

**NAME**

XtPopup, XtPopupSpringLoaded, XtCallbackNone, XtCallbackNonexclusive, XtCallbackExclusive – map a pop-up

**SYNTAX**

```
#include <X11/Intrinsic.h>

void XtPopup(Widget popup_shell, XtGrabKind grab_kind);
void XtPopupSpringLoaded(Widget popup_shell);
void XtCallbackNone(Widget w, XtPointer client_data, XtPointer call_data);
void XtCallbackNonexclusive(Widget w, XtPointer client_data, XtPointer call_data);
void XtCallbackExclusive(Widget w, XtPointer client_data, XtPointer call_data);
```

**ACTIONS**

```
void XtMenuPopup(String shell_name);
```

**ARGUMENTS**

<i>call_data</i>	Specifies the callback data, which is not used by this procedure.
<i>client_data</i>	Specifies the pop-up shell.
<i>grab_kind</i>	Specifies the way in which user events should be constrained.
<i>popup_shell</i>	Specifies the widget shell.
<i>w</i>	Specifies the widget.

**DESCRIPTION**

The **XtPopup** function performs the following:

- Calls **XtCheckSubclass** to ensure *popup\_shell* is a subclass of **Shell**.
- Generates an error if the shell's *popped\_up* field is already **True**.
- Calls the callback procedures on the shell's *popup\_callback* list.
- Sets the shell *popped\_up* field to **True**, the shell *spring\_loaded* field to **False**, and the shell *grab\_kind* field from *grab\_kind*.
- If the shell's *create\_popup\_child* field is non-NULL, **XtPopup** calls it with *popup\_shell* as the parameter.
- If *grab\_kind* is either **XtGrabNonexclusive** or **XtGrabExclusive**, it calls:
 

```
XtAddGrab(popup_shell, (grab_kind == XtGrabExclusive), False)
```
- Calls **XtRealizeWidget** with *popup\_shell* specified.
- Calls **XMapWindow** with *popup\_shell* specified.

The **XtPopupSpringLoaded** function performs exactly as **XtPopup** except that it sets the shell *spring\_loaded* field to **True** and always calls **XtAddGrab** with *exclusive* **True** and *spring\_loaded* **True**.

The **XtCallbackNone**, **XtCallbackNonexclusive**, and **XtCallbackExclusive** functions call **XtPopup** with the shell specified by the client data argument and *grab\_kind* set as the name specifies. **XtCallbackNone**, **XtCallbackNonexclusive**, and **XtCallbackExclusive** specify **XtGrabNone**, **XtGrabNonexclusive**, and **XtGrabExclusive**, respectively. Each function then sets the widget that executed the callback list to be insensitive by using **XtSetSensitive**. Using these functions in callbacks is not required. In particular, an application must provide customized code for callbacks that create pop-up shells dynamically or that must do more than desensitizing the button.

**XtMenuPopup** is known to the translation manager, which must perform special actions for spring-loaded pop-ups. Calls to **XtMenuPopup** in a translation specification are mapped into calls to a nonexported action procedure, and the translation manager fills in parameters based on the event specified on the left-hand side of a translation.

If **XtMenuPopup** is invoked on **ButtonPress** (possibly with modifiers), the translation manager pops up the shell with `grab_kind` set to **XtGrabExclusive** and `spring_loaded` set to **True**. If **XtMenuPopup** is invoked on **EnterWindow** (possibly with modifiers), the translation manager pops up the shell with `grab_kind` set to **XtGrabNonexclusive** and `spring_loaded` set to **False**. Otherwise, the translation manager generates an error. When the widget is popped up, the following actions occur:

- Calls **XtCheckSubclass** to ensure `popup_shell` is a subclass of **Shell**.
- Generates an error if the shell's `popped_up` field is already **True**.
- Calls the callback procedures on the shell's `popup_callback` list.
- Sets the shell `popped_up` field to **True** and the shell `grab_kind` and `spring_loaded` fields appropriately.
- If the shell's `create_popup_child` field is non-NULL, it is called with `popup_shell` as the parameter.
- Calls:
 

```
XtAddGrab(popup_shell, (grab_kind == XtGrabExclusive), spring_loaded)
```
- Calls **XtRealizeWidget** with `popup_shell` specified.
- Calls **XMapWindow** with `popup_shell` specified.

(Note that these actions are the same as those for **XtPopup**.) **XtMenuPopup** tries to find the shell by searching the widget tree starting at the parent of the widget in which it is invoked. If it finds a shell with the specified name in the pop-up children of that parent, it pops up the shell with the appropriate parameters. Otherwise, it moves up the parent chain as needed. If **XtMenuPopup** gets to the application widget and cannot find a matching shell, it generates an error.

#### SEE ALSO

`XtCreatePopupShell(3)`, `XtPopdown(3)`  
*X Toolkit Intrinsic – C Language Interface*  
*Xlib – C Language X Interface*