

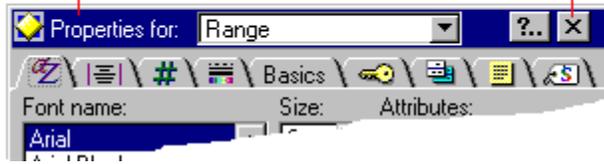
## Using the InfoBox

You can make changes to one control or several controls without closing the InfoBox.

1. Select what you want to modify.
2. Right-click the selection and choose the Properties command.
3. Click the tab for the properties you want to change.
4. Select one or more options.
5. (Optional) Move, collapse, or close the InfoBox.

**To move, collapse, or close the InfoBox**

- To move the InfoBox, drag here
- To collapse or reopen the InfoBox, double-click here
- To close the InfoBox, click here



## SetItemText Method

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Sets the text for an item of the list in a combo box.

## Syntax

[objectreference].SetItemText *index*, *text*

## Parameters

*index*

Integer. The item in the combo box. Use 0 as the value for this argument for the first item.

*text*

String. Text to replace the indexed item's text.

## Return values

None

## Usage

Changing the text for an item in a list when the Sorted property is set to TRUE may cause future entries to be sorted incorrectly.

## Dialog Controls LotusScript A-Z

A  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
O  
P  
Q  
R  
S  
T  
U  
V  
W  
X  
Y  
Z

### A

[Activate Event](#)  
[ActiveControl Property](#)  
[AddItem Method](#)  
[Alignment Property](#)  
[AlwaysOnTop Property](#)  
[Appearance Property](#)  
[AutoRedraw Property](#)  
[AutoSize Property](#)

### B

[BackColor Property](#)  
[BackStyle Property](#)  
[Bold Property](#)  
[BorderStyle Property](#)

### C

[Cancel Property](#)  
[Caption Property](#)  
[Change Event](#)  
[CharSet Property](#)  
[Clear Method](#)  
[ClearSel Method](#)  
[Click Event](#)  
[ClipControls Property](#)  
[Close Method](#)  
[Closing Event](#)  
[Columns Property](#)  
[ControlBox Property](#)  
[Controls Property](#)  
[Count Property](#)

## D

[DbfClick Event](#)  
[Deactivate Event](#)  
[Default Property](#)  
[Delay Property](#)  
[Description Property](#)  
[Displayed Property](#)  
[DoClick Method](#)  
[DropDown Event](#)

## E

[Enabled Property](#)

## F

[Font Class](#)  
[Font Property](#)  
[ForeColor Property](#)

## G

[GetLine Method](#)  
[GetNumTicks Method](#)  
[GotFocus Event](#)

## H

[Handle Property](#)  
[Height Property](#)  
[HelpContextID Property](#)  
[HelpFileName Property](#)  
[HideSelection Property](#)  
[Hide Method](#)  
[hPal Property](#)  
[hWnd Property](#)

## I

[ID Property](#)  
[IntegralHeight Property](#)  
[Italic Property](#)  
[IsItemSelected Method](#)  
[Item Method](#)  
[ItemData Property](#)  
[ItemDataSelected Property](#)

## J

(None)

## K

[KeyDown Event](#)  
[KeyPress Event](#)  
[KeyPreview Property](#)  
[KeyUp Event](#)

## L

[LargeChange Property](#)  
[Left Property](#)  
[LineCount Property](#)  
[List Property](#)  
[ListCount Property](#)  
[ListIndex Property](#)  
[Load Event](#)

[LoadImage Method](#)  
[Locked Property](#)  
[LostFocus Event](#)  
[LotusBaseObject Class](#)  
[LotusCheckBox Control](#)  
[LotusComboBox Control](#)  
[LotusCommandButton Control](#)  
[LotusControl Class](#)  
[LotusControls](#)  
[LotusDialog Control](#)  
[LotusFrame Control](#)  
[LotusImage Control](#)  
[LotusLabel Control](#)  
[LotusListBox Control](#)  
[LotusOptionButton Control](#)  
[LotusProgressBar Control](#)  
[LotusSlider Control](#)  
[LotusSpinButton Control](#)  
[LotusTextBox Control](#)

## M

[Max Property](#)  
[MaxLength Property](#)  
[Min Property](#)  
[Modal Property](#)  
[Move Method](#)  
[MouseDown Event](#)  
[MouseMove Event](#)  
[MousePointer Property](#)  
[MouseUp Event](#)  
[Moved Event](#)  
[MultiSelect Property](#)

## N

[Name Property](#)  
[NewIndex Property](#)

## O

[Orientation Property](#)

## P

[Parent Property](#)  
[PasswordChar Property](#)  
[Pick Event](#)  
[Picture Class](#)  
[Picture Property](#)

## Q

(None)

## R

[Refresh Method](#)  
[RemoveItem Method](#)  
[Render Method](#)  
[ReplaceItem Method](#)

## S

[Scroll Event](#)  
[ScrollBars Property](#)

[SelCount Property](#)  
 [Selected Property](#)  
 [SelectItem Method](#)  
 [SelectItemString Method](#)  
 [SelectItemText Method](#)  
 [SelectRange Property](#)  
 [SelLength Property](#)  
 [SelStart Property](#)  
 [SelText Property](#)  
 [SetFocus Method](#)  
 [SetItemData Method](#)  
 [SetItemText Method](#)  
 [Show Method](#)  
 [Size Property](#)  
 [SmallChange Property](#)  
 [Sorted Property](#)  
 [SpinDown Event](#)  
 [SpinOrientation Property](#)  
 [SpinUp Event](#)  
 [Stretch](#)  
 [Strikethrough Property](#)  
 [Style Property](#)

## **T**

[TabIndex Property](#)  
 [TabStop Property](#)  
 [Tag Property](#)  
 [Text Property](#)  
 [TickFrequency Property](#)  
 [TickStyle Property](#)  
 [Top Property](#)  
 [TopIndex Property](#)  
 [Type Property](#)

## **U**

[Underline Property](#)  
 [Unload Event](#)  
 [UseMnemonic Property](#)  
 [Value Property](#)

## **V**

[Version Property](#)  
 [Visible Property](#)

## **W**

[Weight Property](#)  
 [Width Property](#)  
 [WordWrap Property](#)

## **X, Y,**

(None)

## **Z**

[ZOrder Method](#)

## **Classes used in dialogs**

### **LotusDialog Class**

[LotusDialog](#)

### **Controls Classes**

[LotusCheckBox](#)

[LotusComboBox](#)

[LotusCommandButton](#)

[LotusFrame](#)

[LotusImage](#)

[LotusLabel](#)

[LotusListBox](#)

[LotusOptionButton](#)

[LotusProgressBar](#)

[LotusSlider](#)

[LotusSpinButton](#)

[LotusTextBox](#)

### **Base Classes**

[LotusBaseObject](#)

[LotusControl](#)

### **Other Classes**

[Font](#)

[Picture](#)

[LotusControls](#)



## Dialog Controls Events

A  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
O  
P  
Q  
R  
S  
T  
U  
V  
W  
X  
Y  
Z

### A

Activate

### B

(None)

### C

Change

Click

Closing

### D

DbClick

Deactivate

DropDown

### E

(None)

### F

GotFocus

### G, H, I, J

(None)

### K

KeyDown

KeyPress

KeyUp

### L

Load

LostFocus

**M**

MouseDown

MouseMove

MouseUp

Moved

**N, O**

(None)

**P**

Pick

**Q, R**

(None)

**S**

Scroll

SpinDown

SpinUp

**T**

(None)

**U**

Unload

**V, W, X, Y, Z**

(None)

## SetItemText Method

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;','0)} [See list of classes](#)

Sets the text for an item in the list of a list box.

### Syntax

[objectreference].SetItemText *index*, *text*

### Parameters

*index*

Integer. The item in the list box. Use 0 as the value for this argument for the first item.

*text*

String. Text to replace the indexed item's text.

### Return values

None

### Usage

Changing the text for an item in a list when the Sorted property is set to TRUE may cause future entries to be sorted incorrectly.

## Text Property

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

(Read-only)

Returns the text for the selected item in a text box.

## Data type

String

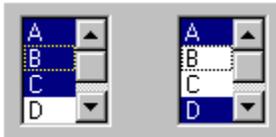
## Syntax

Textvalue = [objectreference].Text

## Usage

If nothing is selected, an empty string ("") is returned.

If the MultiSelect property is set to TRUE, the Text property returns the item with focus. For example, for each of the following list boxes, the Text property returns "B".



## Description Property

{button ,AL(^H\_LDC\_LOTUSBASEOBJECT\_LOTUSBASEOBJECT\_CLASS;H\_LDC\_LOTUSCHECKBOX\_LOTUSCHECK\_BOX\_CLASS;H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;H\_LDC\_LOTUSCOMMANDBUTTON\_LOTUSCOMMANDBUTTON\_CLASS;H\_LDC\_LOTUSCONTROL\_LOTUSCONTROL\_CLASS;H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;H\_LDC\_LOTUSFRAME\_LOTUSFRAME\_CLASS;H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_BOX\_CLASS;H\_LDC\_LOTUSOPTIONBUTTON\_LOTUSOPTIONBUTTON\_CLASS;H\_LDC\_LOTUSPROGRESSBAR\_LOTUSPROGRESSBAR\_CLASS;H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;H\_LDC\_LOTUSSPINBUTTON\_LOTUSSPINBUTTON\_CLASS;H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;','0)} [See list of classes](#)

Returns or sets a description of the dialog or a control within the dialog.

This property is read-only at run time.

## Data type

String

## Syntax

Descriptionvalue = [objectreference].Description

## Legal values

The string is limited to 256 characters.

## **LotusBaseObject**

This is a base class for all Lotus stock controls as well as LotusDialog.

### **Base classes**

None

### **Derived classes**

[LotusControl](#)

[LotusDialog](#)

### **Contained by**

None

### **Usage**

Since all Lotus dialog controls are derived from LotusControl, which is derived from the LotusBaseObject class, the LotusBaseObject class members appear in every control.

## LotusBaseObject members

### Properties

Description AS String

Name AS String

Version AS Long [Read-only]

### Methods

None

### Events

None

## Name Property

```
{button ,AL(^H_LDC_LOTUSBASEOBJECT_LOTUSBASEOBJECT_CLASS;H_LDC_LOTUSCHECKBOX_LOTUSCHECK_BOX_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSDIALOG_LOTUSDIALOG_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_BOX_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;',0)} See list of classes
```

Returns the name of the control or dialog. You can use the name to identify the control in your script.

This property is read-only at run time.

## Data type

String

## Syntax

Namevalue = [objectreference].Name

## Legal values

This string can contain a maximum length of 40 characters. It must start with a letter, but can contain numbers or underscore characters. It cannot include punctuation or spaces (i.e. it must be a valid LotusScript identifier).

## Usage

The default name of a dialog is "dialog" plus a number indicating the order in which you created the dialog.

The default name of a control is the kind of control plus a number indicating the order in which you created the control.

If the control is an OCX, the OLE "short name" is used, followed by a number indicating the order in which you created the control.



## Version Property

```
{button ,AL(^H_LDC_LOTUSBASEOBJECT_LOTUSBASEOBJECT_CLASS;H_LDC_LOTUSCHECKBOX_LOTUSCHECK_BOX_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSDIALOG_LOTUSDIALOG_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_BOX_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;','0)} See list of classes
```

(Read-only)

Returns the version number of the control or dialog.

## Data type

Long

## Syntax

Versionvalue = [objectreference].Version

## Usage

This property contains the major and minor version numbers for the control or dialog.

## Alignment Property

{button ,AL(^H\_LDC\_LOTUSCHECKBOX\_LOTUSCHECK\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value determining the side on which to display the check box Caption.

## Data type

Integer

## Syntax

Alignmentvalue = [objectreference].Alignment

[objectreference].Alignment = Alignmentvalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CHECKBOX_ALIGN_LEFT	Left aligned (default)
1	CHECKBOX_ALIGN_RIGHT	Right aligned

## LotusCheckBox



This control displays as a box which users can select to signify a true or false response. You can have any number of check boxes in a dialog. Use several check boxes to display multiple choices from which users can select one or more.

You can use the Caption property to display a caption next to a check box. Use the Value property to determine the state of the check box (unchecked, checked, or dimmed (greyed out)).

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

Use check boxes when you want a user to be able to make more than one selection. Use option buttons when you want a user to make a single selection from among several choices.

## LotusCheckbox members

### Properties

Alignment AS Integer  
Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
Value AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

DoClick  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Click  
GotFocus  
LostFocus

## **AddItem Method**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;',0)} [See list of classes](#)

Adds an item to a combo box.

### **Syntax**

[objectreference].AddItem *text*, [*index*, *itemdata*]

### **Parameters**

*text*

String. Expression specifying the item to be added to the combo box.

*index*

Long or Integer. Optional argument specifying the position in which to place the item in the combo box. To position an item as the first item, use 0 as the value for this argument. Leaving the argument blank (or giving it a value of -1 or a value equal to the ListCount property) appends the item to the end of the list or sorts it if the Sorted property is set to TRUE.

*itemdata*

Long or Integer. Optional argument specifying the item's data. Leaving this argument blank passes a value of 0.

### **Return values**

None

### **Usage**

When a specific index is used (from 1 to ListCount - 1), the entry is not sorted even if the Sorted property is set to TRUE. Note that adding an item out of order to a sorted list may cause future entries to sort incorrectly.

## Change Event

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Triggered when the edit portion of a combo box changes.

## Syntax

```
Sub Change(Source As Lotuscombobox)
```

## Parameters

*Source As Lotuscombobox*

The LotusComboBox object that generated the event.

## Usage

The following example makes whatever change is made to an item in the edit portion of a drop down combo box appear in the list the next time it is dropped down.

```
Dim x As Integer, y As Integer 'Added to (Globals).
Sub Change(Source As Lotuscombobox)
    x = True
End Sub
Sub Dropdown(Source As Lotuscombobox)
    If x = True Then
        source.removeitem y
        source.additem source.text, y
    End If
End Sub
Sub Pick(Source As Lotuscombobox, Index As Integer)
    x = False
    y = Index
End Sub
```

**Clear Method**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Clears the contents of the combo box.

**Syntax**

[objectreference].Clear

**Parameters**

None

**Return values**

None

## DropDown Event

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Triggered when the list in a combo box drops down. This event is not applicable for a simple combo box (a combo box with a Style property set to 1).

### Syntax

```
Sub Dropdown(Source As Lotuscombobox)
```

### Parameters

*Source As Lotuscombobox*

The combo box that dropped down.

### Usage

The following example makes whatever change is made to an item in the edit portion of a drop-down combo box appear in the list the next time it is dropped down.

```
Dim x As Integer, y As Integer 'Added to (Globals).
Sub Change(Source As Lotuscombobox)
    x = True
End Sub
Sub Dropdown(Source As Lotuscombobox)
    If x = True Then
        source.removeitem y
        source.additem source.text, y
    End If
End Sub
Sub Pick(Source As Lotuscombobox, Index As Integer)
    x = False
    y = Index
End Sub
```



## **IntegralHeight Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value determining whether the list portion of a combo box displays partial items or resizes to display complete items. This property is read-only at runtime.

### **Data type**

[Integer](#)

### **Syntax**

IntegralHeightvalue = [objectreference].IntegralHeight

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

TRUE = List resizes only to display complete items (default).

FALSE = List doesn't resize. Items at bottom may be clipped.

### **IsItemSelected Method**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns a value indicating whether the item at index is selected in the list in the combo box.

### **Syntax**

[objectreference].IsItemSelected(*index*)

### **Parameters**

*index*

Integer. The value of the ItemData property for the item in the list.

### **Return values**

TRUE = The item is selected.

FALSE = The item is not selected.

### **Usage**

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

## **ItemDataSelected Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets the data for a selected item in a combo box. Getting this property when there is no selected item returns a an error. Setting this property when there is no selected item does nothing.

### **Data type**

Variant

### **Syntax**

ItemDataSelectedvalue = [objectreference].ItemDataSelected

[objectreference].ItemDataSelected = ItemDataSelectedvalue

### **Legal values**

When set, this property's value must be convertible to a Long or a run-time error is raised.

## **ItemData Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

This parameterized property returns or sets a number associated with an item in the list in a combo box. This property is runtime-checked and its data type must be a long or an Integer; otherwise an error is returned.

### **Data type**

Variant

### **Syntax**

ItemDatavalue = [objectreference].ItemData(index)

[objectreference].ItemData(index) = ItemDatavalue

### **Parameters**

*index*

Integer. Index of the item. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

## **ListCount Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

(Read-only)

The number of items in the list of a combo box.

## **Data type**

[Integer](#)

## **Syntax**

ListCountvalue = [objectreference].ListCount

## **ListIndex Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets the zero-based index of the selected item from a list box. Returns -1 if no item is selected.

### **Data type**

Integer

### **Syntax**

ListIndexvalue = [objectreference].ListIndex

[objectreference].ListIndex = ListIndexvalue

### **Usage**

When setting, if the index is less than 0 or greater than or equal to the ListCount property, a run-time error is raised.

## List Property

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

This parameterized property represents the contents of the combo-box list. Each element in the array is an item in a list in the combo box.

### Data type

String

### Syntax

Listvalue = [objectreference].List(index)

[objectreference].List(index) = Listvalue

### Parameters

*index*

Integer, which is the zero-based index for the item.

### Usage

*When setting*

If the index is greater than or equal to 0 and less than the value of the ListCount property, the element is replaced with the new string.

To add to the combo list using the List property, use an index of -1 or an index equal to the ListCount property. For a sorted combo box, this will add the item in its sorted position. For a non-sorted combo, it will add the item to the end of the list.

If the index is less than -1 or greater than the value of the ListCount property, a run-time error is raised.

*When getting:*

If the index is less than 0 or greater than or equal to the ListCount property, a run-time error is raised.

## LotusComboBox



A combo box combines the features of a text box and a list box. The user makes an entry in a combo box either through (manually) typing in the text-box section or through selecting an item from the list-box section.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

Use the Style property to determine the type of combo box. There are three distinct styles of combo box and each style has some distinct behavior. Because the control is a union of the three behaviors, some of the properties and methods may not have meaning for a specific style.

The default event for a combo box is the Change event.



## LotusComboBox members

### Properties

Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long  
IntegralHeight AS Integer  
ItemData AS Variant  
ItemDataSelected AS Variant  
Left AS Long  
List AS String  
ListCount AS Integer [Read-only]  
ListIndex AS Integer  
MousePointer AS Long  
Name AS String  
NewIndex AS Integer [Read-only]  
Parent AS LotusDialog [Read-only]  
SelLength AS Long  
SelStart AS Long  
SelText AS String  
Sorted AS Integer  
Style AS Long  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Text AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

AddItem  
Clear  
DoClick  
IsItemSelected  
Move  
Refresh  
RemoveItem  
ReplaceItem  
SelectItem  
SelectItemString  
SetFocus  
SetItemData  
SetItemText  
ZOrder

### Events

Change  
Click

DblClick  
DropDown  
GotFocus  
LostFocus  
Pick

## **NewIndex Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

(Read-only)

Returns a value indicating the index of the item last added to the list in a combo box. Returns -1 if the list is empty or if the item was deleted.

## **Data type**

[Integer](#)

## **Syntax**

NewIndexvalue = [objectreference].NewIndex

## Pick Event

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Triggered when a specific item in the list in a combo box is selected.

## Syntax

```
Sub Pick(Source As Lotuscombobox, Index As Integer)
```

## Parameters

*Source As Lotuscombobox*

The combo box that generated the event.

*Index As Integer*

The number of a selected item in the combo box.

## Usage

The following example makes whatever change is made to an item in the edit portion of a drop-down combo box appear in the list the next time it drops down.

```
Dim x As Integer, y As Integer 'Added to (Globals).
Sub Change(Source As Lotuscombobox)
    x = True
End Sub
Sub Dropdown(Source As Lotuscombobox)
    If x = True Then
        source.removeitem y
        source.additem source.text, y
    End If
End Sub
Sub Pick(Source As Lotuscombobox, Index As Integer)
    x = False
    y = Index
End Sub
```

## **RemoveItem Method**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Removes an item from a combo box.

### **Syntax**

[objectreference].RemoveItem *index*

### **Parameters**

*index*

Integer. The zero-based index of the element to remove. To remove the first item, use 0 as the value for this argument.

### **Return values**

None

### **Usage**

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

## Replaceltem Method

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Replaces an item in a combo box.

### Syntax

[objectreference].Replaceltem *index*, *text* [,*itemdata*]

### Parameters

*index*

Long or Integer. Argument specifying the zero-based position of the item to be replaced in the combo box. To replace the first item, use 0 as the value for this argument. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

*text*

String. Expression specifying the item to be used for replacement.

*itemdata*

Variant (must be Long or Integer or error is returned). Optional argument specifying the item's data. Leaving this argument blank passes a value of 0.

### Return values

None

### Usage

Replacing an item in a list when the Sorted property is set to TRUE may cause future entries to be sorted incorrectly. To replace an item and maintain proper sorting, delete the old item using RemoveItem, and add the new one using AddItem (leaving its index argument blank or using -1).

## SelectItemString Method

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Searches and then selects the next item that starts with a string. The search can begin from any item in the list.

### Syntax

[objectreference].SelectItemString *text*, *index*

### Parameters

*text*

String. The text of the item to select.

*index*

Integer. The item in the combo box before the item from which to start. The first item has 0 as the value for this argument. The index must be greater than or equal to -1 and one less than the value of the ListCount property or a run-time error occurs. To start at the beginning, use an index of -1.

### Return values

Returns the index of the selected item. If no match is made, returns -1.

### Usage

When the search for the string reaches the end of the list, it wraps to the beginning. The search is case sensitive.

## SelectItem Method

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Makes the index the currently selected item.

## Syntax

[objectreference].SelectItem *index*

## Parameters

*index*

Integer. The item in the combo box. The first item has 0 as the value for this argument.

## Return values

None

## Usage

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.



## **SelLength Property**

{button ,AL('H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS';,0)} [See list of classes](#)

Returns or sets a value indicating the number of characters selected in the edit portion of the combo box. A combo box only has an edit portion if its Style property is set to 0 (dropdown combo) or 1 (simple combo).

The SelLength property is used in conjunction with the SelStart property to select a region of text.

### **Data type**

Long

### **Syntax**

SelLengthvalue = [objectreference].SelLength

[objectreference].SelLength = SelLengthvalue

### **Legal values**

Getting or setting the SelLength property when the Style property is set to 2 (dropdown list) raises a run-time error.

### **Usage**

The SelLength property selects characters to the right of the SelStart property's value. The SelStart property is set to 0 if not set or reset.

Setting a SelLength value of -1 selects from the SelStart property's value to the end of the edit portion.

Setting a SelLength value greater than the number of characters in the edit portion selects from the SelStart property's value to the end of the edit portion.

## **SelStart Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value indicating the starting point to select characters in the edit portion of a combo box. (A combo box only has an edit portion if its Style property is set to 0 (a dropdown combo) or 1 (a simple combo)).

The SelStart property is used in conjunction with the SelLength property to select a region of text.

### **Data type**

Long

### **Syntax**

SelStartvalue = [objectreference].SelStart

[objectreference].SelStart= SelStartvalue

### **Legal values**

Do not set the SelStart property to a value less than 0. Attempting to do so results in a run-time error.

SelStart cannot be set greater than the number of characters selected in the edit portion of the combo box.

Attempting to do so results in a value equal to the number of characters length and sets the SelLength property to 0.

Getting or setting the SelStart property when the Style property value is set to 2 (instead of 0 or 1) results in a run-time error.

### **Usage**

Changes the current selection, if any, to an insertion point and sets the SelLength property to 0.

**Note** To access the entire contents of the edit portion, you can also use the Text property.

## **SelText Property**

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value indicating the selected characters in the edit portion of a combo box. A combo box only has an edit portion if its Style property is set to 0 (drop-down combo) or 1 (simple combo).

### **Data type**

String

### **Syntax**

SelTextvalue = [objectreference].SelText

[objectreference].SelText = SelTextvalue

### **Legal values**

Returns an empty string ("" ) if there is no selection in the edit portion of the combo box.

### **Usage**

When set, SelText replaces the current selection with the new string, sets the SelLength property's value to 0, and sets the SelStart property to point to the end of the new string. If nothing is selected, inserts text at SelStart.

Getting or setting the SelText property when the Style property value is set to 2 (instead of 0 or 1) results in a run-time error.

**Note** To access the entire contents of the edit portion, you can also use the Text property.

## SetItemData Method

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Sets the ItemData property for an indexed item.

### Syntax

[objectreference].SetItemData *index*, *ItemData*

### Parameters

*index*

Integer. The item in the combo box. The first item has 0 as the value for this argument. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

*ItemData*

Long. The number to set. The ItemData must be convertible to a Long or a run-time error results.

### Return values

None

## Sorted Property

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns a value determining whether the list in a combo box sorts entries in alphabetical order. The default value of this property is FALSE, indicating that the list does not sort entries in alphabetical order.

This property is read-only at runtime.

## Data type

[Integer](#)

## Syntax

Sortedvalue = [objectreference].Sorted

## Legal values

TRUE = The list is sorted in alphabetical order.

FALSE = (Default) The list is not sorted in alphabetical order.

## Usage

The Sorted property only affects items being added to the list. It does not resort the items already in the list. Note that default items are added to the list when the dialog is run, so they will be sorted if Sorted is set to TRUE.

If items are added to a sorted list at a particular position or using AddItem(index), or if items are replaced in a sorted list: future additions to the list may be sorted incorrectly.

To replace items and maintain proper sorting, use RemoveItem(index) and then use AddItem (leaving its index argument blank so the item gets added correctly).

## Style Property

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value determining the type of combo box. This property is read-only at runtime.

## Data type

Long

## Syntax

Stylevalue = [objectreference].Style

## Usage

Setting a value other than one of the following legal values results in a run-time error.

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Legal value</u>	<u>Constant</u>	<u>Type of combo box and behavior</u>
0	COMBOBOX_STYLE_DROPDOWN	Drop-down combo. Has edit portion and drop-down list (default).
1	COMBOBOX_STYLE_SIMPLE	Simple combo. Has edit portion and list that doesn't drop down. The Height property determines how much of the list gets displayed.
2	COMBOBOX_STYLE_DROPLIST	Drop-down list. Has no edit portion. Only allows selection from the list.

## Text Property

{button ,AL(^H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;!,0)} [See list of classes](#)

Returns or sets text in a combo box. The behavior of the Text property is dependent upon the setting of the Style property. See below for more information.

### Data type

String

### Syntax

Textvalue = [objectreference].Text

[objectreference].Text = Textvalue

### Usage

Depending on how the Style property is set, the Text property behaves in the following ways:

<u>Style Property</u>	<u>Text property behavior</u>
0 (drop-down combo)	Returns or sets the contents of the edit area.
1 (simple combo)	Returns or sets the contents of the edit area.
2 (drop-down list)	Returns the selected item in the list box. For drop-down lists, the Text property is Read-only.

## **Cancel Property**

{button ,AL(^H\_LDC\_LOTUSCOMMANDBUTTON\_LOTUSCOMMANDBUTTON\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a command button is a Cancel button. A value of TRUE indicates that the command button is a cancel button. ESC is defined as a mnemonic for the command button with a Cancel property set to TRUE.

## **Data type**

Integer

## **Syntax**

Cancelvalue = [objectreference].Cancel

[objectreference].Cancel = Cancelvalue

## **Legal values**

The legal values for this property are TRUE and FALSE.

## **Usage**

You can only have one cancel button per dialog. If you set the Cancel properties of more than one command button to TRUE, the dialog editor only picks one of them as the cancel button.



## **Default Property**

{button ,AL(^H\_LDC\_LOTUSCOMMANDBUTTON\_LOTUSCOMMANDBUTTON\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a command button is the default button. A value of TRUE indicates that the command button is a default button. ENTER is defined as a mnemonic for the command button with a Default property set to TRUE.

## **Data type**

Integer

## **Syntax**

Defaultvalue = [objectreference].Default

[objectreference].Default = Defaultvalue

## **Legal values**

The legal values for this property are TRUE and FALSE.

## **Usage**

You can only have one default button per dialog. If you set the Default properties of more than one command button to TRUE, the dialog editor only picks one of them as the default button.

## LotusCommandButton



You can use a command button to allow the user to do a wide variety of things, such as begin, end, or pause a process, close a dialog, etc. It is rare that you will build a dialog that doesn't contain at least one command button. By default, two command buttons appear in a dialog when you first create it. One of these buttons has its Default property set and the other has its Cancel property set. The Caption properties for these command buttons contains the text OK and Cancel, respectively. Neither of the default command buttons contain any script behind them.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

You can do a lot more than close a dialog with a command button. For example, the following script calls a global sub and then returns focus to the OK button.

```
Sub Click(Source As Lotuscommandbutton)
    HelloThere
    Source.Parent.Command1.Setfocus
End Sub
```

'Add this sub to Globals.

```
Sub HelloThere
    Print "Hello There!" 'Prints to the Output window in the IDE
End Sub
```

## LotusCommandButton members

### Properties

Appearance AS Integer  
BackColor AS Long [UNSUPPORTED]  
BorderStyle AS Integer  
Cancel AS Integer  
Caption AS String  
Default AS Integer  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

DoClick  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Click  
GotFocus  
LostFocus

## Appearance Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets a value indicating whether a control appears flat or in 3D.

## Data type

Integer

## Syntax

Appearancevalue = [objectreference].Appearance

[objectreference].Appearance = Appearancevalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CONTROL_APPEAR_FLAT	Flat
1	CONTROL_APPEAR_3D	3D (default)

All top level windows have a "chiseled" edge. 3D appearance corresponds to the button text/face color, while flat corresponds to window text/face color.

## BackColor Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets the background color of a control.

LotusCommandButton, LotusSpinButton and LotusProgressBar don't use this property. Although it is accessible in these controls, nothing happens when you try to set it.

## Data type

Long

## Syntax

BackColorvalue = [objectreference].BackColor

[objectreference].BackColor = BackColorvalue

## Legal values

A Long between 0 and 16,777,215.

The following table specifies the bit numbers for a color:

<u>Not used</u>	<u>R</u>	<u>G</u>	<u>B</u>
31 - 24	23 - 16	15 - 8	7 - 0

## Usage

The following table lists values for some typical colors.

<u>Value</u>	<u>Result</u>
0	Black
255	Red
32,768	Dark green
65,535	Yellow
8,421,504	Dark gray
12,632,256	Light gray
16,711,680	Blue
16,711,935	Magenta
16,777,215	White

## Caption Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets the text displayed by a control in a dialog.

## Data type

String

## Syntax

Captionvalue = [objectreference].Caption

[objectreference].Caption = Captionvalue

## Legal values

Limited to 256 characters.

## Usage

To add a mnemonic to a caption, type an & (ampersand) before the desired character that you want to display underlined. If you enter more than one & in the string, only the last one will display a character underlined. If you want an ampersand character to display in your string, enter &&, instead.

For a combo box, the Caption property is read-write if the Style property is 0 (drop-down combo) or 1 (simple combo). However, if the Style property is 2 (drop-down list), then the Caption property is read-only at run time.

For a list box the Caption property is read-only at run time.

**Note** LotusSpinButton, LotusProgressBar, LotusSlider and LotusImage have this property, but have no visual representation for it.

## Enabled Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)) See list of classes
```

Returns or sets a value determining whether a control can have focus and respond to keyboard and mouse input.

## Data Type

[Integer](#)

## Syntax

Enabledvalue = [objectreference].Enabled

[objectreference].Enabled = Enabledvalue

## Legal values

The legal values for this property are TRUE and FALSE.

## Usage

A value of TRUE indicates that the control can have focus and respond to keyboard and mouse input. A value of FALSE indicates that the control cannot have focus and respond to keyboard and mouse input.

## Font Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

This property contains a Font object. See [Font](#) for more information.

## Data type

[Variant](#)

## Legal values

Standard OLE Font object.

## Usage

You change the font attributes by changing the Font object's properties.



## ForeColor Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets the foreground color of a control.

LotusSlider, LotusCommandButton, LotusSpinButton and LotusProgressBar, and LotusImage don't use this property. Although it is accessible in these controls, nothing happens.

## Data type

Long

## Syntax

ForeColorvalue = [objectreference].ForeColor

[objectreference].ForeColor = ForeColorvalue

## Legal values

A Long between 0 and 16,777,215.

The following table specifies the bit numbers for a color:

<u>Not used</u>	<u>R</u>	<u>G</u>	<u>B</u>
31 - 24	23 - 16	15 - 8	7 - 0

## Usage

The following table lists values for some typical colors.

<u>Value</u>	<u>Result</u>
0	Black
255	Red
32,768	Dark Green
65,535	Yellow
8,421,504	Dark gray
12,632,256	Light gray
16,711,680	Blue
16,711,935	Magenta
16,777,215	White

## Height

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;'0)) See list of classes
```

Returns or sets a control's height in twips. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## Data type

Long

## Syntax

Heightvalue = [objectreference].Height

[objectreference].Height = Heightvalue

## Legal values

Setting this property to less than 15 twips returns an error.

## Usage

The following click event changes the height the image control "image1":

```
Sub Click(Source As Lotuscommandbutton)
    dialog1.image1.height = 300
End Sub
```

## HelpContextID Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets the Help context number for a control. Use this number to provide context-sensitive Help for a control.

## Data type

[Long](#)

## Syntax

HelpContextIDvalue = [objectreference].HelpContextID

[objectreference].HelpContextID = HelpContextIDvalue

## Usage

A value of 0 (default) indicates that no Help context number is associated with the control.

When the user accesses context sensitive Help, the topic with this context ID in the help file named in the [HelpFileName](#) property of the parent dialog is opened.

## **hWnd Property**

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

(Read-only)

Returns a control's hWnd (window handle).

## **Data type**

Long

## **Syntax**

hWndvalue = [objectreference].hWnd

## **Usage**

The hWnd can be used as an argument to functions from the Windows API that are accessed using c-callout. The hWnd is only valid while the dialog is shown, so be careful not to store it in a variable and attempt to use it after the dialog has been closed.

## **ID Property**

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

(Read-only)

Returns the unique ID generated for the control.

## **Data type**

Long

## **Syntax**

IDvalue = [objectreference].ID

## **Usage**

You can use the value of the ID property to identify a control in your scripts.

## Left Property

```
{button ,AL('H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)) See list of classes
```

Returns or sets the distance in twips between the left edge of a control and the left edge of its container. Usually the container is the dialog, but if the control is placed on a frame or similar control, then that control is the container.

The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## Data type

Long

## Syntax

Leftvalue = [objectreference].Left

[objectreference].Left = Leftvalue

## LotusControls Class

This collection class is used by LotusControl to represent the controls on a dialog.

The expression `Dialog1.Item(1)` returns the same value as `Dialog1.Controls(1)`.

The "index" for the Controls collection is generated by sorting the controls by control ID and then applying an ordinal to that sort. In other words, the sequence of controls is not defined; it may (and will) change.

### Base classes

Not applicable

### Contained by

LotusDialog in the Controls property

### Usage

The LotusControls class represents the collection of elements of type LotusControl on a dialog. This means when accessing the items by index, using LotusControls, you can only access the members of the LotusControl base class. The examples below should help to make this point more clear.

The following example hides all of the controls on a dialog:

```
Sub Load(Source As Lotusdialog)
    Forall c In Source.controls
        c.visible = False
    End Forall
End Sub
```

The following example DOES NOT set the value of all of the check boxes on the dialog to zero because the Value property is a member of LotusCheckBox—not LotusControl. Instead, this example returns the "Member not found" error:

'This sub returns an error:

```
Sub Load(Source As Lotusdialog)
    Forall c In Dialog1.controls
        If Typename(c) = "LOTUSCHECKBOX" Then
            c.value = 1
        End If
    End Forall
End Sub
```

The following example DOES set the value of all of the check boxes on the dialog to zero. This is possible because LotusScript supports downcasting:

```
Sub Load(Source As Lotusdialog)
    Forall control In Source.Controls
        If Typename( control ) = "LOTUSCHECKBOX" Then
            Dim chkbox As LotusCheckBox
            Set chkbox = control
            chkbox.visible = False
        End If
    End Forall
End Sub
```

## **LotusControls Members**

### **Properties**

None

### **Methods**

None

### **Events**

None



## **LotusControl**

All Lotus dialog controls derive from this class.

### **Base classes**

LotusBaseObject

### **Derived classes**

All Lotus dialog controls.

### **Contained by**

None

## LotusControl members

### Properties

Appearance AS Integer  
BackColor AS Long  
Caption AS String  
Description AS String  
Enabled AS Integer  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

Move  
Refresh  
SetFocus  
ZOrder

### Events

GotFocus  
LostFocus

## Move Method

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Moves and optionally resizes a control.

## Syntax

[objectreference].Move *left*, [*top*, *width*, *height*]

## Parameters

*left*

Long. The left-most horizontal coordinate of the upper left corner of the control in twips.

*top*

Long. The top-most vertical coordinate of the upper left corner of the control in twips.

*width*

Long. The width of the control in twips.

*height*

Long. The height of the control in twips.

## Return values

None

## Usage

The *top*, *width*, and *height* arguments are optional. However, if you use one of them, you must use all of the ones that precede it.

## Parent Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

(Read-only)

Returns the dialog containing the control.

## Data type

[LotusDialog](#)

## Usage

Use this property to reference the dialog that contains the control without using the dialog's name. This allows you to rename the dialog or copy the control and its script to another dialog without changing the script. For example, the following script for a command button's Click event closes the dialog that contains it.

```
Sub Click(Source As Lotuscommandbutton)
    Source.Parent.Close
End Sub
```

## Refresh Method

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Invalidates the control and repaints it, which may (or may not) actually refresh the data value – it's up to the specific control.

## Syntax

[objectreference].Refresh

## Parameters

None

## Return values

None

## SetFocus Method

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Gives focus to the control.

## Syntax

[objectreference].SetFocus

## Parameters

None

## Return values

None

## Usage

In order for a control to receive focus, you must set both its Enabled property and its Visible property to TRUE.

## TabIndex Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets the tab order for a control within a dialog.

All controls implement this property, but not all controls use it. LotusFrame, LotusImage, LotusLabel, LotusSpinButton and LotusProgressBar don't use this property. Although it is accessible in these controls, nothing happens when you get or set it.

## Data type

Integer

## Syntax

TabIndexvalue = [objectreference].TabIndex

[objectreference].TabIndex = TabIndexvalue

## Usage

Tab indexes start at 1 and are increased sequentially and assigned automatically as controls are added to a dialog.

Controls must have their TabStop property set to TRUE to participate in tabbing.

## TabStop Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;'0)) See list of classes
```

Returns or sets a value determining whether focus can be set to a control with the TAB key. By default, this property is set to TRUE.

All controls implement this property, but not all controls use it. LotusFrame, LotusImage, LotusLabel, LotusSpinButton and LotusProgressBar don't use this property. Although it is accessible in these controls, nothing happens when you get or set it.

### Data type

Integer

### Syntax

TabStopvalue = [objectreference].TabStop

[objectreference].TabStop = TabStopvalue

### Legal values

The legal values for this property are TRUE and FALSE.



## Tag Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets an arbitrary string which is stored by the control. You can safely set this property to any string you choose.

## Data type

String

## Syntax

Tagvalue = [objectreference].Tag

[objectreference].Tag = Tagvalue

## Top Property

```
{button ,AL('H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)) See list of classes
```

Returns or sets the distance in twips between the top edge of a control and the top edge of its container. Usually the container is the dialog, but if the control is placed on a frame or similar control, then that control is the container.

The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## Data type

Long

## Syntax

Topvalue = [objectreference].Top

[objectreference].Top = Topvalue

## Visible Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Returns or sets a value determining whether a control is displayed or hidden.

The default value of this property is TRUE (visible).

## Data type

[Integer](#)

## Syntax

Visiblevalue = [objectreference].Visible

[objectreference].Visible = Visiblevalue

## Legal values

The legal values for this property are TRUE and FALSE.

## Width Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)) See list of classes
```

Returns or sets the control's width in twips. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## Data type

Long

## Syntax

Widthvalue = [objectreference].Width

[objectreference].Width = Widthvalue

## Legal values

Setting this property to less than 15 twips returns an error.

## Usage

For example, the following click event changes the width for an image:

```
Sub Click(Source As Lotuscommandbutton)
    dialog1.image1.width = 300
End Sub
```

## ZOrder Method

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Positions a control at the front or back of the ZOrder. The ZOrder is the stacking order for controls and determines which control appears on top.

## Syntax

[objectreference].ZOrder [*position*]

## Parameters

*position*

Integer. A value of 0, or omitting the argument, brings the control to the top of the ZOrder. A value of 1 sends it to the back of the ZOrder.

## Return values

None

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CONTROL_ZORDER_BOTTOM	Brings control to top of ZOrder.
1	CONTROL_ZORDER_TOP	Sends control to back of ZOrder.

**Activate event**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Occurs when the dialog becomes the active window.

**Syntax**

```
Sub Activate(Source As Lotusdialog)
```

**Parameters**

*Source As Lotusdialog*

The dialog object that generated the event.

**Usage**

When the dialog becomes the active window, the script you add to the dialog's Activate event gets executed. For example, the following event makes a dialog's background appear red:

```
Sub Activate(Source As Lotusdialog)
    source.backcolor = 255
End Sub
```

**ActiveControl Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

The control with focus on the dialog.

**Data type**

[LotusControl](#)

**Syntax**

ActiveControlValue = [objectreference].ActiveControl

[objectreference].ActiveControl = ActiveControlValue

## **AlwaysOnTop Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Sets the dialog to always display on top or indicates whether the dialog always displays on top.

### **Data type**

Integer

### **Syntax**

AlwaysOnTopvalue = [objectreference].AlwaysOnTop

[objectreference].AlwaysOnTop = AlwaysOnTopvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

The default value for this property is FALSE, which keeps a dialog where it is. Setting this property to TRUE will display it on top.



## **Appearance Property**

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns or sets a value indicating whether the dialog appears flat or in 3D.

### **Data type**

Integer

### **Syntax**

Appearancevalue = [objectreference].Appearance

[objectreference].Appearance = Appearancevalue

### **Legal values**

0 = Flat

1 = 3D (default)

### **Usage**

Windows will always paint a top-level window with the "chiseled" edge; setting the Appearance property on a dialog simply uses a different background color.

**AutoRedraw Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;')} [See list of classes](#)

This property is reserved for future use.

## BackColor Property

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns or sets a value determining the background color of the dialog.

### Data type

Long

### Syntax

BackColorvalue = [objectreference].BackColor

[objectreference].BackColor = BackColorvalue

### Legal values

A Long between 0 and 16,777,215.

The following table specifies the bit numbers for a color:

<u>Not used</u>	<u>R</u>	<u>G</u>	<u>B</u>
31 - 24	23 - 16	15 - 8	7 - 0

### Usage

The following table lists values for some typical colors.

<u>Value</u>	<u>Result</u>
0	Black
255	Red
32,768	Dark green
65,535	Yellow
8,421,504	Dark gray
12,632,256	Light gray
16,711,680	Blue
16,711,935	Magenta
16,777,215	White

## BorderStyle Property

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns or sets the border style for the dialog.

### Data type

Integer

### Syntax

[objectreference].BorderStyle = BorderStylevalue

BorderStylevalue = [objectreference].BorderStyle

### Usage

The following table lists the available return values. If you prefer, you can substitute a constant for a value. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
0	DIALOG_BORDER_NONE	No border.
1	DIALOG_BORDER_FIXEDSING	Fixed single border.
2	DIALOG_BORDER_SIZEABLE	Sizeable border.
3	DIALOG_BORDER_FIXEDDLG	Fixed dialog border.
4	DIALOG_BORDER_FIXEDTOOL	Fixed tool border.
5	DIALOG_BORDER_SIZETOOL	Sizeable tool border.

**Caption Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Returns or sets the text displayed in the title of the dialog.

**Data type**

String

**Syntax**

Captionvalue = [objectreference].Caption

[objectreference].Caption = Captionvalue

**Legal values**

Up to 256 characters.

**Close Method**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;')} [See list of classes](#)

Closes the dialog.

**Syntax**

[objectreference].Close

**Return values**

None

**Parameters**

None

## Closing Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Occurs when the dialog is closing.

## Syntax

Sub Closing(Source As Lotusdialog, Mode As Integer, Cancel As Integer)

## Parameters

*Source As Lotusdialog*

The dialog that generated the event.

*Mode As Integer*

Indicates how the dialog was closed. If you prefer, you can substitute a constant for a value. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

Value	Constant	Description
0	CLOSING_CONTROL_MENU	The close button in the dialog's title bar was clicked.
1	CLOSING_DIALOG_CODE	The dialog was closed internally by the Close method.
2	CLOSING_WINDOWS	The application in which the dialog is running has closed the dialog.
3	CLOSING_TASK_MANAGER	Windows has been instructed to shut down the application running the dialog.

*Cancel As Integer*

This value is returned by the event. It determines whether or not to "really" close the dialog. The default value TRUE closes the dialog. Changing this value to FALSE keeps the dialog open.

## Usage

The following example demonstrates how you can test whether the user really wants to close a dialog:

```
%INCLUDE "lsconst.lss" ' This goes in Globals
' This is the Cancel button's Click handler:
Sub Click(Source As Lotuscommandbutton)
    Source.parent.Close ' Causes the Closing event to be fired.
End Sub
' This is Dialog1's Closing handler:
Sub Closing(Source As Lotusdialog, Mode As Integer, Cancel As Integer)
    Dim r as integer
    r = MsgBox("Do you really want to close the dialog?", MB_OKCANCEL)
    If r = IDOK Then
        Cancel = False
    Else
        Cancel = True ' the default
    End If
End Sub
```

## **ControlBox Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets a value indicating whether a control menu and control box are present in the dialog.

The control menu is the drop-down menu that appears when you click the icon in the top-left corner of the dialog.

The control box is the box located in the top-right corner of the dialog which closes the dialog and looks as follows:



### **Data type**

Integer

### **Syntax**

ControlBoxvalue = [objectreference].ControlBox

[objectreference].ControlBox = ControlBoxvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

A value of TRUE indicates that a control menu and control box are present. FALSE indicates neither is present.



## Controls Property

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

(Read-only)

Returns a fixed collection of the controls on the dialog.

## Data type

LotusControls

## Syntax

Controlsvalue = [objectreference].Controls(*index*)

## Legal values

Returns a collection of controls. Each is an instance of LotusControl.

## Usage

For more information about using this property see [LotusControls](#)

**Count Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

(Read-only)

Returns a value indicating the number of controls on the dialog.

**Data type**

[Integer](#)

**Syntax**

Countvalue = [objectreference].Count

## Deactivate Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Occurs when the dialog stops being the active window.

### Syntax

```
Sub Deactivate(Source As Lotusdialog)
```

### Parameters

*Source As Lotusdialog*

The dialog object that generated the event.

### Usage

When the dialog loses focus and is no-longer the active window, the script added to the dialog's Deactivate event gets executed. For example, the following Deactivate event makes a dialog's background appear green.

```
Sub Deactivate(Source As Lotusdialog)
    source.backcolor = 32768
End Sub
```

## **Displayed Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;')} [See list of classes](#)

(Read-only)

Returns a value indicating whether the dialog is currently displayed.

## **Data type**

[Integer](#)

## **Syntax**

Displayedvalue = [objectreference].Displayed

## **Return values**

TRUE = The dialog is currently displayed.

FALSE = The dialog is not currently displayed.

**Enabled Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Returns or sets a value determining whether the dialog can respond to events.

This property is ignored for modal dialogs.

**Data Type**

Integer

**Syntax**

Enabledvalue = [objectreference].Enabled

[objectreference].Enabled = Enabledvalue

**Legal values**

The legal values for this property are TRUE and FALSE.

**Usage**

A value of TRUE indicates that the control can respond to events. A value of FALSE indicates that the control cannot respond to events.

## ForeColor Property

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns or sets the foreground color of the dialog.

### Data type

Long

### Syntax

BackColorvalue = [objectreference].BackColor

[objectreference].BackColor = BackColorvalue

### Legal values

A Long between 0 and 16,777,215.

The following table specifies the bit numbers for a color:

<u>Not used</u>	<u>R</u>	<u>G</u>	<u>B</u>
31 - 24	23 - 16	15 - 8	7 - 0

### Usage

The following table lists values for some typical colors.

<u>Value</u>	<u>Result</u>
0	Black
255	Red
32,768	Dark Green
65,535	Yellow
8,421,504	Dark gray
12,632,256	Light gray
16,711,680	Blue
16,711,935	Magenta
16,777,215	White

## Height Property

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;';0)} [See list of classes](#)

Returns or sets a value that determines the height of the dialog in twips. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

### Data type

Long

### Syntax

Heightvalue = [objectreference].Height

[objectreference].Height = Heightvalue

## **HelpContextID Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets the Help context number for the dialog. Use this number to provide context-sensitive Help for the dialog.

### **Data type**

Long

### **Syntax**

HelpContextIDvalue = [objectreference].HelpContextID

[objectreference].HelpContextID = HelpContextIDvalue

### **Usage**

A value of 0 (default) indicates that no Help context number is associated with the dialog.

When the user accesses context sensitive Help, the topic with this context ID in the help file named in the [HelpFileName](#) property for the dialog is opened.



## **HelpFileName Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;')} [See list of classes](#)

The complete path to a Help file for the dialog and all the controls it contains.

### **Data type**

String

### **Syntax**

HelpFileNamevalue = [objectreference].HelpFileName

[objectreference].HelpFileNamevalue = HelpFileName

**Hide Method**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Makes the dialog invisible.

**Syntax**

[objectreference].Hide

**Parameters**

None

**Return values**

None

**Usage**

Using the Hide method on a modal dialog raises a run-time error.

**hWnd Property**

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

(Read-only)

Returns a dialog's hWnd (window handle).

**Data type**

Long

**Syntax**

hWndvalue = [objectreference].hWnd

**Usage**

The hWnd can be used as an argument to functions from the Windows API that are accessed using c-callout. The hWnd is only valid while the dialog is shown, so be careful not to store it in a variable and attempt to use it after the dialog has been closed.

## Item Method

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns an item in the LotusControls collection.

## Syntax

[objectreference].Item(*Index*)

## Parameters

*Index*

Integer. The position of the member in the collection.

## Return values

[LotusControl](#)

## Usage

At run time, when a command button containing the following Click event is clicked, the name of all of the controls in the dialog will print to the Output panel in the IDE.

```
Sub Click(Source As Lotuscommandbutton)
    'Source is the button, so to get to the dialog we follow its Parent property.
    Dim ThisDialog as LotusDialog
    Set ThisDialog = Source.Parent
    Dim i As Integer
    For i = 0 To (ThisDialog.Count - 1)
        Print ThisDialog.Item(i).Name
    Next
End Sub
```

## **KeyPreview Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets a value determining whether keyboard events for the dialog are invoked before keyboard events for the active control. The default setting for this property is FALSE.

### **Data type**

[Integer](#)

### **Syntax**

KeyPreviewvalue = [objectreference].KeyPreview

[objectreference].KeyPreview = KeyPreviewvalue

### **Legal values**

The legal values for this property are TRUE and FALSE

### **Usage**

Setting this property to FALSE (default) indicates that the active control gets the events and the dialog doesn't.

Setting this property to TRUE indicates that the dialog gets events first, and then the active control.

**Note** Setting this property one way or the other doesn't affect accelerator keys.

## **Left Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets a value in twips specifying the distance between the left edge of a dialog and the left edge of the screen. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## **Data type**

Long

## **Syntax**

Leftvalue = [objectreference].Left

[objectreference].Left = Leftvalue

## Load Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;',0)} [See list of classes](#)

Triggered when the dialog is loaded into memory.

### Syntax

```
Sub Load(Source As Lotusdialog)
```

### Parameters

*Source As Lotusdialog*

The dialog that generated the event.

### Usage

The script you add to the Load event executes when the dialog is loaded into memory. This is an excellent place to add script for initializing a dialog's state. For example, you can initialize values in any or all of its controls.

When you invoke a dialog using the Show method, the document's Initialize event is first triggered.

### Example 1

In the following example, assume you have a dialog that contains the text box "Text1". When the dialog is loaded into memory, the text box's Caption is set.

```
Sub Load(Source As Lotusdialog)
    Source.Text1.caption = "Hello there!"
End Sub
```

### Example 2

You'll find the Load event be a good place to set the value of global variables. When a dialog containing the following scripts is loaded, its position is read into two global variables. When the dialog unloads, if it has changed positions, a message box is displayed.

```
Dim y As Long, x As Long 'This goes in (Globals)
Sub Load(Source As Lotusdialog)
    y = Source.Top
    x = Source.Left
End Sub

Sub Unload(Source As Lotusdialog)
    If Source.Top <> y Or Source.Left <> x Then
        MsgBox "The dialog's position has changed."
    End If
End Sub
```

## LotusDialog

This class represents a single dialog box created by the dialog editor. A dialog is comprised of the dialog's window frame plus all of the controls placed on that window. See Usage, below, for more information.

### Base classes

[LotusBaseObject](#)

### Derived classes

None

### Contained by

[LotusCheckBox](#) in the [Parent](#) Property

[LotusComboBox](#) in the [Parent](#) Property

[LotusCommandButton](#) in the [Parent](#) Property

[LotusControl](#) in the [Parent](#) Property

[LotusDialog](#) in the [Parent](#) Property

[LotusFrame](#) in the [Parent](#) Property

[LotusImage](#) in the [Parent](#) Property

[LotusLabel](#) in the [Parent](#) Property

[LotusListBox](#) in the [Parent](#) Property

[LotusOptionButton](#) in the [Parent](#) Property

[LotusProgressBar](#) in the [Parent](#) Property

[LotusSlider](#) in the [Parent](#) Property

[LotusSpinButton](#) in the [Parent](#) Property

[LotusTextBox](#) in the [Parent](#) Property

### Usage

When you create a dialog, a global variable is implicitly defined for it. You can access the a dialog from host scripts through this global variable.

For example, clicking a button in a sheet in 1-2-3 that contains the following script displays the dialog whose Name property is set to "Dialog1":

```
'Script for a button in 1-2-3:
Sub Click(Source As Buttoncontrol)
    Dialog1.Show
End Sub
```

In addition to the set of properties listed in Help or the IDE browser, the LotusDialog class dynamically creates a property for each of its controls. To get properties, set properties, or call methods on a control on the dialog, you access the object by its name. For example, you create a dialog and drop a label onto it named "Label1". To change the Caption property for the label when the dialog loads, you make the dialog's Load event as follows:

```
Sub Load(Source As LotUSDIALOG)
    Source.Label1.Caption = "Hello there!"
End Sub
```

Suppose you want to change the label's caption from a host script. In this case use both the dialog name and the control name to access the control. For example, you add the following script to the Click event for a button on a sheet in 1-2-3. When the button is clicked, the dialog named "Dialog1" displays and the Caption for the label named "Label1" is set:

```
'Script for a button in 1-2-3:
Sub Click(Source As Buttoncontrol)
    Dialog1.Show
    Dialog1.Label1.Caption = "Hello there!"
End Sub
```





## Click Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Occurs when the dialog is clicked once (i.e. the pointer is positioned over the dialog and the default mouse button is lowered and raised once).

## Syntax

```
Sub Click(Source As Lotusdialog)
```

## Parameters

*Source As Lotusdialog*

The dialog that generated the event.

## Usage

When the dialog is clicked, the script you added to its Click event gets executed. For example, you might want to display a message box whenever the user clicks the surface of a dialog.

```
Sub Click(Source As Lotusdialog)
msgbox "...some instructions on using the dialog box here, perhaps?"
End Sub
```

## LotusDialog members

### Properties

ActiveControl AS LotusControl  
AlwaysOnTop AS Integer  
Appearance AS Integer  
AutoRedraw AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
ControlBox AS Integer  
Controls AS LotusControls [Read-only]  
Count AS Integer [Read-only]  
Description AS String  
Displayed AS Integer [Read-only]  
Enabled AS Integer  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
HelpFileName AS String  
hWnd AS Long [Read-only]  
KeyPreview AS Integer  
Left AS Long  
Modal AS Integer [Read-only]  
MousePointer AS Integer  
Name AS String  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

Close  
Hide  
Item  
Move  
Refresh  
SetFocus  
Show  
ZOrder

### Events

Activate  
Click  
Closing  
DbClick  
Deactivate  
GotFocus  
KeyDown  
KeyPress  
KeyUp  
Load  
LostFocus  
MouseDown  
MouseMove  
MouseUp  
Moved  
Unload



## **Modal Property**

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

(Read-only)

Returns a value of TRUE or FALSE indicating whether the dialog is modal or modeless.

## **Data type**

Integer

## **Syntax**

Modalvalue = [objectreference].Modal

## **Usage**

Modal dialogs require that the user complete them before doing anything else. You can make a dialog modal or modeless by using different arguments in the Show method. See the Show method for more information.

There are a variety of rules regarding modal and modeless dialogs:

### *Modal dialogs:*

- The show method does not return until the dialog is dismissed.

- Ignore the value of the Enabled and Visible properties.

- If closed from a script command, all other dialogs launched by that dialog close, as well.

### *Modeless dialogs:*

- Attempting to invoke a modal dialog from a modeless one raises an error.

## MousePointer Property

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Returns or sets a value indicating the mouse pointer (shape) to display when the pointer is over the dialog.

### Data type

Integer

### Syntax

MousePointervalue = [objectreference].MousePointer

[objectreference].MousePointer = MousePointervalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CONTROL_MP_DEFAULT	Shape determined by object (default)
1	CONTROL_MP_ARROW	Arrow
2	CONTROL_MP_CROSS	Cross-hair
3	CONTROL_MP_IBEAM	I-Beam
4	CONTROL_MP_ICON	[Reserved for future use.]
5	CONTROL_MP_SIZE	4-point arrow
6	CONTROL_MP_NESW	Northeast-Southwest
7	CONTROL_MP_NS	North-South
8	CONTROL_MP_NWSE	Northwest-Southeast
9	CONTROL_MP_WE	West-East
10	CONTROL_MP_UPARROW	Up-arrow
11	CONTROL_MP_HOURLASS	Hourglass
12	CONTROL_MP_NODROP	No-drop
13	CONTROL_MP_ARROWHG	Arrow and Hourglass
14	CONTROL_MP_ARROWQ	Arrow and question mark
15	CONTROL_MP_SIZEALL	Size all

## Moved Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;',0)} [See list of classes](#)

Triggered when the dialog moves to a new position or is resized.

## Syntax

```
Sub Moved(Source As Lotusdialog, X As Single, Y As Single)
```

## Parameters

*Source As Lotusdialog*

The dialog that generated the event.

*X As Single*

Single. Returns the X coordinate of the LotusDialog at the time the event is triggered.

*Y As Single*

Single. Returns the Y coordinate of the LotusDialog at the time the event is triggered.

## Usage

After the dialog has moved, whatever script you've added to its Moved event executes. For example, you create a dialog and you add a text box to it. You then make the dialog's Moved event as follows. When the dialog has moved, its new position is displayed in the text box.

```
Sub Moved(Source As Lotusdialog, X As Single, Y As Single)
    source.text1.caption = Cstr(X) + ", " + Cstr(Y)
End Sub
```

## Move Method

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;',0)} [See list of classes](#)

Moves the dialog to a new position on the screen and optionally resizes it.

## Syntax

[objectreference].Move *left, top, width, height*

## Parameters

*left*

Long. The left-most horizontal coordinate of the upper left corner of the dialog in twips.

*top*

Long. The top-most vertical coordinate of the upper left corner of the dialog in twips.

*width*

Long. Value indicating the (new) width of the dialog in twips.

*height*

Long. Value indicating the (new) height of the dialog in twips.

## Return values

None

## Usage

The top, width, and height arguments are optional. However, if you use one of them, you must use all of the ones that precede it.



**Refresh Method**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;')} [See list of classes](#)

This method is reserved for future use.

**SetFocus Method**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Gives focus to the dialog.

**Syntax**

[objectreference].SetFocus

**Parameters**

None

**Return values**

None

**Usage**

In order for a dialog to receive focus, you must set both the Enabled property and the Visible property to TRUE.

## Show Method

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Makes the dialog visible.

## Syntax

[objectreference].Show [*Style*]

## Parameters

*Style*

Integer. This optional parameter determines whether the dialog is modal or modeless.

The following table lists the available values for the *Style* argument. If you prefer, you can substitute a constant for a value. To use these constants, you must add the [constant file](#) `LSDCNST.LSS` to your script.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
0	DIALOG_MODELESS	Sets the Modal property to False (0).
1	DIALOG_MODAL	Sets the Modal property to True (-1).
2	DIALOG_DOCKABLE	Not available in this release. (Behaves as modeless in this release.)

## Return values

The return value for this method will always be TRUE or FALSE.

## Usage

*On modal dialogs:*

Does not return until the dialog has closed.

*On modeless dialogs:*

Returns immediately.

## **Tag Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;';0)} [See list of classes](#)

Returns or sets an arbitrary string which is stored by the dialog. You can safely set this property to any string you choose.

### **Data type**

String

### **Syntax**

Tagvalue = [objectreference].Tag

[objectreference].Tag = Tagvalue

## Top Property

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets a value in twips specifying the distance between the top edge of the dialog and the top edge of the screen. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## Data type

Long

## Syntax

Topvalue = [objectreference].Top

[objectreference].Top = Topvalue

## Unload Event

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;0)} [See list of classes](#)

Triggered when the dialog is about to unload from memory.

## Syntax

```
Sub Unload(Source As Lotusdialog)
```

## Parameters

*Source As Lotusdialog*

The dialog that generated the event.

## Usage

The script you add to the Unload event executes right before the dialog is removed from memory. When a dialog containing the following scripts is loaded, its position is read into two global variables. When the dialog unloads, if it has changed positions, a message box is displayed.

```
Dim y As Long, x As Long 'This goes in (Globals)
Sub Load(Source As Lotusdialog)
    y = Source.Top
    x = Source.Left
End Sub

Sub Unload(Source As Lotusdialog)
    If Source.Top <> y Or Source.Left <> x Then
        MsgBox "The dialog's position has changed."
    End If
End Sub
```

**Visible Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;'0)} [See list of classes](#)

Returns or sets a value determining whether the dialog is displayed or hidden.

The default value of this property is TRUE (visible).

**Data type**

Integer

**Syntax**

Visiblevalue = [objectreference].Visible

[objectreference].Visible = Visiblevalue

**Legal values**

The legal values for this property are TRUE or FALSE.

**Usage**

The Visible property is ignored for modal dialogs.

## **Width Property**

{button ,AL(^H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS;';0)} [See list of classes](#)

Returns or sets a value (in twips) that determines width of the dialog. The size of a twip is dependent upon your screen resolution. See product constants in the browser for twip to pixel conversion.

## **Data type**

Long

## **Syntax**

Widthvalue = [objectreference].Width

[objectreference].Width = Widthvalue



## ZOrder Method

{button ,AL('H\_LDC\_LOTUSDIALOG\_LOTUSDIALOG\_CLASS';0)} [See list of classes](#)

Positions the dialog at the front or back of the ZOrder. The ZOrder is the stacking order for dialogs and determines which dialog appears on top.

### Syntax

[objectreference].ZOrder [*position*]

### Parameters

*position*

Integer. A value of 0, or omitting the argument, brings the control to the top of the ZOrder. A value of 1 sends it to the back of the ZOrder.

### Return values

None

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CONTROL_ZORDER_BOTTOM	Brings dialog to top of ZOrder.
1	CONTROL_ZORDER_TOP	Sends dialog to back of ZOrder.

## **Bold Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS';,0)} [See list of classes](#)

Returns or sets a value determining whether text appears bold. The default value for this property is FALSE.

## **Data type**

[Integer](#)

## **Syntax**

Boldvalue = [objectreference].Bold

[objectreference].Bold = Boldvalue

## **Legal values**

The legal values for this property are TRUE and FALSE.

## **CharSet Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS';,0)} [See list of classes](#)

(Read-only)

Returns the character set of the font. This property is read-only; although if you set it you don't get an error (you just can't change it).

### **Data type**

[Integer](#)

### **Syntax**

CharSetvalue = [objectreference].CharSet

### **Legal values**

The value returned is the Windows identifier for the current character set. Refer to Windows documentation for more information.

## **Italic Property**

{button ,AL(^H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS;',0)} [See list of classes](#)

Sets or returns a value determining whether text appears in italics.

### **Data type**

Integer

### **Syntax**

Italicvalue = [objectreference].Italic

[objectreference].Italic = Italicvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

Setting this property to TRUE makes text appear in italics

## Font Class (Standard OLE)

This is the standard OLE Font class. In the IDE browser, this class appear as type Variant. See Usage, below, for an example of changing a font at run time.

### Base classes

Not applicable

### Derived classes

Not applicable

### Contained by

[LotusCheckBox](#) in the [Font](#) Property

[LotusComboBox](#) in the [Font](#) Property

[LotusCommandButton](#) in the [Font](#) Property

[LotusControl](#) in the [Font](#) Property

[LotusFrame](#) in the [Font](#) Property

[LotusLabel](#) in the [Font](#) Property

[LotusListBox](#) in the [Font](#) Property

[LotusOptionButton](#) in the [Font](#) Property

[LotusTextBox](#) in the [Font](#) Property

### Usage

You won't find this class with your products classes in the IDE browser. To view this class choose OLE Classes and select Standard OLE Types.

This class is used to access the font attributes of the containing control. Some methods are read-write and may be used to change font attributes, but some are read-only.

To make the text in the caption appearing with the LotusLabel "Label1" appear bold and change its font face to Courier.

You set the dialog's Load event as follows:

```
Sub Load(Source As Lotusdialog)
    Dialog1.Label1.Font.Bold = True
    Dialog1.Label1.Font.Name = "Courier"
End Sub
```

## Font members

### Properties

Bold AS Integer

CharSet AS Integer [Read-only]

Italic AS Integer

Name AS String

Size AS Currency

Strikethrough AS Integer

Underline AS Integer

Weight AS Integer

### Methods

None

### Events

None

**Name Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS';,0)} [See list of classes](#)

Returns or sets a value indicating the face name of a font.

**Data type**

String

**Syntax**

Namevalue = [objectreference].Name

[objectreference].Name = Namevalue

## Size Property

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS';,0)} [See list of classes](#)

Returns or sets a value indicating the point size of a Font.

## Data type

[Integer](#)

## Syntax

Sizevalue = [objectreference].Size

[objectreference].Size = Sizevalue

## Legal values

While you can set a TrueType font to virtually any size, some other fonts are only available in specific point sizes (4,6,9,10,16, 24 etc.). If you are setting a non-TrueType font's Size in code, you may find it helpful to first test to make sure the font can appear in the new size.



## **Strikethrough Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether text appears as strikethrough. The default value of this property is FALSE.

### **Data type**

[Integer](#)

### **Syntax**

Strikethroughvalue = [objectreference].Strikethrough

[objectreference].Strikethrough = Strikethroughvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

Setting this property to TRUE, means text appears strikethrough.

## **Underline Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether text appears underlined. The default value of this property is FALSE.

### **Data type**

[Integer](#)

### **Syntax**

Underlinevalue = [objectreference].Underline

[objectreference].Underline = Underlinevalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

Setting this property to TRUE, means text appears underlined.

## **Weight Property**

{button ,AL('H\_LDC\_LOTUSFONT\_LOTUSFONT\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining the weight of a font.

### **Data type**

Integer

### **Syntax**

Weightvalue = [objectreference].Weight

[objectreference].Weight = Weightvalue

### **Legal values**

A positive integer from 1 to 1000.

### **Usage**

The value of the Weight property affects the value of the Bold property as follows:

<u>Weight value</u>	<u>Effect</u>
Less than or equal to 550	Windows converts it to 400 and sets the Bold property to FALSE.
Greater than 550	Windows converts it to 700 and sets the Bold property to TRUE.

**ClipControls Property**

{button ,AL(^H\_LDC\_LOTUSFRAME\_LOTUSFRAME\_CLASS;')} [See list of classes](#)

This property is reserved for future use.

## LotusFrame



Use a frame to group controls. For example, to have two separate groups of option buttons, you first draw two frames and then draw the option buttons inside each frame.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

You cannot draw controls outside of a frame and then move them inside of the frame. For example, you cannot draw option buttons outside of a frame and then move it inside of the frame. Attempting the latter results in an option button residing on top of, but not in, the frame. Instead, draw each new option button inside of the frame in which you want it grouped.

## LotusFrame members

### Properties

Appearance AS Integer  
BackColor AS Long [UNSUPPORTED]  
BorderStyle AS Integer  
Caption AS String  
ClipControls AS Integer  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer [UNSUPPORTED]  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

Move  
Refresh  
SetFocus  
ZOrder

### Events

GotFocus  
LostFocus

## LoadImage Method

{button ,AL(^H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;!,0)} [See list of classes](#)

Loads an image file at runtime. Passing a Null filename will clear the image from the control.

### Syntax

[objectreference].LoadImage(*FileName*)

### Parameters

*FileName*

String. Path to the image file.

### Return values

Returns TRUE if the image was successfully loaded. Returns FALSE if the image was not successfully loaded.

### Usage

Use the LoadImage method to change an image at runtime.

For example, you create two option buttons and one image in a dialog and you want a different image to display, depending upon which button the user selects at run time.

You make the Click event for the first option button as follows:

```
Sub Click(Source As Lotusoptionbutton)
    Dialog1.Image1.LoadImage "C:\temp\foo1.bmp"
End Sub
```

You make the Click event for the second option button as follows:

```
Sub Click(Source As Lotusoptionbutton)
    Dialog1.Image1.LoadImage "C:\temp\foo2.bmp"
End Sub
```

## LotusImage



Use image to display a graphic in a dialog.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

To add or change a graphic in an image at run time, see [LoadImage](#).

To add a graphic to an image at design time, use the InfoBox.



## LotusImage members

### Properties

Appearance AS Integer  
BackColor AS Long [UNSUPPORTED]  
BorderStyle AS Integer  
Caption AS String [UNSUPPORTED]  
Description AS String  
Enabled AS Integer  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
Picture AS Variant  
Stretch AS Integer  
TabIndex AS Integer [UNSUPPORTED]  
TabStop AS Integer [UNSUPPORTED]  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

DoClick  
LoadImage  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Click  
DbClick  
GotFocus  
LostFocus  
MouseDown  
MouseMove  
MouseUp

**Picture Property**

{button ,AL(^H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;!,0)} [See list of classes](#)

Contains a standard OLE Picture object.

**Data type**

Picture

**Syntax**

Picturevalue = [objectreference].Picture

[objectreference].Picture = Picturevalue

## **Stretch Property**

{button ,AL(^H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;!,0)} [See list of classes](#)

Returns or sets a value determining whether an image control resizes to fit the image it contains or the image is stretched to fit the within the control.

### **Data type**

Integer

### **Syntax**

Stretchvalue = [objectreference].Stretch

[objectreference].Stretch = Stretchvalue

### **Legal values**

FALSE = (Default) The control resizes so that the whole image is displayed.

TRUE = The image resizes to fit the control.

## Alignment Property

{button ,AL('H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS';0)} [See list of classes](#)

Returns or sets a value determining the alignment of a label.

## Data type

Integer

## Syntax

Alignmentvalue = [objectreference].Alignment

[objectreference].Alignment = Alignmentvalue

## Usage

The following table lists the available values. If you prefer, you can substitute a constant for a value. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
0	LABEL_ALIGN_LEFT	Left aligned (default).
1	LABEL_ALIGN_RIGHT	Right aligned.
2	LABEL_ALIGN_CENTER	Center aligned.

## **AutoSize Property**

{button ,AL(^H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;'0)} [See list of classes](#)

Returns or sets a value determining whether a label resizes to display all of the text it contains.

### **Data type**

Integer

### **Syntax**

AutoSizevalue = [objectreference].AutoSize

[objectreference].AutoSize = AutoSizevalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

The default value of this property is FALSE, which truncates the label's contents so that only what fits in the is displayed.

## LotusLabel



Use the label control to display text that the user cannot (directly) change.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

You can enter the text for a label at design time. The dialog's user cannot modify this text at run time. However, you can modify the text at run time through script. For example, you modify a command button's click event as follows and when it is clicked the text in the label's caption changes:

```
Sub Click(Source As Lotuscommandbutton)
    Dialog1.Label1.Caption = "Hello world!"
```

```
End Sub
```

By default, only what fits within the predefined label dimension is displayed. Depending upon the length of your Caption, you'll want to set the AutoSize and WordWrap properties.

## LotusLabel members

### Properties

Alignment AS Integer  
Appearance AS Integer  
AutoSize AS Integer  
BackColor AS Long  
BackStyle AS Integer  
BorderStyle AS Integer  
Caption AS String  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
UseMnemonic AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long  
WordWrap AS Integer

### Methods

DoClick  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Click  
GotFocus  
LostFocus

## **UseMnemonic Property**

{button ,AL(^H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;'0)} [See list of classes](#)

Returns or sets a value determining whether the label's caption may contain a mnemonic (an access key).

### **Data type**

Integer

### **Syntax**

UseMnemonicvalue = [objectreference].UseMnemonic

[objectreference].UseMnemonic = UseMnemonicvalue

### **Legal values**

TRUE = Use Mnemonic.

FALSE = (Default) Don't use Mnemonic.

### **Usage**

You can specify the letter for the mnemonic in the Caption.

When a user presses the ALT+ letter of mnemonic, focus is given to the control with the next TabIndex.

Setting UseMnemonic to FALSE means that if you've entered an & character (an ampersand) in the Caption, it will be treated literally (and appear as an ampersand).



## **WordWrap Property**

{button ,AL(^H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;'0)} [See list of classes](#)

Returns or sets a value determining whether to expand and wrap text (the value of the Caption property) in a label.

### **Data type**

[Integer](#)

### **Syntax**

WordWrapvalue = [objectreference].WordWrap

[objectreference].WordWrap = WordWrapvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

TRUE = When the AutoSize property is also set to TRUE, wraps text and expands or contracts (vertically, not horizontally) the label.

FALSE = (Default) When the AutoSize property is also set to TRUE, does not wrap, but expands or contracts (vertically and horizontally) the label.

## **AddItem Method**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;')} [See list of classes](#)

Adds an item to a list box.

### **Syntax**

[objectreference].AddItem *text*, [*index*, *itemdata*]

### **Parameters**

*text*

String. Expression specifying the item to be added to the list box.

*index*

Long or Integer. Optional argument specifying the position in which to place the item in the list box. To position an item as the first item, use 0 as the value for this argument. Leaving the argument blank (or giving it a value of -1 or a value equal to the ListCount property) appends the item to the end of the list or sorts it if the Sorted property is set to TRUE.

*itemdata*

Long or Integer. Optional argument specifying the item's data. Leaving this argument blank passes a value of 0.

### **Return values**

None

### **Usage**

When a specific index is used (from 1 to ListCount - 1), the entry is not sorted even if the Sorted property is set to TRUE. Note that adding an item out of order to a sorted list may cause future entries to sort incorrectly.

**Clear Method**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;'0)} [See list of classes](#)

Clears the contents of the list box.

**Syntax**

[objectreference].Clear

**Parameters**

None

**Return values**

None

## Columns Property

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a list box scrolls horizontally or vertically and, if the list box scrolls horizontally, determines how many columns the list box has.

### Data type

Integer

### Syntax

Columnsvalue = [objectreference].Columns

[objectreference].Columns = Columnsvalue

### Usage

The default value of this property is 0 (list box scrolls vertically). A value of 1 or more indicates the number of horizontal snaking columns.

### Legal value Result

0        Scrolls vertically (default)

>= 1     Scrolls horizontally (value is the number of columns)

Setting the Columns property to a value less than 0 results in a run-time error.

## **IntegralHeight Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a list box displays partial items or resizes to display complete items. This property is read-only at runtime.

### **Data type**

Integer

### **Syntax**

IntegralHeightvalue = [objectreference].IntegralHeight

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

TRUE = List resizes only to display complete items (default).

FALSE = List doesn't resize. Items at bottom may be clipped.

### **IsItemSelected Method**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Returns a value indicating whether the item at index is selected in the list in the list box.

### **Syntax**

[objectreference].IsItemSelected(*index*)

### **Parameters**

*index*

Integer. The zero-based index of the item to check selection on.

### **Return values**

TRUE = The item is selected.

FALSE = The item is not selected.

### **Usage**

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

## ItemDataSelected Property

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Returns or sets the item data for the selected item a list box. Getting this property when there is no selected item returns an error. Setting this property when there is no selected item does nothing.

### Data type

Variant

### Syntax

ItemDataSelectedvalue = [objectreference].ItemDataSelected

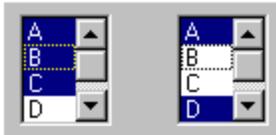
[objectreference].ItemDataSelected = ItemDataSelectedvalue

### Legal values

This property's value must be convertible to a Long or a run-time error is raised.

### Usage

If the MultiSelect property is set to TRUE, the ItemDataSelected property returns the item with focus. For example, for each of the following list boxes, the ItemDataSelected property returns "B".



## **ItemData Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;'0)} [See list of classes](#)

This parameterized property returns or sets a number associated with an item in the list in a list box. This property is runtime-checked and its data type must be a long or an Integer; otherwise an error is returned.

### **Data type**

Variant

### **Syntax**

ItemDatavalue = [objectreference].ItemData(index)

[objectreference].ItemData(index) = ItemDatavalue

### **Parameters**

*index*

Integer. Index of the item. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.



## **ListCount Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;'0)} [See list of classes](#)

(Read-only)

The number of items in the list in a list box.

## **Data type**

[Integer](#)

## **Syntax**

ListCountvalue = [objectreference].ListCount

## **ListIndex Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;'0)} [See list of classes](#)

Returns or sets the zero-based index of the selected item from a list box. Returns -1 if no item is selected.

### **Data type**

Integer

### **Syntax**

ListIndexvalue = [objectreference].ListIndex

[objectreference].ListIndex = ListIndexvalue

### **Usage**

When setting, if the index is less than 0 or greater than or equal to the ListCount property, a run-time error is raised.

If the MultiSelect property is set to 1 or 2, then ListIndex returns the index for the last item selected.

## List Property

{button ,AL('H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS','0)} [See list of classes](#)

This parameterized property is used to individually get and set the items of the list in a list box.

### Data type

String

### Syntax

Listvalue = [objectreference].List(index)

[objectreference].List(index) = Listvalue

### Parameters

*index*

Integer, which is the zero-based index for the item.

### Usage

*When setting*

If the index is greater than or equal to 0 and less than the value of the ListCount property, the element is replaced with the new string.

To add to the list box's list using the List property, use an index of -1 or an index equal to the ListCount property. For a sorted list box, this will add the item in its sorted position. For a non-sorted list box, it will add the item to the end of the list.

If the index is less than -1 or greater than the value of the ListCount property, a run-time error is raised.

*When getting:*

If the index is less than 0 or greater than or equal to the ListCount property, a run-time error is raised.

## LotusListBox



A list box allows the user to select an item from the items stored in a list. This list is accessible through the List property.

### **Base classes**

[LotusBaseObject](#)

[LotusControl](#)

### **Derived classes**

None

### **Contained by**

None

### **Usage**

Use the Style property to determine the type of list box.

## LotusListBox members

### Properties

Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
Columns AS Integer  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long  
IntegralHeight AS Integer  
ItemData AS Variant  
ItemDataSelected AS Variant  
Left AS Long  
List AS String  
ListCount AS Integer [Read-only]  
ListIndex AS Integer  
MousePointer AS Long  
MultiSelect AS Integer  
Name AS String  
NewIndex AS Integer [Read-only]  
Parent AS LotusDialog [Read-only]  
SelCount AS Integer [Read-only]  
Selected AS Integer  
Sorted AS Integer  
Style AS Integer  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Text AS String [Read-only]  
Top AS Long  
TopIndex AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

AddItem  
Clear  
DoClick  
IsItemSelected  
Move  
Refresh  
RemoveItem  
ReplaceItem  
SelectItem  
SelectItemString  
SetFocus  
SetItemData  
SetItemText  
ZOrder

### Events

Click  
DbClick  
GotFocus  
KeyDown  
KeyPress  
KeyUp  
LostFocus  
Pick

## MultiSelect Property

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a user can make multiple selections in a list box.

### Data type

Integer

### Syntax

MultiSelectvalue = [objectreference].MultiSelect

[objectreference].MultiSelect = MultiSelectvalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	LISTBOX_MSEL_NONE	Cannot make multiple selections.
1	LISTBOX_MSEL_SIMPLE	Can make simple multiple selections. Allows for non-contiguous selections. User can click or use arrows and SPACEBAR to select an item.
2	LISTBOX_MSEL_EXTEND	Can make extended multiple selections. Allows standard multiple selections and deselections (SHIFT + click, SHIFT + arrow keys, CTRL + click, CTRL + arrow keys).

Setting MultiSelect with a value other than a legal value results in a run-time error.

## **NewIndex Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;0)} [See list of classes](#)

(Read-only)

Returns a value indicating the index of the item last added to the list in a list box. Returns -1 if the list is empty or if the item was deleted.

## **Data type**

[Integer](#)

## **Syntax**

NewIndexvalue = [objectreference].NewIndex



## Pick Event

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Triggered when a specific item in the list in a list box is selected.

## Syntax

```
Sub Pick(Source As Lotuslistbox, Index As Integer)
```

## Parameters

*Source As Lotuslistbox*

The list box that generated the event.

*Index As Integer*

The zero-based number of a selected item in the list box.

## Usage

When you pick an item in a list box, whatever script has been added to the Pick event executes. For example, suppose you have a list box and a text box on a dialog. You make the list box's Pick event as follows. Whatever item is picked in the list box appears in the text box.

```
Sub Pick(Source As Lotuslistbox, Index As Integer)
    source.parent.text1.caption = source.list(index)
End Sub
```

## **RemoveItem Method**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Removes an item from a list box.

### **Syntax**

[objectreference].RemoveItem *index*

### **Parameters**

*index*

Integer. The zero-based index of the element to remove. To remove the first item, use 0 as the value for this argument.

### **Return values**

None

### **Usage**

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

## ReplaceItem Method

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;',0)} [See list of classes](#)

Replaces an item in a list box.

### Syntax

[objectreference].ReplaceItem *index*, *text* [,*itemdata*]

### Parameters

*index*

Long or Integer. Argument specifying the zero-based position of the item to be replaced in the list box. To replace the first item, use 0 as the value for this argument. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

*text*

String. Expression specifying the item to be used for replacement.

*itemdata*

Variant (must be Long or Integer or error is returned). Optional argument specifying the item's data. Leaving this argument blank passes a value of 0.

### Return values

None

### Usage

Replacing an item in a list when the Sorted property is set to TRUE may cause future entries to be sorted incorrectly. To replace an item and maintain proper sorting, delete the old item using RemoveItem, and add the new one using AddItem (leaving its index argument blank or using -1).

**SelCount Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

(Read-only)

Returns the number of items selected in a list box. Returns 0 if no items in the list box are selected.

**Data type**

[Integer](#)

**Syntax**

SelCountvalue = [objectreference].SelCount

## Selected Property

{button ,AL('H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS','0)} [See list of classes](#)

This parameterized property is used for getting or setting the selected state of an item in a list box.

### Data type

Integer

### Syntax

Selectedvalue = [objectreference].Selected(index)

[objectreference].Selected(index)= Selectedvalue

### Parameters

*index*

Integer. Specifies the item in the list box. The first item has 0 as the value for this argument. An index less than -1, or greater than or equal to the value of the ListCount property raises a run-time error.

### Legal values

TRUE = The item is selected.

FALSE = The item is not selected (default).

### Usage

*When getting the value of the Selected property:*

An index less than 0, or greater than or equal to ListCount, raises a run-time error.

*When setting the value of the Selected property:*

If the MultiSelect property is set to 0 (a single-select list-box), an index of -1 clears all selections, regardless of whether you set the Selected property to TRUE or FALSE.

If the MultiSelect property is set to 1 or 2 (a multi-select list-box), using an index of -1 and setting the Selected property to TRUE (-1) selects all items.

If the MultiSelect property is set to 1 or 2 (a multi-select list-box), using an index of -1 and setting the Selected property to FALSE clears all selections.

## SelectItemString Method

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Searches and then selects the next item that starts with a string. The search can begin from any item in the list.

### Syntax

[objectreference].SelectItemString *text*, *index*

### Parameters

*text*

String. The text of the item to select.

*index*

Integer. The item in the list box before the item from which to start. The first item has 0 as the value for this argument. The index must be greater than or equal to -1 and one less than the value of the ListCount property or a run-time error occurs. To start at the beginning, use an index of -1.

### Return values

Returns the index of the selected item. If no match is made, returns -1.

### Usage

When the search for the string reaches the end of the list, it wraps to the beginning. The search is case sensitive.

This method only selects a single item. In other words, even if the MultiSelect property is set to 1 or 2, the SelectItemString method only selects one item at a time (rather than extending a selection).

## SelectItem Method

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;')0)} [See list of classes](#)

Makes the index the currently selected item.

### Syntax

[objectreference].SelectItem *index* [, *IsSelected*]

### Parameters

*index*

Integer. The item in the list box. To select the first item use 0 as the value for this argument.

*IsSelected*

Variant. Optional argument which must be Long or Integer or an error is raised. Setting this argument to TRUE (the default) selects the indexed item. Setting this argument to FALSE deselects the indexed item.

### Return values

None

### Usage

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

If the MultiSelect property is set so that the list box can accept multiple selections, and IsSelected is TRUE, then all existing selected items are deselected before the new selection is made. If IsSelected is FALSE, then the item is deselected and the selection state of other items is not changed.

## **SetItemData Method**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Sets the ItemData property for an indexed item.

### **Syntax**

[objectreference].SetItemData *index*, *ItemData*

### **Parameters**

*index*

Integer. The item in the list box. The first item has 0 as the value for this argument. The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

*ItemData*

Long. The number to set. The ItemData must be convertible to a Long or a run-time error results.

### **Return values**

None



## Sorted Property

{button ,AL('H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS';0)} [See list of classes](#)

Returns a value determining whether the items inserted into the list in a list box appear sorted alphabetically. The default value of this property is FALSE, indicating that insertions are not sorted. This property is read-only at runtime.

## Data type

[Integer](#)

## Syntax

Sortedvalue = [objectreference].Sorted

## Legal values

TRUE = The list is sorted in alphabetical order.

FALSE = (Default) The list is not sorted in alphabetical order.

## Usage

The Sorted property only affects items being added to the list. It does not resort the items already in the list. Note that default items are added to the list when the dialog is run, so they will be sorted if Sorted is set to TRUE.

If items are added to a sorted list at a particular position or using AddItem(index), or if items are replaced in a sorted list: future additions to the list may be sorted incorrectly.

To replace items and maintain proper sorting, use RemoveItem(index) and then use AddItem (leaving its index argument blank so the item gets added correctly).

## Style Property

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;';0)} [See list of classes](#)

Returns or sets a value determining the scroll style for a list box.

## Data type

Integer

## Syntax

Stylevalue = [objectreference].Style

[objectreference].Style = Stylevalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Legal value</u>	<u>Constant</u>	<u>Type of list box and behavior</u>
0	LIST_STYLE_AUTOSCROLL	(Default) If there are more items in the list than the list box can display, it will display a scroll bar. Otherwise, it doesn't display the scroll bar.
1	LIST_STYLE_SCROLL	Always displays scroll bar.
2	LIST_STYLE_NEVER_SCROLL	Never displays scroll bar.

Setting a value other than one of the legal values results in a run-time error.

## **TopIndex Property**

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining which item currently appears as the topmost item in a list box. It doesn't change the order of the list; rather, it scrolls through the list box to the indexed item and puts the item at the top of the list.

### **Data type**

Integer

### **Syntax**

TopIndexvalue = [objectreference].TopIndex

[objectreference].TopIndex = TopIndexvalue

### **Legal values**

The index must be greater than or equal to 0 and less than the value of the ListCount property or a run-time error occurs.

### **Usage**

This property is useful for scrolling the list without making any selections.

The TopIndex property causes the item at index to appear as the top item on the list (provided there are enough items following the indexed item). If there are not enough items to fill the list box, then the list cannot scroll and the list index will not change.

*When setting the TopIndex property:*

For list boxes with columns greater than 0, the item at index is scrolled horizontally until it appears in the leftmost column. Its vertical position does not change.

*When getting the TopIndex property:*

For list boxes with columns greater than 0, the item at the top of the left column is returned.

## Alignment Property

{button ,AL(^H\_LDC\_LOTUSOPTIONBUTTON\_LOTUSOPTIONBUTTON\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining which side of the button the text appears on a option button.

## Data type

Integer

## Syntax

Alignmentvalue = [objectreference].Alignment

[objectreference].Alignment = Alignmentvalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	OPTBUTTON_ALIGN_LEFT	Left aligned (default)
1	OPTBUTTON_ALIGN_RIGHT	Right aligned

## LotusOptionButton



Use an option button when you want the user to choose a single selection from one or more options. The option buttons work in groups so that only one button in the group is selected at a time. When an option button is selected, any currently selected option in the group is deselected.

There is one option button group for the entire dialog. Each frame is its own option button group. So, you can have multiple option button groups on a dialog by using frames.

Once an option button is created in a group, it remains a part of that group no matter where you move it. If you create an option button on the dialog and then try to move it on a frame, it will still be part of the dialog option button group.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

Use option buttons when you want the user to make a single selection from among several choices. Use check boxes when you want the user to be able to make more than one selection.

## LotusOptionButton members

### Properties

Alignment AS Integer  
Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
Value AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

DoClick  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Click  
GotFocus  
LostFocus

**Handle Property**

{button ,AL(^H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS;',0)} [See list of classes](#)

A GDI handle to the image.

**Data type**

Long

**Syntax**

Handlevalue = [objectreference].Handle

[objectreference].Handle = Handlevalue

**Legal values**

A (Windows) GDI handle to the picture managed by this picture object.

## Height Property

{button ,AL(^H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS;',0)} [See list of classes](#)

(Read-only)

Returns a value in Himetrics that determines the height of a Picture.

## Data type

Long

## Syntax

Heightvalue = [objectreference].Height

## Usage

If you want to adjust the height of a image, use the [Height](#) property from LotusControl, instead.



**hPal Property**

{button ,AL(^H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS;',0)} [See list of classes](#)

Returns or sets a handle to the palette currently used by a picture.

**Data type**

Long

**Syntax**

hPalvalue = [objectreference].hPal

[objectreference].hPal = hPalvalue

## Picture Class (Standard OLE)

This is the standard OLE Picture class. In the IDE browser, this class appears as type Variant.

### Base classes

Not applicable

### Derived classes

Not applicable

### Contained by

LotusImage in the Picture Property

### Usage

You won't find this class with your products classes in the IDE browser. To view this class choose OLE Classes and select Standard OLE Types.

The following example consists of a button which uses the LotusImage method LoadImage to load a metafile into the image named "Image1". Afterwards, a message box displays the image's type—in this case 2 indicating a metafile. The type is located in the Type property of the picture class.

```
Sub Click(Source As Lotuscommandbutton)
    Call Dialog1.Image1.LoadImage("C:\foo.wmf")
    MsgBox(Dialog1.Image1.Picture.Type)
End Sub
```

## Picture members

### Properties

Handle AS Long

Height AS Long [Read-only]

hPal AS Long

Type AS Integer [Read-only]

Width AS Long [Read-only]

### Methods

Render

### Events

None

## Render Method

{button ,AL(^H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS;',0)} [See list of classes](#)

Renders (draws) the picture or part of the picture.

## Syntax

[objectreference].Render *hDC*, *x*, *y*, *cx*, *cy*, *xsrc*, *ysrc*, *cxsrc*, *cysrc*, *lprcwbounds*

## Parameters

*hDC*

Long. The device context on which to render.

*x*

Long. Horizontal coordinate to place the image (in *hDC*).

*y*

Long. Vertical coordinate to place the image (in *hDC*).

*cx*

Long. Length (horizontally) of the destination rectangle.

*cy*

Long. Height (vertically) of the destination rectangle.

*xsrc*

Long. Horizontal offset from which to start copying source.

*ysrc*

Long. Vertical offset from which to start copying source.

*cxsrc*

Long. Amount to copy horizontally from source.

*cysrc*

Long. Amount to copy vertically from source.

*lprcwbounds*

Variant. Points to the position of the destination for a metafile *hDC* (and in such cases, this parameter cannot be Null). This is necessary so you can place one metafile on another.

## Return values

None

## Type Property

{button ,AL(^H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS;',0)} [See list of classes](#)

(Read-only)

Returns a value indicating the graphic format of a Picture object.

## Data type

Integer

## Syntax

Typevalue = [objectreference].Type

## Legal values

The file you use with a LotusImage control may be of type .BMP, .WMF, or .ICO.

The following table lists the available values for the type property. If you get the Type for a .BMP file, it will return PICTURE\_TYPE\_METAFILE for both metafiles and bitmaps because LotusImage internally converts bitmap files into metafiles.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
2	PICTURE_TYPE_METAFILE	Metafile (.WMF file)
3	PICTURE_TYPE_ICON	Icon (.ICO file)

If you prefer, you can substitute a constant for a value. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

## **Width Property**

{button ,AL('H\_LDC\_LOTUSPICTUREOBJECT\_LOTUSPICTUREOBJECT\_CLASS','0)} [See list of classes](#)

(Read-only)

Returns a value in Himetrics that determines width of a Picture.

## **Data type**

Long

## **Syntax**

Widthvalue = [objectreference].Width

## **Usage**

If you want to set the width of an image, use the [Width](#) property from the base class, instead.

## LotusProgressBar



Use a progress bar to indicate the (approximate) progress of an operation.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

The properties Min and Max determine the limits of the range "monitored" by the progress bar. The Value property displays the current position within the range.

For example, a dialog contains a slider and a progress bar. When you change the slider the progress bar updates. Note that in this example none of the progress bar's events are triggered. Instead, when the slider changes positions the progress bar is updated by the slider's Change event.

'The dialog's Load event is as follows:

```
Sub Load(Source As Lotusdialog)
    Source.Slider1.Min = 1
    Source.Slider1.Max = 100
    Source.Slider1.Value = 1
    Source.Progress1.Min = 1
    Source.Progress1.Max = 100
    Source.Progress1.Value = 1
End Sub
```

'The slider's Change event is as follows:

```
Sub Change(Source As Lotusslider)
    Source.Parent.Progress1.Value = Source.value
End Sub
```

## LotusProgressBar members

### Properties

Appearance AS Integer  
BackColor AS Long [UNSUPPORTED]  
BorderStyle AS Integer  
Caption AS String [UNSUPPORTED]  
Description AS String  
Enabled AS Integer  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
Max AS Integer  
Min AS Integer  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
TabIndex AS Integer  
TabStop AS Integer [UNSUPPORTED]  
Tag AS String  
Top AS Long  
Value AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

Move  
Refresh  
SetFocus  
ZOrder

### Events

GotFocus  
LostFocus



**Max Property**

{button ,AL('H\_LDC\_LOTUSPROGRESSBAR\_LOTUSPROGRESSBAR\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining the maximum integer value that the Value property can hold for a progress bar.

**Data type**

Integer

**Syntax**

Maxvalue = [objectreference].Max

[objectreference].Max = Maxvalue

**Legal values**

The default value is 32,767.

## **Min Property**

{button ,AL('H\_LDC\_LOTUSPROGRESSBAR\_LOTUSPROGRESSBAR\_CLASS';,0)} [See list of classes](#)

Returns or sets a value determining the minimum integer value that the Value property can hold for a progress bar.

## **Data type**

Integer

## **Syntax**

Maxvalue = [objectreference].Max

[objectreference].Max = Maxvalue

## **Legal values**

The default value is 0.

## Change Event

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Triggered when a slider changes.

## Syntax

```
Sub Change(Source As Lotuslider)
```

## Parameters

*Source As Lotuslider*

The slider that generated the event.

## Usage

When you change the position of a slider, the script you added to the Change event executes. For example, the Caption property in a text box updates whenever the slider's position changes.

```
Sub Change(Source As Lotuslider)
    Source.parent.text1.caption = Cstr(source.value)
End Sub
```

## **ClearSel Method**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Clears a selected range in a slider.

### **Syntax**

[objectreference].ClearSel

### **Parameters**

None

### **Return values**

None

### **Usage**

Ignored if the SelectRange property is set to FALSE.

## **GetNumTicks Method**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns the number of ticks between the Min property and the Max property.

### **Syntax**

*TickValue* = [objectreference].GetNumTicks( )

### **Parameters**

None

### **Return values**

The number of ticks is an integer.

## **LargeChange Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the amount to change the Value property for a slider when the user clicks the slide area (the area between the slider and the edge of the control). The Value property is also changed by this amount when the user clicks the PAGEUP or PAGEDOWN keys.

### **Data type**

[Integer](#)

### **Syntax**

LargeChangevalue = [objectreference].LargeChange

[objectreference].LargeChange = LargeChangevalue

### **Legal values**

Legal values are from 0 through Max - Min.

## LotusSlider



A slider (control) contains a slider and, optionally, contains tick marks. The user can move the slider by dragging it or clicking either side of it to select a value or range.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

You'll find a slider useful when you want the user to specify a discrete value by moving a slider rather than by entering values.

#### Example 1

You want the value for whatever tick the user selects in a slider to appear in the text box "text1". You make the slider's Change event as follows:

```
'Update the text box:
Sub Change(Source As LotusSlider)
    Source.Parent.Text1.Caption = Cstr(source.value)
End Sub
```

#### Example 2

You want to allow users to be able to press SHIFT and select a range of ticks and you want the selected range to appear in the text box "text1". In this case, the range must be programmed:

```
'Dim the following global variables:
Dim SelectRange    'True if selecting
Dim Anchor         'The starting selection value

'Then you make the slider's events as follows:

Sub Mousedown(Source As LotusSlider, Button As Integer, Shift As Integer, X As Long, Y As Long)
    'Initialize to no selection, if no Shift, done:
    Source.ClearSel
    Let SelectRange = False
    If Shift = 0 Then Exit Sub
    'Start the selection:
    Let SelectRange = True
    Let Anchor = Source.Value
    Let Source.SelStart = Anchor
End Sub

Sub Mousemove(Source As LotusSlider, Button As Integer, Shift As Integer, X As Long, Y As Long)
    'If selecting and Shift was let up, cancel selection:
```

```

    If Not SelectRange Then Exit Sub
    If Shift Then Exit Sub
    Let SelectRange = False
    Source.ClearSel
End Sub

Sub Mouseup(Source As LotusSlider, Button As Integer, Shift As Integer, X As Long, Y
As Long)
    'Use MouseMove to check for Shift, then end selection:
    MouseMove Source, Button, Shift, X, Y
    Let SelectRange = False
End Sub

Sub Scroll(Source As LotusSlider)
    'If not selecting, done:
    If Not SelectRange Then Exit Sub
    'Draw the selection based on the anchor and current value:
    Dim Change As Integer
    Let Change = Source.Value - Anchor
    If Change >= 0 Then
        Let Source.SelStart = Anchor
        Let Source.SelLength = Change
    Else
        Let Source.SelStart = Source.Value
        Let Source.SelLength = -Change
    End If
End Sub

'Update the text box:
Sub Change(Source As LotusSlider)
    source.parent.text1.caption = Cstr(source.selLength)
End Sub

```



## LotusSlider members

### Properties

Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String [UNSUPPORTED]  
Description AS String  
Enabled AS Integer  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
LargeChange AS Integer  
Left AS Long  
Max AS Integer  
Min AS Integer  
MousePointer AS Long  
Name AS String  
Orientation AS Integer  
Parent AS LotusDialog [Read-only]  
SelectRange AS Integer  
SelLength AS Integer  
SelStart AS Integer  
SmallChange AS Integer  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
TickFrequency AS Integer  
TickStyle AS Integer  
Top AS Long  
Value AS Integer  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

ClearSel  
DoClick  
GetNumTicks  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Change  
Click  
GotFocus  
LostFocus  
MouseDown  
MouseMove  
MouseUp  
Scroll

**Max Property**

{button ,AL('H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the maximum integer value that the Value property can hold for a slider.

**Data type**

[Integer](#)

**Syntax**

Maxvalue = [objectreference].Max

[objectreference].Max = Maxvalue

**Legal values**

The default value is 10.

**Min Property**

{button ,AL('H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the minimum integer value that the Value property can hold for a slider.

**Data type**

[Integer](#)

**Syntax**

Maxvalue = [objectreference].Max

[objectreference].Max = Maxvalue

**Legal values**

The default value is 0.

## Orientation Property

{button ,AL('H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS';,0)} [See list of classes](#)

Returns or sets a value determining the orientation of a slider. The default value of this property is 0 (horizontal).

## Data type

Integer

## Syntax

Orientationvalue = [objectreference].Orientation

[objectreference].Orientation = Orientationvalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	SLIDER_ORIENT_HORIZ	Horizontal (default)
1	SLIDER_ORIENT_VERT	Vertical

## **Scroll Event**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Triggered when a user moves the slider in a slider and crosses a tick.

## **Syntax**

Sub Scroll(Source As Lotuslider)

## **Parameters**

*Source As Lotuslider*

The slider that generated the event.

## **Usage**

An isolated example of this event would be of little value. See the second example in [LotusSlider](#) for an example with some context.

## **SelectRange Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a slider can be used to select a range. The default value of this property is 0, indicating that the slider cannot be used to select a range.

### **Data type**

[Integer](#)

### **Syntax**

SelectRangevalue = [objectreference].SelectRange

[objectreference].SelectRange = SelectRangevalue

### **Legal values**

The legal values for this property are 1 and 0.

### **Usage**

1 = Can select a range using the slider.

0 = (Default) Cannot select a range using the slider.

## **SelLength Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the size of the selected range.

### **Data type**

Integer

### **Syntax**

SelLengthvalue = [objectreference].SelLength

[objectreference].SelLength = SelLengthvalue

### **Legal values**

Legal values are from 0 through Max - Min.

### **Usage**

This property is ignored when the SelectRange property is set to 0.

**SelStart Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the beginning of a selected range.

**Data type**

[Integer](#)

**Syntax**

SelLengthvalue = [objectreference].SelLength

[objectreference].SelLength = SelLengthvalue

**Usage**

This property is ignored when the SelectRange property is set to 0.



## **SmallChange Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining an amount to change the Value property for a slider when a user presses one of the arrow keys.

### **Data type**

Integer

### **Syntax**

SmallChangevalue = [objectreference].SmallChange

[objectreference].SmallChange = SmallChangevalue

### **Legal value**

Legal values are from 0 through Max - Min.

## **TickFrequency Property**

{button ,AL(^H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets the number of values each tick mark represents in a slider.

### **Data type**

Integer

### **Syntax**

TickFrequencyvalue = [objectreference].TickFrequency

[objectreference].TickFrequency = TickFrequencyvalue

## TickStyle Property

{button ,AL('H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining where ticks display for a slider. See below for settings.

### Data type

Integer

### Syntax

TickStylevalue = [objectreference].TickStyle

[objectreference].TickStyle = TickStylevalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	SLIDER_TICK_BOTTOMRIGHT	(default) Ticks on bottom if horizontal slider, or on right if vertical slider.
1	SLIDER_TICK_TOPLEFT	Ticks on top if horizontal slider, or on left if vertical slider.
2	SLIDER_TICK_BOTH	Ticks on both bottom/right and top/left.
3	SLIDER_TICK_NOTICKS	No ticks displayed.

## **Delay Property**

{button ,AL(^H\_LDC\_LOTUSSPINBUTTON\_LOTUSSPINBUTTON\_CLASS;';0)} [See list of classes](#)

Returns or sets a value determining the millisecond delay between SpinUp and SpinDown when a user holds down an ARROW key. The default value for this property is 250 milliseconds.

## **Data type**

[Integer](#)

## **Syntax**

Delayvalue = [objectreference].Delay

[objectreference].Delay = Delayvalue

## LotusSpinButton



Used with another control, such as a text box, to increment and decrement numbers or to scroll back and forth through a range of values.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

Use a spin button when you want the user to increment or decrement numbers or scroll through values in a collection.

For example, assume the global variable `foo` is declared as an integer. You can use the following events to increment or decrement the value of `foo`. Also, each time a change is made to `foo`, you update the contents of a text box:

```
Sub Spindown(Source As LotusSpinButton)
    foo = foo - 100
    Dialog1.Text1.Caption = Cstr(foo)
End Sub

Sub Spinup(Source As LotusSpinButton)
    foo = foo + 100
    Dialog1.Text1.Caption = Cstr(foo)
End Sub
```

## LotusSpinButton members

### Properties

Appearance AS Integer  
BackColor AS Long [UNSUPPORTED]  
BorderStyle AS Integer  
Caption AS String [UNSUPPORTED]  
Delay AS Integer  
Description AS String  
Enabled AS Integer  
ForeColor AS Long [UNSUPPORTED]  
Height AS Long  
HelpContextID AS Long  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
MousePointer AS Long  
Name AS String  
Parent AS LotusDialog [Read-only]  
SpinOrientation AS Integer  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

Move  
Refresh  
SetFocus  
ZOrder

### Events

GotFocus  
LostFocus  
SpinDown  
SpinUp

## **SpinDown Event**

{button ,AL(^H\_LDC\_LOTUSSPINBUTTON\_LOTUSSPINBUTTON\_CLASS;';0)} [See list of classes](#)

Triggered when a user clicks a spin button (down).

### **Syntax**

Sub Spindown(Source As LotusSpinbutton)

### **Parameters**

*Source As LotusSpinbutton*

The spin button that generated the event.

### **Usage**

See [LotusSpinButton](#) for an example of using this event.

## SpinOrientation Property

{button ,AL(^H\_LDC\_LOTUSSPINBUTTON\_LOTUSSPINBUTTON\_CLASS;';0)} [See list of classes](#)

Returns or sets a value determining whether the orientation of a spin button is vertical or horizontal. The default value of this property is 0, indicating a vertical spin button.

### Data type

[Integer](#)

### Syntax

SpinOrientationvalue = [objectreference].SpinOrientation

[objectreference].SpinOrientation = SpinOrientationvalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<b>Setting</b>	<b>Constant</b>	<b>Description</b>
0	SPIN_ORIENT_VERT	Vertical (default)
1	SPIN_ORIENT_HORIZ	Horizontal



## **SpinUp Event**

{button ,AL(^H\_LDC\_LOTUSSPINBUTTON\_LOTUSSPINBUTTON\_CLASS;';0)} [See list of classes](#)

Triggered when a user clicks a spin button (up).

### **Syntax**

```
Sub SpinUp(Source As LotusSpinbutton)
```

### **Parameters**

*Source As LotusSpinbutton*

The spin button that generated the event.

### **Usage**

See [LotusSpinButton](#) for an example of using this event.

## **Alignment Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining the alignment of a text box.

### **Data type**

Long

### **Syntax**

Alignmentvalue = [objectreference].Alignment

[objectreference].Alignment = Alignmentvalue

### **Legal values**

0 = Left aligned (default)

1 = Right aligned

### **Usage**

When you change the value of the Alignment property to 1, the MultiLine property must be set to TRUE. Alignment has no effect on a single-lined text box.

## Change Event

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Triggered when the contents of a text box changes. This event is triggered for every character that changes.

## Syntax

```
Sub Change(Source As Lotustextbox)
```

## Parameters

*Source As Lotustextbox*

The text box that generated the event.

## Usage

When you change the contents of a text box, the script added to its Change event gets executed. With the following script, when a user changes the contents of a check box, the global variable x is set to TRUE. Such information could be useful later, when the dialog box is about to close, for example.

```
Dim x As Integer 'Added to (Globals).  
Sub Change(Source As Lotustextbox)  
    x = True  
End Sub
```

## **GetLine Method**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns a specific line from a text box.

### **Syntax**

[objectreference].GetLine(*index*)

### **Parameters**

*index*

The zero based index of the line number you wish to access. To access the first line, use 0.

### **Return value**

Returns the line as a string.

### **Usage**

An index less than 0, or greater than or equal to LineCount, raises a run-time error.

## HideSelection Property

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether selected text appears highlighted. Under the default setting (TRUE), selected text does not appear highlighted in the text box.

### Data type

[Integer](#)

### Syntax

HideSelectionvalue = [objectreference].HideSelection

[objectreference].HideSelection = HideSelectionvalue

### Legal values

The legal values for this property are TRUE and FALSE.

### Usage

TRUE = Selected text doesn't appear highlighted.

FALSE = Selected text appears highlighted.

**LineCount Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

(Read-only)

Number of lines in a text box.

**Data type**

Long

**Syntax**

LineCountvalue = [objectreference].LineCount

## **Locked Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether the text in a text box can be edited. The default setting for this property is FALSE.

### **Data type**

Integer

### **Syntax**

Lockedvalue = [objectreference].Locked

[objectreference].Locked = Lockedvalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

TRUE = Text cannot be edited.

FALSE = Text can be edited (default).

## LotusTextBox



This control contains a text field in which text can be entered programatically or by the user.

### Base classes

[LotusBaseObject](#)

[LotusControl](#)

### Derived classes

None

### Contained by

None

### Usage

Set the MultiLine property to TRUE to display multiple lines of text. If MultiLine is set to TRUE, you can change the Alignment property from 0 (left aligned) to 1 (right aligned). Setting Alignment to 1 without setting the MultiLine property to TRUE has no effect.



## LotusTextBox members

### Properties

Alignment AS Long  
Appearance AS Integer  
BackColor AS Long  
BorderStyle AS Integer  
Caption AS String  
Description AS String  
Enabled AS Integer  
Font AS Variant  
ForeColor AS Long  
Height AS Long  
HelpContextID AS Long  
HideSelection AS Integer  
hWnd AS Long [Read-only]  
ID AS Long [Read-only]  
Left AS Long  
LineCount AS Long [Read-only]  
Locked AS Integer  
MaxLength AS Long  
MousePointer AS Long  
MultiLine AS Integer  
Name AS String  
Parent AS LotusDialog [Read-only]  
PasswordChar AS String  
ScrollBars AS Long  
SelLength AS Long  
SelStart AS Long  
SelText AS String  
TabIndex AS Integer  
TabStop AS Integer  
Tag AS String  
Text AS String  
Top AS Long  
Version AS Long [Read-only]  
Visible AS Integer  
Width AS Long

### Methods

DoClick  
GetLine  
Move  
Refresh  
SetFocus  
ZOrder

### Events

Change  
Click  
DbtClick  
GotFocus  
KeyDown  
KeyPress  
KeyUp  
LostFocus

## **MaxLength Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining the maximum number of characters and spaces in a text box.

For example, setting this property to 6 means that a maximum of 6 characters and spaces may be entered in the text box.

### **Data type**

[Integer](#)

### **Syntax**

Maxvalue = [objectreference].Max

[objectreference].Max = Maxvalue

### **Usage**

By default, this property is set to 0, meaning that there is no MaxLength and the text box can hold up to 32,767 characters and spaces.

## **MultiLine Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether a text box can contain multiple lines of text or only a single line.

### **Data type**

Integer

### **Syntax**

MultiLinevalue = [objectreference].MultiLine

[objectreference].MultiLine = MultiLinevalue

### **Legal values**

The legal values for this property are TRUE and FALSE.

### **Usage**

TRUE = (Default) Can contain multiple lines.

FALSE = Cannot contain multiple lines.

## **PasswordChar Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining whether what the user types or password characters, such as asterisks, appear in a text box. The default value of this property is an empty string (""), indicating that there is no password character and what the user types appears.

### **Data type**

String

### **Syntax**

PasswordCharvalue = [objectreference].PasswordChar

[objectreference].PasswordChar = PasswordCharvalue

### **Legal values**

Only the first character of the string is used. The rest of the string is ignored.

### **Usage**

The value of this property is always the first character in the string. In other words, if you set the property to "Xfoobar" the value "X" will appear for each character typed in the text box. Likewise, if you set the property to "foobarX", the value "f" will appear for each character typed in the text box.

Setting the PasswordChar property only affects single-line text boxes. If you set the MultiLine property to TRUE, the password character is ignored.

## ScrollBars Property

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value determining the type of scroll bars for a text box. The default value of this property is 0 (no scroll bars).

### Data type

[Integer](#)

### Syntax

ScrollBarsvalue = [objectreference].ScrollBars

[objectreference].ScrollBars = ScrollBarsvalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

<b>Setting</b>	<b>Constant</b>	<b>Description</b>
0	TEXTBOX_SCROLL_NONE	None (default)
1	TEXTBOX_SCROLL_HORIZ	Horizontal
2	TEXTBOX_SCROLL_VERT	Vertical
3	TEXTBOX_SCROLL_BOTH	Both

## **SelLength Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value indicating the number of characters selected.

The SelLength property is used in conjunction with the SelStart property to select a region of text.

### **Data type**

Long

### **Syntax**

SelLengthvalue = [objectreference].SelLength

[objectreference].SelLength = SelLengthvalue

### **Legal values**

Returns zero if nothing is selected.

Setting SelLength to less than -1 results in a run-time error.

### **Usage**

A SelLength value of -1 selects from the SelStart property's value to the end of the text box.

**SelStart Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value indicating the starting point of selected characters. If no characters are selected, indicates the insertion point.

The SelStart property is used in conjunction with the SelLength property to select a region of text.

**Data type**

[Integer](#)

**Syntax**

SelLengthvalue = [objectreference].SelLength

[objectreference].SelLength = SelLengthvalue

**Legal values**

Don't set this property to greater than the selected number of characters in the text box. Attempting to do so results in a value equal to the number of characters and sets the SelLength property to 0.

## **SelText Property**

{button ,AL(^H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets a value indicating the selected characters in the edit portion of a text box.

### **Data type**

String

### **Syntax**

SelTextvalue = [objectreference].SelText

[objectreference].SelText = SelTextvalue

### **Legal values**

Returns an empty string ("" ) if there is no selection.

### **Usage**

Setting SelText replaces the current selection with the new text, sets SelStart to the end of the new text and sets SelLength to zero. If there is no selection, setting SelText inserts the text at the insertion point.



## **Text Property**

{button ,AL('H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;',0)} [See list of classes](#)

Returns or sets contents of the entire edit area for a text box.

## **Data type**

String

## **Syntax**

Textvalue = [objectreference].Text

[objectreference].Text = Textvalue

## **Usage**

The default value for this property is the value of the Name property for the text box.

Use SelStart, SelLength, and SelText to get or set portions of the text in a text box.

## Dialog Controls Methods

A  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
O  
P  
Q  
R  
S  
T  
U  
V  
W  
X  
Y  
Z

### A

AddItem

### B

(None)

### C

Clear

ClearSel

Close

### D

DoClick

### E, F

(None)

### G

GetLine

GetNumTicks

### H

Hide

### I

IsItemSelected

Item

### J, K

(None)

### L

LoadImage

**M**

Move

**N, O, P, Q**

(None)

**R**

Refresh

RemoveItem

Render

ReplaceItem

**S**

SelectItem

SelectItemString

SelectItemText

SetFocus

SetItemData

SetItemText

Show

**T, U, V, W, X, Y**

(None)

**Z**

ZOrder

## Dialog Controls Properties

A  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
O  
P  
Q  
R  
S  
T  
U  
V  
W  
X  
Y  
Z

### A

[ActiveControl](#)  
[Alignment](#)  
[AlwaysOnTop](#)  
[Appearance](#)  
[AutoRedraw](#)  
[AutoSize](#)

### B

[BackColor](#)  
[BackStyle](#)  
[Bold](#)  
[BorderStyle](#)

### C

[Cancel](#)  
[Caption](#)  
[CharSet](#)  
[ClipControls](#)  
[Columns](#)  
[ControlBox](#)  
[Controls](#)  
[Count](#)

### D

[Default](#)  
[Delay](#)  
[Description](#)  
[Displayed](#)

### E

[Enabled](#)

**F**

Font  
ForeColor

**G**

(None)

**H**

Handle  
Height  
HelpContextID  
HelpFileName  
HideSelection  
hPal  
hWnd

**I**

ID  
IntegralHeight  
Italic  
ItemData  
ItemDataSelected

**J**

(None)

**K**

KeyPreview

**L**

LargeChange  
Left  
LineCount  
List  
ListCount  
ListIndex  
Locked

**M**

Max  
MaxLength  
Min  
Modal  
MousePointer  
MultiSelect

**N**

Name  
NewIndex

**O**

Orientation

**P**

Parent  
PasswordChar  
Picture

**Q**

(None)

**R**

(None)

**S**

ScrollBars

SelCount

Selected

SelectRange

SelLength

SelStart

SelText

Size

SmallChange

Sorted

SpinOrientation

Stretch

Strikethrough

Style

**T**

TabIndex

TabStop

Tag

Text

TickFrequency

TickStyle

Top

TopIndex

Type

**U**

Underline

UseMnemonic

Value

**V**

Version

Visible

**W**

Weight

Width

WordWrap

**X, Y, Z**

(None)

## Overview: Scalar Data Types

LotusScript recognizes the following scalar (numeric and string) data types:

Data type	Suffix	Value range	Size
Integer	%	-32,768 to 32,767 Initial value: 0	2 bytes
Long	&	-2,147,483,648 to 2,147,483,647 Initial value: 0	4 bytes
Single	!	-3.402823E+38 to 3.402823E+38 Initial value: 0	4 bytes
Double	#	-1.7976931348623158+308 to 1.7976931348623158+308 Initial value: 0	8 bytes
Currency	@	-922,337,203,685,477.5807 to 922,337,203,685,477.5807 Initial value: 0	8 bytes
String	\$	(String length ranges from 0 to 32K characters) Initial value: "" (empty string)	(2 bytes/character)

Besides these scalar data types, LotusScript supports the following additional data types and data structures:

Data type or structure	Description	Size
Array	An aggregate set of elements having the same data type. An array can comprise up to 8 dimensions whose subscript bounds can range from -32768 to 32767.  Initial value: Each element in a fixed array has an initial value appropriate to its data type.	Up to 64K bytes
List	A one-dimensional aggregate set whose elements have the same data type and are referred to by name rather than by subscript.	Up to 64K bytes
Variant	A special data type that can contain any scalar value, array, list, or object reference.  Initial value: EMPTY	16 bytes
User-defined data type	An aggregate set of elements of possibly disparate data types. Comparable to a record in Pascal or a struct in C.  Initial value: Member variables have initial values appropriate to their data type.	Up to 64K bytes
Class	An aggregate set of elements of possibly disparate data types together with procedures that operate on them.  Initial value: When you create an	

instance of a class, LotusScript initializes its member variables to values appropriate to their data types, and generates an object reference to it.

Object reference

A pointer to an OLE Automation object or an instance of a product class or user-defined class. 4 bytes

Initial value: NOTHING.

In each of the preceding tables, the specified storage size is platform-independent.



**BackStyle Property**

{button ,AL('H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS';0)} [See list of classes](#)

This property is reserved for future use.

## BorderStyle Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;'0)} See list of classes
```

Returns or sets the border style for a control.

### Data type

Integer

### Syntax

BorderStylevalue = [objectreference].BorderStyle

[objectreference].BorderStyle = BorderStylevalue

### Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

Setting	Constant	Description
0	CONTROL_BORDER_NONE	Control has no border.
1	CONTROL_BORDER_SINGLE	Control has border.

## Click Event

{button ,AL(^H\_LDC\_LOTUSCHECKBOX\_LOTUSCHECK\_CLASS;H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;H\_LDC\_LOTUSCOMMANDBUTTON\_LOTUSCOMMANDBUTTON\_CLASS;H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;H\_LDC\_LOTUSOPTIONBUTTON\_LOTUSOPTIONBUTTON\_CLASS;H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;');0)} [See list of classes](#)

Occurs when a control is clicked once (i.e. the pointer is positioned over the control and the default mouse button is lowered and raised once). For some controls, this event is also triggered when the control has focus and the user presses SPACEBAR or ENTER.

## Syntax

Sub Click(Source As [Data type])

## Parameters

*Source As [Data type]*

The control that generated the event.

## Usage

When the control is clicked, the script you added to its Click event gets executed. For example, the following script for a command button closes the dialog that contains it.

```
Sub Click(Source As Lotuscommandbutton)
    source.parent.close
End Sub
```

## **DbIclicK Event**

```
{button ,AL(^H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLISTBOX_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;','0)}  
See list of classes
```

Occurs when a control or dialog is double clicked (i.e. the pointer is positioned over the object and the default mouse button is lowered and raised twice).

## **Syntax**

```
Sub DbIclicK(Source As [Data type])
```

## **Parameters**

*Source As [Data type]*

The control or dialog that generated the event.

## **Usage**

When you double click the control or dialog, whatever script you've added to the DbIclicK event executes. For example, suppose you have an image control on a dialog. With the DbIclicK event as follows, when the user double clicks the image, the message box appears.

```
Sub DbIclicK(Source As LotusImage)  
    MsgBox ("...something about the image here.")  
End Sub
```

## DoClick Method

{button ,AL(^H\_LDC\_LOTUSCHECKBOX\_LOTUSCHECK\_CLASS;H\_LDC\_LOTUSCOMBOBOX\_LOTUSCOMBOBOX\_CLASS;H\_LDC\_LOTUSCOMMANDBUTTON\_LOTUSCOMMANDBUTTON\_CLASS;H\_LDC\_LOTUSIMAGE\_LOTUSIMAGE\_CLASS;H\_LDC\_LOTUSLABEL\_LOTUSLABEL\_CLASS;H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;H\_LDC\_LOTUSOPTIONBUTTON\_LOTUSOPTIONBUTTON\_CLASS;H\_LDC\_LOTUSSLIDER\_LOTUSSLIDER\_CLASS;H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;')0)} [See list of classes](#)

Triggers the Click event. A control has a DoClick method only if it has a Click event.

## Syntax

[objectreference].DoClick

## Parameters

None

## Return values

None

## Return values

None

## GotFocus Event

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)) See list of classes
```

Triggered when a control receives focus. You can give a control focus by tabbing to it or clicking it. You can also give a control focus using the SetFocus method.

In order for a control to receive focus, its Enabled property and its Visible property must be set to TRUE.

## Syntax

```
Sub Gotfocus(Source As [Data type])
```

## Parameters

*Source As [Data type]*

The control that generated the event.

## Usage

When a control gets focus, whatever script has been added to the GotFocus event executes. For example, you want a text box to appear yellow whenever it has focus.

```
Sub Gotfocus(Source As Lotustextbox)
    Source.backcolor = 65535 'Makes the control appear yellow.
End Sub
```

To change the text box's background to white when focus is lost, you modify the LostFocus event:

```
Sub Lostfocus(Source As Lotustextbox)
    Source.backcolor = 16777215 'Makes the control appear white.
End Sub
```

## KeyDown Event

```
{button ,AL(^H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;'
,0)} See list of classes
```

Triggered when a key is pressed (down) while an object has focus.

### Syntax

```
Sub Keydown(Source As [Data type], Keycode As Integer, Shift As Integer)
```

### Parameters

*Source As [Data type]*

The dialog or control that generated the event.

*Keycode As Integer*

The Windows virtual keycode corresponding to the pressed key.

*shift*

Integer. An value corresponding to the state of the SHIFT, CTRL and ALT keys at the time the event is triggered.

The SHIFT key has a value of 1, the CTRL key has a value of 2, and the SHIFT key has a value of 4. Some, or all, or none of the values can be set, and the total value indicates whether any, all, or none of the keys were pressed. For example, a value of 7 indicates that all keys were pressed. Likewise, a value of 3 indicates that SHIFT and CTRL were pressed.

### Usage

To find out a keycode for a key, add a text box to a dialog and make its KeyDown event as follows:

```
Sub Keydown(Source As Lotustextbox, Keycode As Integer, Shift As Integer)
    Print Keycode
End Sub
```

At run time, as you type keys in the text box their keycodes will appear in the Output panel of the IDE.

## **KeyPress Event**

```
{button ,AL(^H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;' ,0)} See list of classes
```

Triggered when an ANSI key has been pressed and released while an object has focus.

### **Syntax**

```
Sub Keypress(Source As [Data type], Keyascii As Integer)
```

### **Parameters**

*Source As [Data type]*

The dialog or control that generated the event.

*Keyascii As Integer*

A standard numeric ANSI keycode.

### **Usage**

To find out the standard numeric ANSI keycode for a key, add a text box to a dialog and make its KeyPress event as follows. At run time, as you enter text in the text box, the corresponding ANSI keycodes will appear in the Output panel of the IDE.

```
Sub Keypress(Source As Lotustextbox, Keyascii As Integer)
    Print Keyascii
End Sub
```



## KeyUp Event

{button ,AL(^H\_LDC\_LOTUSLISTBOX\_LOTUSLIST\_CLASS;H\_LDC\_LOTUSTEXTBOX\_LOTUSTEXTBOX\_CLASS;' ,0)} [See list of classes](#)

Triggered when pressed key is released while an object has focus.

### Syntax

Sub KeyUp(Source As [Data type], Keycode As Integer, Shift As Integer)

### Parameters

*Source As [Data type]*

The dialog or control that generated the event.

*Keycode As Integer*

The Windows virtual keycode corresponding to the pressed key.

*shift*

Integer. An value corresponding to the state of the SHIFT, CTRL and ALT keys at the time the event is triggered.

The SHIFT key has a value of 1, the CTRL key has a value of 2, and the SHIFT key has a value of 4. Some, or all, or none of the values can be set, and the total value indicates whether any, all, or none of the keys were pressed. For example, a value of 7 indicates that all keys were pressed. Likewise, a value of 3 indicates that SHIFT and CTRL were pressed.

### Usage

To learn how to find out the keycode for a key see the example in the KeyDown event topic.

## LostFocus Event

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSCONTROL_LOTUSCONTROL_CLASS;H_LDC_LOTUSFRAME_LOTUSFRAME_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLABEL_LOTUSLABEL_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;';0)} See list of classes
```

Triggered when a control loses focus. You can give a control focus by tabbing to it or clicking it. You can give another control focus in code, using the SetFocus method.

In order for a control to receive focus, its Enabled property and its Visible property must be set to TRUE.

## Syntax

```
Sub Lostfocus (Source As [Data type])
```

## Parameters

*Source As [Data type]*

The control that generated the event.

## Usage

When a control gets focus, whatever script has been added to the GotFocus event executes. For example, you want a text box to appear yellow whenever it gets focus:

```
Sub Gotfocus(Source As Lotustextbox)
    Source.backcolor = 65535 'Makes the control appear yellow.
End Sub
```

To change the text box's background to white when focus is lost, you modify the LostFocus event:

```
Sub Lostfocus(Source As Lotustextbox)
    Source.backcolor = 16777215 'Makes the control appear white.
End Sub
```

## MouseDown Event

```
{button ,AL('H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;',0)}
```

[See list of classes](#)

Triggered when a mouse button is pressed while an object has focus.

### Syntax

Sub Mousedown(Source As [Data type], Button As Integer, Shift As Integer, X As Long, Y As Long)

### Parameters

*Source As [Data type]*

The control or dialog that generated the event.

*Button As Integer*

An value indicating the mouse button that triggered the event.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
1	MOUSE_BUTTON_LEFT	The left button.
2	MOUSE_BUTTON_MIDDLE	The right button.
4	MOUSE_BUTTON_RIGHT	The middle button.

*Shift As Integer*

A value corresponding to the state of the SHIFT, CTRL and ALT keys at the time the event is triggered. The SHIFT key has a value of 1, the CTRL key has a value of 2, and the ALT key has a value of 4. Some, or all, or none of the values can be set, and the total value indicates whether any, all, or none of the keys were pressed. For example, a value of 7 indicates that all keys were pressed. Likewise, a value of 3 indicates that SHIFT and CTRL were pressed.

*X As Long*

Returns the x coordinate of the mouse pointer at the time the event is triggered (in twips)

*Y As Long*

Returns the y coordinate of the mouse pointer at the time the event is triggered (in twips).

### Usage

An isolated example of this event would be of little value. See the second example in [LotusSlider](#) for an example with some context.

## MouseMove Event

```
{button ,AL('H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;',0)}
```

[See list of classes](#)

Triggered when the mouse pointer is moved while an object has focus.

### Syntax

Sub Mousemove(Source As *[Data type]*, Button As Integer, Shift As Integer, X As Long, Y As Long)

### Parameters

*Source As [Data type]*

The control or dialog that generated the event.

*Button As Integer*

A value indicating whether any mouse buttons were pressed when the event was triggered. These values let you tell whether a combination of buttons has been clicked. For example a value of 3 indicates that the both the left button and right button were pressed.

Value	Constant	Description
1	MOUSE_BUTTON_LEFT	The left button.
2	MOUSE_BUTTON_MIDDLE	The right button.
4	MOUSE_BUTTON_RIGHT	The middle button.

*Shift As Integer*

A value corresponding to the state of the SHIFT, CTRL and ALT keys at the time the event is triggered. The SHIFT key has a value of 1, the CTRL key has a value of 2, and the ALT key has a value of 4. Some, or all, or none of the values can be set, and the total value indicates whether any, all, or none of the keys were pressed. For example, a value of 7 indicates that all keys were pressed. Likewise, a value of 3 indicates that SHIFT and CTRL were pressed.

*X As Long*

Returns the x coordinate of the mouse pointer at the time the event is triggered (in twips).

*Y As Long*

Returns the y coordinate of the mouse pointer at the time the event is triggered (in twips).

### Usage

An isolated example of this event would be of little value. See the second example in [LotusSlider](#) for an example with some context.

## MousePointer Property

```
{button ,AL(^H_LDC_LOTUSCHECKBOX_LOTUSCHECK_CLASS;H_LDC_LOTUSCOMBOBOX_LOTUSCOMBOBOX_CLASS;H_LDC_LOTUSCOMMANDBUTTON_LOTUSCOMMANDBUTTON_CLASS;H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSLISTBOX_LOTUSLIST_CLASS;H_LDC_LOTUSOPTIONBUTTON_LOTUSOPTIONBUTTON_CLASS;H_LDC_LOTUSPROGRESSBAR_LOTUSPROGRESSBAR_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;H_LDC_LOTUSSPINBUTTON_LOTUSSPINBUTTON_CLASS;H_LDC_LOTUSTEXTBOX_LOTUSTEXTBOX_CLASS;','0)} See list of classes
```

Returns or sets a value indicating the mouse pointer (shape) to display when the pointer is over the control.

## Data type

Long

## Syntax

MousePointervalue = [objectreference].MousePointer

[objectreference].MousePointer = MousePointervalue

## Usage

The following table lists the available settings. If you prefer, you can substitute a constant for a setting. To use these constants, you must add the [constant file](#) LSDCNST.LSS to your script.

Setting	Constant	Description
0	CONTROL_MP_DEFAULT	Shape determined by object (default)
1	CONTROL_MP_ARROW	Arrow
2	CONTROL_MP_CROSS	Cross-hair
3	CONTROL_MP_IBEAM	I-Beam
4	CONTROL_MP_ICON	[Reserved for future use.]
5	CONTROL_MP_SIZE	4-point arrow
6	CONTROL_MP_NESW	Northeast-Southwest
7	CONTROL_MP_NS	North-South
8	CONTROL_MP_NWSE	Northwest-Southeast
9	CONTROL_MP_WE	West-East
10	CONTROL_MP_UPARROW	Up-arrow
11	CONTROL_MP_HOURLASS	Hourglass
12	CONTROL_MP_NODROP	No-drop
13	CONTROL_MP_ARROWHG	Arrow and Hourglass
14	CONTROL_MP_ARROWQ	Arrow and question mark
15	CONTROL_MP_SIZEALL	Size all

## MouseUp Event

```
{button ,AL('H_LDC_LOTUSIMAGE_LOTUSIMAGE_CLASS;H_LDC_LOTUSSLIDER_LOTUSSLIDER_CLASS;',0)}
```

[See list of classes](#)

Triggered when a pressed mouse button is released while an object has focus.

### Syntax

Sub Mouseup(Source As [Data type], Button As Integer, Shift As Integer, X As Long, Y As Long)

### Parameters

*Source As [Data type]*

The control or dialog that generated the event.

*Button As Integer*

Integer. An value indicating whether any mouse buttons were released, thus triggering the event.

<u>Value</u>	<u>Constant</u>	<u>Description</u>
1	MOUSE_BUTTON_LEFT	The left button.
2	MOUSE_BUTTON_MIDDLE	The right button.
4	MOUSE_BUTTON_RIGHT	The middle button.

*Shift As Integer*

A value corresponding to the state of the SHIFT, CTRL and ALT keys at the time the event is triggered. The SHIFT key has a value of 1, the CTRL key has a value of 2, and the ALT key has a value of 4. Some, or all, or none of the values can be set, and the total value indicates whether any, all, or none of the keys were pressed. For example, a value of 7 indicates that all keys were pressed. Likewise, a value of 3 indicates that SHIFT and CTRL were pressed.

*X As Long*

Returns the x coordinate of the mouse pointer at the time the event is triggered (in twips).

*Y As Long*

Returns the y coordinate of the mouse pointer at the time the event is triggered (in twips).

### Usage

An isolated example of this event would be of little value. See the second example in [LotusSlider](#) for an example with some context.

## Value Property

{button ,AL(^H\_LDC\_LOTUSCHECKBOX\_LOTUSCHECK\_CLASS;H\_LDC\_LOTUSOPTIONBUTTON\_LOTUSOPTIO  
NBUTTON\_CLASS;H\_LDC\_LOTUSPROGRESSBAR\_LOTUSPROGRESSBAR\_CLASS;H\_LDC\_LOTUSSLIDER  
\_LOTUSSLIDER\_CLASS;','0)} [See list of classes](#)

Returns or sets a value determining the state or position for a control. The default value of this property is 0.

## Data type

[Variant](#) for LotusOptionButton

[Integer](#) for other controls

## Syntax

Valuevalue = [objectreference].Value

[objectreference].Value = Valuevalue

## Legal values

These depend upon which control you're using.

For *LotusCheckBox* use the following settings, or add the [constant file](#) LSDCNST.LSS to your script and use constants, instead.

<u>Setting</u>	<u>Constant</u>	<u>Description</u>
0	CHECKBOX_VAL_UNCHECKED	Unchecked (default)
1	CHECKBOX_VAL_CHECKED	Checked
2	CHECKBOX_VAL_GREYED	Dimmed

For *LotusOptionButton*

FALSE = Unclicked (default)

TRUE = Clicked

For *LotusCheckBox*

FALSE = Unchecked (default)

TRUE = Checked

2 = Greyed (but not disabled)

For *LotusProgressBar*

Position of the scroll bar (must fall between the Max and Min properties).

For *LotusSlider*

Position of the scroll bar (must fall between the Max and Min properties).

**UNSUPPORTED**

The property is not supported for this control.

Because this control inherits from the base class LotusControl, it contains all of the properties from LotusControl. However, the control doesn't use some of the LotusControl properties.

For example, because LotusImage inherits from LotusControl, it contains all of the properties in LotusControl, including the Caption property. However, there is no means of displaying a caption on an image. Hence, you can set the Caption property of a LotusImage, but its contents will never display at runtime.



