See Also

Chapter III-14, **Controls and Control Panels**, for details about control panels and controls.

Control Panel Units on page III-444 for a discussion of the units used for controls.

Related functions **ModifyControlList** and **ControlNameList**.

The **Button**, **Chart**, **CheckBox**, **GroupBox**, **ListBox**, **PopupMenu**, **SetVariable**, **Slider**, **TabControl**, **TitleBox**, and **ValDisplay** controls.

ModifyControlList

ModifyControlList [**/Z**] ***listStr*** [**, keyword =** ***value***]…

The ModifyControlList operation modifies the controls named in the *listStr* string expression. ModifyControlList works on any kind of existing control.

Parameters

listStr is a semicolon-separated list of names in a string expression. The expression can be an explicit list of control names such as "button0;checkbox1;" or it can be any string expression such as a call to the ControlNameList string function:

```
ModifyControlList ControlNameList("",";","*_tab0") disable=1
```

The controls must exist.

Keywords

The following keyword=value parameters are supported:

activate	align	appearance	bodywidth	disable	fColor
focusRing	font	fSize	fStyle	help	labelBack
noproc	pos	proc	rename	size	title
userdata	valueBackColor	valueColor	win		

Coordinates are in **Control Panel Units**.

For details on these keywords, see the documentation for **SetVariable** on page V-854.

The following keywords are not supported:

mod	popmatch	popvalue	value	variable

Flags

/Z	No error reporting.

Details

Use ModifyControlList to move, hide, disable, or change the appearance of multiple controls without regard to their kind.

If *listStr* contains the name of a nonexistent control, an error is generated.

if *listStr* is `""` (or any list element in *listStr* is `""`), it is ignored and no error is generated.

Example

Here is the **TabControl** procedure example from **ModifyControl** rewritten to use ModifyControlList. It shows and hides all controls in the tabs appropriately, without knowing what kind of controls they are, but the code is simpler. This method does not, however, preserve the enable bit when a control is hidden.

The “trick” here is that all controls that are to be shown within particular tab *n* have been assigned names that end with “_tab*n*” such as “_tab0” and “_tab1”:

```
// Action procedure
Function TabProc2(ctrlName,tabNum) : TabControl
    String ctrlName
    Variable tabNum

    String controlsInATab= ControlNameList("",";","*_tab*")
```