Appendix B: Functions

canvas_shell_split_proc()

This is the default OPENWIN_SPLIT_INIT_PROC for a canvas_shell. If your application has its own split init proc, make sure the split init proc calls this function.

```
void canvas_shell_split_proc(
       Xv_window     origview,
       Xv_window     newview,
       int           pos)
```

event_to_rectobj()

Given an event and a rectobj to identify the canvas shell that the event occured in, this function will return the rectobj that the event occured on.

```
Rectobj event_to_rectobj(
       Rectobj       rectobj,
       Event         *event);
```

rectobj_add_to_selected_list()

```
void   rectobj_add_to_selected_list(
       Rectobj       rectobj,
       int           exclusive,
       Event         *event);
```

rectobj_button_event_proc()

```
void   rectobj_button_event_proc(
       Xv_window     paint_window,
       Event         *event,
       Canvas_shell  canvas_shell,
       Rectobj       rectobj);
```

rectobj_del_from_selected_list()

```
void   rectobj_del_from_selected_list(
       Rectobj       rectobj,
       Event         *event);
```

rectobj_delta_move_children()

Moves all the children of a rectobj by calculating the new position using the delta_x and delta_y and then calling the child's set_geometry_proc. Typically used by classes that manage geometry of children.

```
void   rectobj_delta_move_children(
       Rectobj       rectobj,
       int           delta_x,
       int           delta_y);
```

rectobj_destroy_children()

Calls xv_destroy on all the children of a rectobj.

```
void   rectobj_destroy_children(
              Rectobj       rectobj);
```

rectobj_finish_set()

For package implementors only. Call at the end of the xv_set operation by subclasses of the rectobj to handle callbacks for geometry management and repainting.

```
void   rectobj_finish_set(
               Rectobj      rectobj);
```

rectobj_finish_set1()

For package implementors only. Call at the end of the xv_set operation by subclasses of the rectobj to handle callbacks for managing children (add_child_proc and del_child_proc). This gives a finer grain of control than rectobj_finish_set(). This detects when xv_set() has been called recursively and returns TRUE if it is actually the last xv_set. If TRUE is returned, rectobj_finish_set2() should be called.

```
int    rectobj_finish_set1(
               Rectobj      rectobj);
```

rectobj_finish_set2()

For package implementors only. Call when rectobj_finish_set1() returns TRUE. This invokes geometry management functions if the object's dimensions have changed and invokes repaint functions for damaged areas.

```
void   rectobj_finish_set2(
               Rectobj      rectobj);
```

rectobj_fit()

Calculates the minimum rect that contains an objects children and sets the objects XV_WIDTH and XV_HEIGHT accordingly.

```
void   rectobj_fit(
               Rectobj      rectobj);
```

rectobj_flush_repaint()

Calls the repaint functions of objects that need repainting in all canvas_shells in the application. If the force argument is FALSE, repainting may not occur if CANVAS_SHELL_DELAY_REPAINT is set to TRUE. If force is TRUE, the repainting will happen regardless.

```
void   rectobj_flush_repaint(
               int          force);
```

rectobj_geometry_manage()

For package implementors only. Requests a change in geometry by calling the object's parent's geometry_manage_proc. rectobj_finish_set2() calls this automatically if the geometry has changed during the xv_set operation, it may also be called by a geometry_manage_proc to request a new size. This returns TRUE if the desired_rect was set exactly as requested.

```
int    rectobj_geometry_manage(
               Rectobj      rectobj,
               Rect         *desired_rect);
```

rectobj_get_selected_list()

Returns the list of selected objects. There is one list per application. It is advisable to use xv_destroy_safe() rather than xv_destroy() if you are destroying objects on this list.

```
Rectobj_list *rectobj_get_selected_list(void);
```

To destroy all objects on a list:

```
{
        Rectobj_list *list;
        list = rectobj_get_selected_list();
        list_for(list)
                xv_destroy_safe(RECTOBJ_LIST_HANDLE(list));
}
```

rectobj_help_show()

Displays help for an object. Help is associated with a rectobj with the XV_HELP_DATA attribute. This function first looks for help data in the rectobj (argument), then checks it's owners until it finds help data. Usually applications don't need to call this function because displaying help is handled internally.

```
void   rectobj_help_show(
               Xv_Window    paint_window,
               Event        *event,
               Rectobj      rectobj);
```

rectobj_invalidate_repaint()

This function cancels the repainting of any damaged areas. The area that needed to be repainted is returned in the second argument and the r_width and r_height fields of *repaint_area* will be zero if there is nothing to be repainted. This is usually only useful if you have specific knowledge of what needs to be repainted. This must be used carefully because if you do not manually repaint damaged areas with rectobj_repaint_rect() and rectobj_flush_repaint() the state of the application may appear incorrect.

```
void   rectobj_invalidate_repaint(
               Rectobj      rectobj,
               Rect         *repaint_area);
```

rectobj_menu_show()

Displays the menu for an object. Menus are associated with a rectobj with the RECTOBJ_MENU attribute. This function first looks for the menu in the rectobj (argument), then looks in it's owners until it finds a menu. Usually applications don't need to call this function because displaying menus are handled internally.

```
void   rectobj_menu_show(
               Xv_Window    paint_window,
               Event        *event,
               Rectobj      rectobj);
```

rectobj_min_enclosing_rect()

Calculates the minimum enclosing rect for a list of rectobjs. See also rectobj_fit().

```
void   rectobj_min_enclosing_rect(
               Rectobj_list *list,
               Rect*        return_rect);
```

rectobj_move_children()

For package implementors only. Move children according to how the parent has been moved. This is generally used during a set_geometry_proc.

```
void   rectobj_move_children(
               Rectobj      rectobj);
```

rectobj_paint_child()

Calls the paint_proc of the child if the damaged area intersect the child's rect.

```
void   rectobj_paint_child(
               Rectobj      child,
               Display      *display,
               Window       window,
               Xv_xrectlist *damage);
```

rectobj_paint_children()

Calls the paint_procs of a rectobj's children when they intersect the damaged area.

```
void   rectobj_paint_children(
               Rectobj      rectobj,
               Display      *display,
               Window       window,
               Xv_xrectlist *damage);
```

rectobj_process_drop_event()

For use by event handlers. Handles ACTION_DRAG events and calls the rectobj's drop_proc.

```
void   rectobj_process_drop_event(
               Xv_window    paint_window,
               Event        *event,
               Canvas_shell canvas_shell,
               Rectobj      rectobj);
```

rectobj_repaint_rect()

Marks a rect of the canvas_shell that rectobj is on as damaged. If clear is set, the area needs a full repaint and will be cleared to background color before repainting. If damaged_rect is NULL, the XV_RECT of the rectobj will be used as the damaged rect.

```
void   rectobj_repaint_rect(
               Rectobj      rectobj,
               Rect         *damaged_rect,
               int          clear);
```

rectobj_reset_set_info()

For package implementors only. Resets internal rectobj data structures in case of error.

```
void   rectobj_reset_set_info(
               Rectobj      rectobj);
```

rectobj_selection_event_proc()

Used as the default event_proc by many classes, it supports the single click, double click, menu, help, start_drag callbacks and maintains the selected list. Applications may set the RECTOBJ_EVENT_PROC to this function if the default is something else.

```
void   rectobj_selection_event_proc(
              Xv_window     paint_window,
              Event         *event,
              Canvas_shell  canvas_shell,
              Rectobj       rectobj);
```

rectobj_set_delay_repaint()

Turns on/off immediate repaint for the canvas_shell that rectobj is on. Equivalent to CANVAS_SHELL_DELAY_REPAINT.

```
void   rectobj_set_delay_repaint(
              Rectobj       rectobj,
              int           flag);
```

rectobj_set_event_grab()

For event handlers only. Redirects all future events on the canvas_shell's paint windows to callback_function, regardless of what rectobj they map to. This is useful for focusing events while waiting for a specific event. Setting the function to NULL restores the event processing to normal.

```
void   rectobj_set_event_grab(
              Canvas_shell  canvas_shell,
              Rectobj       rectobj,
              Proc_ptr      callback_function,
              void          *associated_data);
```

rectobj_set_geometry()

For package implementors only. Sets the size and position of rectobj to newrect. Usually used to manipulate children during geometry management.

```
void   rectobj_set_geometry(
              Rectobj       rectobj,
              Rect          *newrect);
```

rectobj_set_paint_style()

Set the paint style for an object.

```
void   rectobj_set_paint_style(
              Rectobj       rectobj,
              Event         *event,
              Attr_attributenew_style);
```

rectobj_set_stacking_position()

Sets the stacking position for an object.

```
void   rectobj_set_stacking_position(
              Rectobj       rectobj,
              int           new_pos);
```

rectobj_toggle_event_proc()

Causes the object to act as a two state toggle. To use this, set RECTOBJ_EVENT_PROC to this function.

```
void   rectobj_toggle_event_proc(
               Xv_window     paint_window,
               Event         *event,
               Canvas_shell  canvas_shell,
               Rectobj       rectobj);
```

rectobj_upsearch()

Walks up the rectobj hierarchy and calls xv_get with get_attribute and get_arg. Returns the first object that the return value from xv_get is non-NULL and sets return_value to that value. If the canvas_shell is reached and no non-NULL value has been returned, rectobj_upsearch returns NULL.

```
Rectobj rectobj_upsearch(
               Rectobj       rectobj,
               Xv_opaque     *return_value,
               Attr_attributeget_attribute,
               int           get_arg);
```

traverse_rectobj_tree()

Recursively walks down the rectobj tree calling function for each rectobj in the tree.

```
void   *traverse_rectobj_tree(
               Rectobj       rectobj,
               void*(*)()    traverse_function_arg,
               void          *call_data_arg);
```

Function has the following form:

```
void   *traverse_function(
               Rectobj       rectobj,
               void          *call_data);
```

The call_data_arg is passed to the traversal callback function unchanged. If the traversal callback function ever returns non-NULL, the traversal is stopped and the return value is returned to the caller of traverse_rectobj_tree.