



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

***Rocky Enterprise Linux 9.2 Manual Pages on command 'XtVaSetSubvalues.3'***

***\$ man XtVaSetSubvalues.3***

XtSetValues(3) XT FUNCTIONS XtSetValues(3)

**NAME**

XtSetValues, XtVaSetValues, XtSetSubvalues, XtVaSetSubvalues, XtGetValues, XtVaGetValues, XtGetSubvalues, XtVaGetSubvalues - obtain and set widget resources

**SYNTAX**

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

void XtSetValues(Widget w, ArgList args, Cardinal num_args);

void XtVaSetValues(Widget w, ...);

void XtSetSubvalues(XtPointer base, XtResourceList resources, Cardinal num_resources, ArgList args, Cardinal num_args);

void XtVaSetSubvalues(XtPointer base, XtResourceList resources, Cardinal num_resources, ...);

void XtGetValues(Widget w, ArgList args, Cardinal num_args);

void XtVaGetValues(Widget w, ...);

void XtGetSubvalues(XtPointer base, XtResourceList resources, Cardinal num_resources, ArgList args, Cardinal num_args);

void XtVaGetSubvalues(XtPointer base, XtResourceList resources, Cardinal num_resources, ...);
```

**ARGUMENTS**

**args** Specifies the argument list of name/address pairs that contain the resource name and either the address into which the resource value is to be stored or their new values.

**base** Specifies the base address of the subpart data structure where the resources

should be retrieved or written.

num\_args Specifies the number of arguments in the argument list.

resources Specifies the nonwidget resource list or values.

num\_resources

Specifies the number of resources in the resource list.

w Specifies the widget.

... Specifies the variable argument list of name/address pairs that contain the re?

source name and either the address into which the resource value is to be stored or their new values.

## DESCRIPTION

The XtSetValues function starts with the resources specified for the Core widget fields and proceeds down the subclass chain to the widget. At each stage, it writes the new value (if specified by one of the arguments) or the existing value (if no new value is specified) to a new widget data record. XtSetValues then calls the set\_values procedures for the widget in superclass-to-subclass order. If the widget has any non-NULL set\_val? ues\_hook fields, these are called immediately after the corresponding set\_values procedure. This procedure permits subclasses to set nonwidget data for XtSetValues.

If the widget's parent is a subclass of constraintWidgetClass, XtSetValues also updates the widget's constraints. It starts with the constraint resources specified for constraintWidgetClass and proceeds down the subclass chain to the parent's class. At each stage, it writes the new value or the existing value to a new constraint record. It then calls the constraint set\_values procedures from constraintWidgetClass down to the parent's class. The constraint set\_values procedures are called with widget arguments, as for all set\_values procedures, not just the constraint record arguments, so that they can make adjustments to the desired values based on full information about the widget.

XtSetValues determines if a geometry request is needed by comparing the current widget to the new widget. If any geometry changes are required, it makes the request, and the geometry manager returns XtGeometryYes, XtGeometryAlmost, or XtGeometryNo. If XtGeometryYes, XtSetValues calls the widget's resize procedure. If XtGeometryNo, XtSetValues resets the geometry fields to their original values. If XtGeometryAlmost, XtSetValues calls the set\_values\_almost procedure, which determines what should be done and writes new values for the geometry fields into the new widget. XtSetValues then repeats this process, deciding once more whether the geometry manager should be called.

Finally, if any of the `set_values` procedures returned `True`, `XtSetValues` causes the widget's `expose` procedure to be invoked by calling the `Xlib XClearArea` function on the widget's window.

The `XtSetSubvalues` function stores resources into the structure identified by `base`.

The `XtGetValues` function starts with the resources specified for the core widget fields and proceeds down the subclass chain to the widget. The `value` field of a passed argument list should contain the address into which to store the corresponding resource value. It is the caller's responsibility to allocate and deallocate this storage according to the size of the resource representation type used within the widget.

If the widget's parent is a subclass of `constraintWidgetClass`, `XtGetValues` then fetches the values for any constraint resources requested. It starts with the constraint resources specified for `constraintWidgetClass` and proceeds down to the subclass chain to the parent's constraint resources. If the argument list contains a resource name that is not found in any of the resource lists searched, the value at the corresponding address is not modified. Finally, if the `get_values_hook` procedures are non-`NULL`, they are called in superclass-to-subclass order after all the resource values have been fetched by `XtGetValues`. This permits a subclass to provide nonwidget resource data to `XtGetValues`.

The `XtGetSubvalues` function obtains resource values from the structure identified by `base`.

#### SEE ALSO

X Toolkit Intrinsic - C Language Interface

Xlib - C Language X Interface

X Version 11

libXt 1.2.1

`XtSetValues(3)`