrawRectangle.) Uses the configurable (double) coordinate system.

```
void DFillRectangle(
      Drawarea     drawarea,
      double       x1,
      double       y1,
      double       w,
      double       h);
```

DFillArc()

Fills an arc fitting inside the rectangle at (x1, y1), width w, height h. The start_angle and end_angle are measured in 64th of a degree. (See XFillArc.) Uses the configurable (double) coordinate system.

```
void DFillArc(
      Drawarea     drawarea,
      double       x1,
      double       y1,
      double       w,
      double       h,
      int          start_angle,
      int          end_angle);
```

DFillPoly()

Fills a region closed by points. (See XFillPolygon.) Uses the configurable (double) coordinate system.

```
void DFillPoly(
      Drawarea     drawarea,
      DPoint       *points,
      int          npoints);
```

VClear()

Clears all entries from the display list.

```
void VClear(
      Drawarea     drawarea);
```

VDrawArc()

Draws an arc fitting inside the rectangle at (x1, y1), width w, height h. The start_angle and end_angle are measured in 64th of a degree. (See XDrawArc.) Uses the [0..10000][0..10000] coordinate system.

```
void VDrawArc(
      Drawarea     drawarea,
      short        x1,
      short        y1,
      short        w,
      short        h,
      int          start_angle,
      int          end_angle);
```

VDrawImage()

Draws a XView server image at (x1, y1). The image may be multi colored, but the mask should be mono. No scaling of the image is done during rendering. If mask is non-NULL, only those bits that are on in mask will be painted. Uses the [0..10000][0..10000] coordinate system.

```
void VDrawImage(
       Drawarea      drawimage,
       short         x1,
       short         y1,
       Server_image  image,
       Server_image  mask);
```

VDrawLine()

Draws a line from (x1, y1) to (x2, y2). (See XDrawLine.) Uses the [0..10000][0..10000] coordinate system.

```
void VDrawLine(
       Drawarea      drawarea,
       short         x1,
       short         y1,
       short         x2,
       short         y2);
```

VDrawLines()

Draws the lines as specified in points. The underlying Xlib routine is XDrawLines, with mode set to CoordModeOrigin. (See XDrawLines) Uses the [0..10000][0..10000] coordinate system.

```
void VDrawLines(
       Drawarea      drawarea,
       XPoint        *points,
       int           npoints);
```

VDrawPoly()

Draws the outline of a polygon as specified by points and closes the polygon. It has the same semantics as VDrawLines, except it closes the polygon by connecting the beginning and ending points. Uses the [0..10000][0..10000] coordinate system.

```
void VDrawPoly(
       Drawarea      drawarea,
       XPoint        *points,
       int           npoints);
```

VDrawRectangle()

Draws a rectangle at (x1, y1) with width w, and height h. (See XDrawRectangle.) Uses the [0..10000][0..10000] coordinate system.

```
void VDrawRectangle(
       Drawarea      drawarea,
       short         x1,
       short         y1,
       short         w,
       short         h);
```

VDrawString()

Draws a string at x1, baseline at y1. (See XDrawImageString.) Uses the [0..10000][0..10000] coordinate system.

```
void VDrawString(
      Drawarea      drawarea,
      short         x1,
      short         y1,
      char*         string,
      int           length);
```

VFillArc()

Fills an arc fitting inside the rectangle at (x1, y1), width w, height h. The start_angle and end_angle are measured in 64th of a degree. (See XFillArc.) Uses the [0..10000][0..10000] coordinate system.

```
void VFillArc(
      Drawarea      drawarea,
      short         x1,
      short         y1,
      short         w,
      short         h,
      int           start_angle,
      int           end_angle);
```

VFillPoly()

Fills a region closed by points. (See XFillPolygon.) Uses the [0..10000][0..10000] coordinate system.

```
void VFillPoly(
      Drawarea      drawarea,
      XPoint        *points,
      int           npoints);
```

VFillRectangle()

Fills a rectangle at (x1, y1) with width w, and height h. (See XFillRectangle.) Uses the [0..10000][0..10000] coordinate system.

```
void VFillRectangle(
      Drawarea      drawarea,
      short         x1,
      short         y1,
      short         w,
      short         h);
```

VFlush()

Flushes drawing commands that have not been painted since the last refresh of the object. To enhance performance, this should only be used in those situations where no other refresh is pending (such as those generated by xv_set).

```
void VFlush(
      Drawarea      drawarea);
```

VSetBgColor()

Sets the background color to be used for future rendering commands. The color is an index into the cms of the window. Setting it to -1 will set it to the default (RECTOBJ_BG_COLOR).

```
void VSetBgColor(
        Drawarea      drawarea,
        int           color);
```

VSetColor()

Sets the foreground color to be used for future rendering commands. The color is an index into the cms of the window. Setting it to -1 will set it to the default (RECTOBJ_FG_COLOR).

```
void VSetColor(
        Drawarea      drawarea,
        int           color);
```

VSetCms()

Allows a seperate cms to be used. Color indicies referenced in VSetForeground and VSetBackground will use the colors from the most recently set cms.

```
void VSetCms(
        Drawarea      drawarea,
        Cms           cms);
```

VSetFillStyle()

Sets the fill style to be used by following fill commands. (See XSetFillStyle.)

```
void VSetFillStyle(
        Drawarea      drawarea,
        int           fillstyle);
```

VSetFont()

Sets the font to be used in following VDrawString and DDrawString commands. The default font is the CANVAS_SHELL_FONT.

```
void VSetFont(
        Drawarea      drawarea,
        Xv_Font       font);
```

VSetLineStyle()

Sets the line style to be used by following line drawing commands. (See XSetLineAttributes.)

```
void VSetLineStyle(
        Drawarea      drawarea,
        int           linestyle));
```

VSetLineWidth()

Sets the line thickness to be used by following drawing commands. The line width are the same as in the X coordinate system. (See XSetLineAttributes.)

```
void VSetLineWidth(
        Drawarea      drawarea,
        int           linewidth);
```

VSetMarkKey()

Puts a marker with an opaque key into the display list. This function does not render any image, but can be used by the application to segregate portions of the display list. For example, if you have `DRAWAREA_MAP_EVENTS` set to `DRAWAREA_MAP_LAST`, getting `DRAWAREA_MARK_KEY` in an event callback will return the opaque key for the section of the display list that the event occurred on. The key is defined by the application and might be a pointer or a integer, but it is advisable to keep keys unique within a display list.

```
void VSetMarkKey(
       Drawarea      drawarea,
       Xv_opaque     key);
```

VSetStipple()

Sets the stipple to be used by following fill commands. (See XSetStipple.)

```
void VSetStipple(
       Drawarea      drawimage,
       Server_image  image);
```

VSetTile()

Sets the tile to be used by following fill commands. (See XSetTile.)

```
void VSetTile(
       Drawarea      drawimage,
       Server_image  image);
```