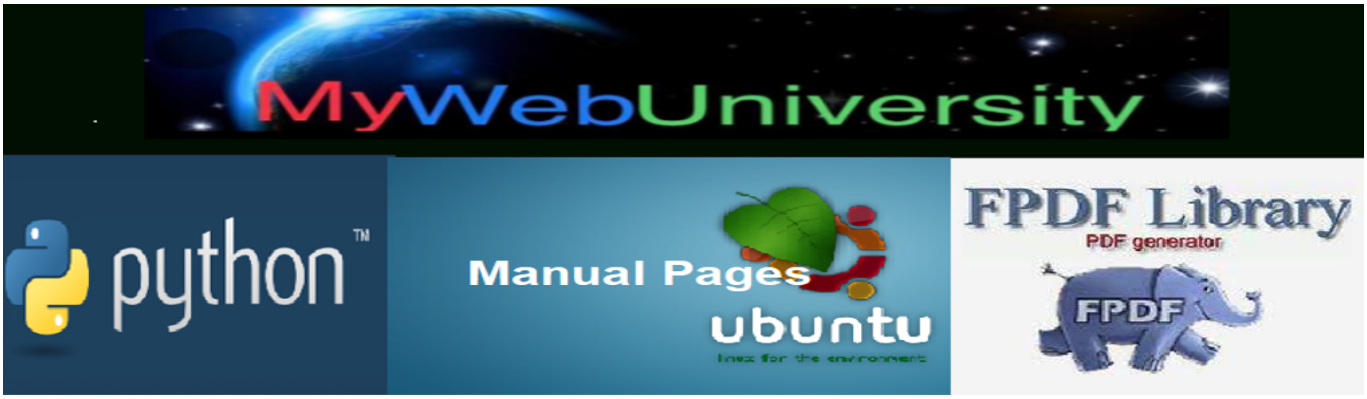

Rocky Enterprise Linux 9.2 Manual Pages on command 'XtUnmanageChildren.3'

$ man XtUnmanageChildren.3

XtManageChildren(3) XT FUNCTIONS XtManageChildren(3)

NAME

XtManageChildren, XtManageChild, XtUnmanageChildren, XtUnmanageChild, XtChangeManagedSet, XtIsManaged - manage and unmanage children

SYNTAX

```
#include <X11/Intrinsic.h>
typedef Widget *WidgetList;
void XtManageChildren(WidgetList children, Cardinal num_children);
void XtManageChild(Widget child);
void XtUnmanageChildren(WidgetList children, Cardinal num_children);
void XtUnmanageChild(Widget child);
void XtChangeManagedSet(WidgetList unmanage_children, Cardinal num_unmanage_children,
    XtDoChangeProc do_change_proc, XtPointer client_data, WidgetList manage_children,
    Cardinal num_manage_children);
Boolean XtIsManaged(Widget widget);
```

ARGUMENTS

child Specifies the child.

children Specifies a list of child widgets.

num_children

Specifies the number of children.

widget Specifies the widget.

manage_children

Specifies the list of widget children to add to the managed set.

num_manage_children

Specifies the number of entries in the manage_children list.

unmanage_children

Specifies the list of widget children to remove from the managed set.

num_unmanage_children

Specifies the number of entries in the unmanage_children list.

do_change_proc

Specifies the post unmanage, pre manage hook procedure to invoke.

client_data

Specifies the client data to be passed to the hook procedure.

DESCRIPTION

The XtManageChildren function performs the following:

? Issues an error if the children do not all have the same parent or if the parent is not a subclass of compositeWidgetClass.

? Returns immediately if the common parent is being destroyed; otherwise, for each unique child on the list, XtManageChildren ignores the child if it already is managed or is being destroyed and marks it if not.

? If the parent is realized and after all children have been marked, it makes some of the newly managed children viewable:

- Calls the change_managed routine of the widgets' parent.
- Calls XtRealizeWidget on each previously unmanaged child that is unrealized.
- Maps each previously unmanaged child that has map_when_managed True.

Managing children is independent of the ordering of children and independent of creating and deleting children. The layout routine of the parent should consider children whose managed field is True and should ignore all other children. Note that some composite wid? gets, especially fixed boxes, call XtManageChild from their insert_child procedure.

If the parent widget is realized, its change_managed procedure is called to notify it that its set of managed children has changed. The parent can reposition and resize any of its children. It moves each child as needed by calling XtMoveWidget, which first updates the x and y fields and then calls XMoveWindow if the widget is realized.

The XtManageChild function constructs a WidgetList of length one and calls XtManage? Children.

The XtUnmanageChildren function performs the following:

? Issues an error if the children do not all have the same parent or if the parent is not a subclass of compositeWidgetClass.

? Returns immediately if the common parent is being destroyed; otherwise, for each unique child on the list, XtUnmanageChildren performs the following:

- Ignores the child if it already is unmanaged or is being destroyed and marks it if not.
- If the child is realized, it makes it nonvisible by unmapping it.

? Calls the change_managed routine of the widgets' parent after all children have been marked if the parent is realized.

XtUnmanageChildren does not destroy the children widgets. Removing widgets from a par? ent's managed set is often a temporary banishment, and, some time later, you may manage the children again.

The XtUnmanageChild function constructs a widget list of length one and calls XtUnmanage? Children.

The XtChangeManagedSet function performs the following:

? Issues an error if the widgets specified in the manage_children and the unman? age_children lists to no all have the same parent, or if that parent is not a sub? class of compositeWidgetClass.

? Returns immediately if the common parent is being destroyed.

? If no CompositeClassExtension is defined, or a CompositeClassExtension is defined but with an allows_change_managed_set field with a value of False, and XtChangeManagedSet was invoked with a non-NULL do_change_proc procedure then XtChangeManagedSet performs the following:

- Calls XtUnmanageChildren (unmanage_children, num_unmanage_children).
- Calls the do_change_proc specified.
- Calls XtManageChildren (manage_children, num_manage_children) and then returns immediately.

? Otherwise, if a CompositeClassExtension is defined with an allows_change_managed_set field with a value of True, or if no CompositeClassExtension is defined, and XtChangeManagedSet was invoked with a NULL do_change_proc procedure, then the follow? ing is performed:

- For each child on the unmanage_children list; if the child is already unmanaged or is being destroyed it is ignored, otherwise it is marked as being unmanaged

and if it is realized it is made nonvisible by being unmapped.

- If the do_change_proc procedure is non-NULL then it is invoked as specified.
- For each child on the manage_children list; if the child is already managed or it is being destroyed it is ignored, otherwise it is marked as managed

? If the parent is realized and after all children have been marked, the change_managed method of the parent is invoked and subsequently some of the newly managed children are made viewable by:

- Calling XtRealizeWidget on each of the previously unmanaged child that is unre? alized.
- Mapping each previously unmanaged child that has map_when_managed True.

The XtIsManaged function returns True if the specified widget is of class RectObj or any subclass thereof and is managed, or False otherwise.

SEE ALSO

XtMapWidget(3), XtRealizeWidget(3)

X Toolkit Intrinsics - C Language Interface

Xlib - C Language X Interface