Button Custom Controls

SRC Enterprises

Steven Clabaugh
550 N. Pantano Road #286
Tucson, Arizona 85710

(520) 296-7570
Voice or FAX

Button Concepts

While Windows provides standard button controls, programming these standard buttons can be somewhat tedious at times, especially when dealing with bitmap buttons, thus this Custom Button Control was created to simplify button control programming and add additional button styles and features.

Several button types and features are available for button controls and are set at Button creation time and some features can be controlled on the fly in an application program.

- User defined bitmap Momentary Button.
- Predefined bitmap Spin Button.
- User defined bitmap Spin Button.
- Value wrapping feature for Spin Buttons.
- Programmable spin speed for Spin Buttons.
- Color Button.
- Push style 2-State Button.
- Toggle style 2-State Button.
- Vertical/Horizontal style for Spin Buttons and Toggle Buttons.
- Custom Buttons can be added as a custom control to the Windows SDK Dialog Editor.
- C/C++ language API programming support, including a C++ Button Class.
- Support for Visual Basic Extention (VBX).

Button Types and Features

Momentary Push Buttons are virtually the same as standard bitmap button controls.

A Spin Button is a dual button control used for numeric value control. Numeric values can be either incremented or decremented with a Spin Button. Spin Buttons have a value wrap feature which can be turned ON or OFF. With the wrap feature turned ON, the value will wrap around from the range maximum value to the range minimum value and from the range minimum value to the range maximun value.

Color Buttons are a solid color button used in applications to allow easy selection of colors without invoking a color selction dialog. Only a limited number of colors are supported for color buttons, ie: Red, Green, Blue, Yellow, Cyan, Magenta, Gray, and Black.

Two-State Buttons are two position selection switches implemented visually as either a single push button switch or a toggle type switch. These button type are designed as an alternative to the standard CHECKBOX control.

The Horizontal/Vertical feature allows Spin Buttons and Toggle type buttons to be operated vertically or horizontally.

Button Message

All Button controls send a `WM_COMMAND` message to its parent window message processing function. The ***wParam*** parameter contains the ID value of the Button control. The ***lParam*** low-order word contains the Button window handle. The ***lParam*** high-order word is not used and contains a value of 0(Zero).

Button Files

The required files for incorporating Button controls in an application are:

SRCENTBN.DLL — This file contains the window function for the Button control along with the API functions and the interface to the SDK Dialog Editor and the Visual Basic Extention (VBX) support

To use the Button Custom Control as a Visual Basic Extenton(VBX), you need only to copy or rename this file to SRCENTBN.VBX to use the Button in the Visual Basic and Visual C++ MFC environments.

SRCENTBN.H — This file should be included in the application C(PP) source code file. It contains the definitions for the Button control style and feature words along with the API function definitions and the delclaration for the Button class for C++ programming.

SRCENTBN.LIB — Library of import functions created with IMPLIB for Button controls. Specify this library in the LINK command's *lib-files* parameter when linking the application program.

C++ Button Class

For C++ programmers, the SRCENTBN.DLL file also contains implementation for a C++ class for Button controls. The Button class declaration is in the SCRENTBN.H file. Since the class implememtation is simply an encapsulation of the Button API functions, all parameters and return values for the Button class member functions are the same as the API functions, but with the omission of the API's first parameter, which is the Button window handle parameter (`HWND` *`hWnd`*). The Button class also includes a member function to retreive the Button window handle.

Creating Button Controls with C/C++

Button controls are child windows. They are initially created by one of two ways: Using the **SRCEntBNCreate()** function in an application C(PP) file or with the dialog CONTROL statement in an application's resource script file.

When using the CONTROL statement in a resource file, keep in mind the location and size specified are in dialog box base units, ie: relative to the size of the font in use for the dialog box. This does not enable the the Button control to be located and sized with pixel precision. If pixel precision location and size are required in a dialog box you will have to use the **SRCEntBNCreate()** function in the WM_INITDIALOG message for the dialog.

An example of the CONTROL statement in a resource script file looks like this:

```
CONTROL "", ID_MYBUTTON,"SRCEnt_Button",
        BTN_SPIN6 | WS_CHILD, 10,10,30,25
```

This specifies a horizontal Spin Button with no value wrapping, using the predefined 25 x 19 pixel bitmap. Note the width and height of the Button will be stretched to a width and height of 30 x 25 as specified by the width and height in the control statement. To set the window the same size as the bitmap used for the Button, use size parms of 0,0.

Button Type Details and Bitmap Requirements

Button type and features are specified during button creation with the ***dwStyle*** parameter of the **`SRCEntBNCreate()`** function.

The following table details each style word used for setting the desired style of button.

`BTN_MNTRY` User defined bitmap Momentary Button. This is the default button type when this or no other button type is specified. This type requires two bitmaps and a third bitmap is optional.

A bitmap for the unpushed state.
A bitmap for the pushed state.
An optional bitmap for the disabled state.

`BTN_SPIN1` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

`15x9` pixel bitmap for horizontal Spin Button

`9x15` pixel bitmap for vertical Spin Button

`BTN_SPIN2` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

`17x11` pixel bitmap for horizontal Spin Button

`11x17` pixel bitmap for vertical Spin Button

`BTN_SPIN3` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

`19x13` pixel bitmap for horizontal Spin Button

`13x19` pixel bitmap for vertical Spin Button

`BTN_SPIN4` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

`21x15` pixel bitmap for horizontal Spin Button

`15x21` pixel bitmap for vertical Spin Button

`BTN_SPIN5` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

◀▶ `23x17` pixel bitmap for horizontal Spin Button

▲▼ `17x23` pixel bitmap for vertical Spin Button

`BTN_SPIN6` Spin Button w/ predefined bitmaps. No user defined bitmaps are required. This button type uses the following predefined bitmaps.

◀▶ `25x19` pixel bitmap for horizontal Spin Button

▲▼ `19x25` pixel bitmap for vertical Spin Button

`BTN_SPINU` User defined bitmap Spin Button. This type requires three bitmaps and a forth bitmap is optional.

A bitmap for the unpushed state.
A bitmap for the incrementing pushed state.
A bitmap for the decrementing pushed state.
An optional bitmap for the disabled state.

`BTN_RED` Red Color Button. No user defined bitmaps are required.

`BTN_GRN` Green Color Button. No user defined bitmaps are required.

`BTN_BLU` Blue Color Button. No user defined bitmaps are required.

`BTN_YEL` Yellow Color Button. No user defined bitmaps are required.

`BTN_CYN` Cyan Color Button. No user defined bitmaps are required.

`BTN_MAG` Magenta Color Button. No user defined bitmaps are required.

`BTN_GRY` Gray Color Button. No user defined bitmaps are required.

`BTN_BLK` Black Color Button. No user defined bitmaps are required.

`BTN_P2S` User defined 2-State Push button. This type requires three bitmaps and a forth bitmap is optional.

A bitmap for the unpushed state 1.
A bitmap for the unpushed state 2.
A bitmap for the pushed state.
An optional bitmap for the disabled state.

`BTN_T2S` User defined 2-State Toggle button. This type requires two bitmaps and a third and forth bitmap is optional.

A bitmap for the state 1 position.
A bitmap for the state 2 position.
An optional bitmap for the disabled state 1 position.
An optional bitmap for the disabled state 2 position.

`BTN_VERT` Specify this style for a vertical Spin Button or 2-State Toggle Button. The default is horizontal. Invalid for all other button types.

`BTN_WRAP` Specify this style to turn ON Spin Button value wrapping. The default is no value wrapping. Invalid for all other button types.

The bitmaps used to paint buttons are specified as bitmap resource names with the ***BM1***, ***BM2***, ***BM3***, ***BM4*** parameters of the **`SRCEntBNCreate()`** and **`SRCEntBNInit()`** functions.

The following chart specifies which bitmap to assign to which bitmap resource name paramerter for the various button types.

	Spin Button (User defined)	**Push Style 2-State Button**	**Toggle Style 2-State Button**	**Momentary Push Button**	**Color Button**
BM1	Unpushed State	Unpushed State 1	State 1 Position	Unpushed State	Undefined (Set to 0)
BM2	Incrementing Pushed State	Unpushed State 2	State 2 Position	Pushed State	Undefined (Set to 0)
BM3	Decrementing Pushed State	Pushed State	Optional Bitmap for Disabled State 1 Position	Undefined (Set to 0)	Undefined (Set to 0)
BM4	Optional Bitmap for Disabled State	Optional Bitmap for Disabled State	Optional Bitmap for Disabled State 2 Position	Optional Bitmap for Disabled State	Undefined (Set to 0)

Button C/C++ API Functions

Button API Function Summary

Function	Purpose
SRCEntBNCreate	Creates a Button control and initializes its settings.
SRCEntBNEnable	Enable/Disable a Button.
SRCEntBNGetDelay	Retrieve the current delay value of a Button.
SRCEntBNGetRange	Retrieve the minimum and maximum values of a Button.
SRCEntBNGetSpeed	Retrieve the current speed value of a Button.
SRCEntBNGetStep	Retrieve the current step value of a Button.
SRCEntBNGetVal	Retrieve the current value of a Button.
SRCEntBNInit	Initialize a Button.
SRCEntBNSetDelay	Sets the delay value of a Button.
SRCEntBNSetHand	Sets a Button window mouse cursor.
SRCEntBNSetRange	Sets the minimum and maximum values of a Button.
SRCEntBNSetSpeed	Sets the speed value of a Button.
SRCEntBNSetStep	Sets the step value of a Button.
SRCEntBNSetVal	Sets the value of a Button.
SRCEntBNSetWrap	Sets the value wrap feature of a Button to ON or OFF.
SRCEntBNStop	Stops a Button.

Button C/C++ API Function Details

SRCEntBNCreate

Purpose

Creates a Button control and initializes its settings.

Syntax

```
HWND SRCEntBNCreate(DWORD dwStyle, LPSTR BM1,
LPSTR BM2, LPSTR BM3, LPSTR BM4, int X, int Y,
int nWidth, int nHeight, int nStep, int rmin,
int rmax, int val, int ndelay, int nspeed,
BOOL bHand, HWND hWndParent, HMENU IDValue,
HINSTANCE hInstance);
```

Description

Use this function to create Button controls.

Returns

`HWND`: A handle to the Button control created.

Parameters

dwStyle

`DWORD`: This specifies the style for the Button being created. See the style word description table for an explaination of all the style words.

BM1

`LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM2

`LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM3

`LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM4

`LPSTR`: The resource name to a bitmap. This is alway the optional bitmap used for disabled buttons. Set to 0(Zero) when no bitmap is specified.

X

`int`: The horizontal position of the upper left corner of the Button control within the parent window client area.

Y

`int`: The vertical position of the upper left corner of the Button control within the parent window client area.

nWidth `int`: The horizontal size of the Button control. When set to 0, the window size is determined by the size of the bitmap used.

nHeight `int`: The vertical size of the Button control. When set to 0, the window size is determined by the size of the bitmap used.

nStep `int`: The step value for a Spin Button control. This is the value by which a Spin Button increments and decrements. This pararameter is ignored by all other button types.

rmin `int`: For Spin Buttons, this is minimum range value. For 2-State Toggle/Push Buttons, this is the State 1 value This pararameter is ignored for Color Buttons and Momentary Buttons.

rmax `int`: For Spin Buttons this is the maximum range value. For 2-State Toggle/Push Buttons, this is the State 2 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

val `int`: The Button initial value. For Spin Buttons, this is any value within its range. For 2-State Buttons, this must be either the State1 or State 2 value. If the neither State 1 or State 2 is specified, the value defaults to the State 1 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

ndelay `int`: The delay value of a Spin Button. This is the time between the moment a Spin Button is pushed and when the Spin Button begins to continue to change value. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value, the longer the delay. A value of 1000 is about 1 second.

nspeed `int`: The speed at which a Spin Button changes value. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value the slower the speed. A value of 1000 is about 1 second.

bHand `BOOL`: Specifies the mouse cursor used within the Button control. Set to TRUE to use the hand mouse cursor. Set to FALSE to use the arrow mouse cursor.

hWndParent `HWND`: A handle to the Button control's parent window.

hMenu `HMENU`: The Button control's ID value.

hInstance `HINSTANCE`: The instance handle for the program module creating the Button control.

SRCEntBNEnable

Purpose Enable or Disable a Button.

Syntax `VOID` **`SRCEntBNEnable`**`(HWND` *`hWnd`*`, BOOL` *`bEnable`*`);`

Returns `VOID`. No returned value.

Parameters

hWnd `HWND`: The Button window handle.

bHand `BOOL`: Set to TRUE to enable the Button. Set to FALSE to disable the Button.

SRCEntBNGetDelay

Purpose Retrieve the current delay value of a Button.

Syntax `int` **`SRCEntBNGetDelay`**`(HWND` *`hWnd`*`);`

Returns `int`: The Button delay value.

Parameters

hWnd `HWND`: The Button window handle.

SRCEntBNGetRange

Purpose Retrieve the minimum and maximum range values of a Button.

Syntax `VOID` **`SRCEntBNGetRange`**`(HWND` *`hWnd`*`, LPINT` *`lpMinVal`*`, LPINT` *`lpMaxVal`*`;`

Returns `VOID`: No returned value.

Parameters

hWnd `HWND`: The Button window handle.

lpMinVal `LPINT`: A pointer to the integer variable that will receive the minimum value of the Button range.

lpMaxVal `LPINT`: A pointer to the integer variable that will receive the maximum value of the Button range.

SRCEntBNGetSpeed

Purpose Retrieve the current speed value of a Button.

Syntax `int` **`SRCEntBNGetSpeed`**`(HWND` *`hWnd`*`);`

Returns `int`: The Button speed value.

Parameters

hWnd `HWND`: The Button window handle.

SRCEntBNGetStep

Purpose Retrieve the current step value of a Button.

Syntax `int` **`SRCEntBNGetStep`**`(HWND` *`hWnd`*`);`

Returns `int`: The Button step value.

Parameters

hWnd `HWND`: The Button window handle.

SRCEntBNGetVal

Purpose Retrieve the current value of a Button.

Syntax `int SRCEntBNGetVal(HWND hWnd);`

Returns `int`: The Button value.

Parameters

hWnd `HWND`: The Button window handle.

SRCEntBNInit

Purpose Initializes a Button's settings.

Syntax

```
HWND SRCEntBNInit(HWND hWnd, LPSTR BM1, LPSTR
BM2, LPSTR BM3, LPSTR BM4, int X, int Y, int
nWidth, int nHeight, int nStep,  int rmin, int
rmax, int val, int ndelay, int nspeed, BOOL
bHand);
```

Description This function will initialize a Button's settings. A call to this function is normally only used **(and must be used)** in a dialog's `WM_INITDIALOG` message handler to initialize a Button created with a `CONTROL` statement in a resource file.

Returns `VOID`: No returned value.

Parameters

hWnd `HWND`: The Button window handle.

BM1 `LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM2 `LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM3 `LPSTR`: The resource name to a bitmap. See the bitmap assignment chart to determine which bitmap to assign to this parameter. Set to 0(Zero) when no bitmap is specified.

BM4 `LPSTR`: The resource name to a bitmap. This is alway the optional bitmap used for disabled buttons. Set to 0(Zero) when no bitmap is specified.

nStep `int`: The step value of the Button control. This is the value by which a Button increments and decrements. This pararameter is ignored for Color Buttons and Momentary Buttons.

rmin `int`: For Spin Buttons, this is minimum range value. For 2-State Toggle Buttons, this is the State 1 value This pararameter is ignored for Color Buttons and Momentary Buttons.

rmax `int`: For Spin Buttons this is the maximum range value. For 2-State Toggle Button, this is the State 2 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

val `int`: The Button initial value. For Spin Buttons, this is any value within its range. For 2-State Buttons, this must be either the State1 or State 2 value. If the neither State 1 or State 2 is specified, the value defaults to the State 1 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

ndelay `int`: The delay value of a Spin Button button. This is the time between the moment a Spin Button is pushed and when the Spin Button begins to continue to change value. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value, the longer the delay. A value of 1000 is about 1 second.

nspeed `int`: The speed at which a Spin Button changes value. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value the slower the speed. A value of 1000 is about 1 second.

bHand `BOOL`: Specifies the mouse cursor used within the Button control. Set to TRUE to use the hand mouse cursor. Set to FALSE to use the arrow mouse cursor.

SRCEntBNSetDelay

Purpose Sets the delay value of a Spin Button.

Syntax `int SRCEntBNSetDelay(HWND hWnd, int ndelay);`

Description This is the time between the moment a Spin Button is pushed and when the Spin Button begins to continue to change value. Ignored by all other button types. Use this function to change the delay value from the one set during creation. Valid value is between 1 and 1000. The larger the value, the longer the delay. A value of 1000 is about 1 second.

Returns `int`: The delay value set.

Parameters

hWnd `HWND`: The Button window handle.

ndelay `int`: The new delay value.

SRCEntBNSetHand

Purpose Sets the Button window mouse cursor.

Syntax `VOID SRCEntBNSetHand(HWND hWnd, BOOL bHand);`

Returns `VOID`. No returned value.

Parameters

hWnd `HWND`: The Button control handle

bHand `BOOL`: Set to TRUE to use the hand mouse cursor. Set to FALSE to use the arrow mouse cursor.

SRCEntBNSetRange

Purpose Sets the upper and lower ranges of a Button control.

Syntax `VOID` **`SRCEntBNSetRange`**`(HWND` *`hWnd`*`, int` *`nMin`*`, int` *`nMax`*`);`

Description Use this function to set a different range from the one set during creation.

Returns `VOID`: No returned value.

Parameters

hWnd `HWND`: The Button window handle.

nMin `int`: For Spin Buttons, this is minimum range value. For 2-State Toggle Buttons, this is the State 1 value This pararameter is ignored for Color Buttons and Momentary Buttons.

nMax `int`: For Spin Buttons this is the maximum range value. For 2-State Toggle Button, this is the State 2 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

SRCEntBNSetSpeed

Purpose Sets the speed value of a Spin Button.

Syntax `int` **`SRCEntBNSetSpeed`**`(HWND` *`hWnd`*`, int` *`nspeed`*`);`

Description The speed value specifies how fast a Spin Button changes its value while it is pushed. Use this function to change the speed value from the one set during creation. Valid value is between 1 and 1000. The larger the value the slower the speed. A value of 1000 is about 1 second.

Returns `int`: The speed value set.

Parameters

hWnd `HWND`: The Button window handle.

nspeed `int`: The new speed value.

SRCEntBNSetStep

Purpose Sets the step value of a Spin Button.

Syntax `int SRCEntBNSetStep(HWND hWnd, int nStep);`

Description The step value specifies the value by which a Spin Button will be incremented or decremented. Use this function to change the step value from the one set during creation.

Returns `int`: The step value set.

Parameters

hWnd `HWND`: The Button window handle.

nStep `int`: The new step value.

SRCEntBNSetVal

Purpose Sets the value of a Spin Button.

Syntax `int SRCEntBNSetVal(HWND hWnd, int nVal);`

Returns `int`: The value set.

Parameters

hWnd `HWND`: The Button window handle.

nVal `int`: The Button value. For Spin Buttons, this is any value within its range. For 2-State Buttons, this must be either the State1 or State 2 value. If the neither State 1 or State 2 is specified, the value defaults to the State 1 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

SRCEntBNSetWrap

Purpose Sets the wrap feature of a Spin Button ON or OFF. This valid for Spin Buttons only.

Syntax `VOID SRCEntBNSetWrap(HWND hWnd, BOOL nWrap);`

Description With the wrap feature turned ON, the Spin Button value will wrap around from the range maximum value to the range minimum value and from the range minimum value to the range maximun value.

Returns `VOID`: No return value.

Parameters

hWnd `HWND`: The Spin Button window handle.

bWrap `BOOL`: TRUE to turn the wrap feature ON, FALSE to turn the wrap feature OFF. The wrap feature can be set ON with the `SPB_WRAP` style when the Spin Button is created.

SRCEntBNStop

Purpose Stops a Spin Button.

Syntax `VOID SRCEntBNStop(HWND hWnd);`

Description With this call, the Spin Button can be stopped while it is still activated with the mouse. Mainly used if an error occures while the Spin Button is activated. It should be called just before an error handling routine is called.

Returns `VOID`: No return value.

Parameters

hWnd `HWND`: The Spin Button window handle.

Button Visual Basic Extention (VBX)

For Visual Basic and Visual C++ programmers using MFC, the Button Custom Control can be used as a Visual Basic Extention by copying or renaming the SRCENTBN.DLL file to SRCENTBN.VBX. The Button can then be added to the Visual Basic design mode control toolbox.

Button VBX Properties

BM1_Bitmap — See the bitmap assignment chart to determine which bitmap to assign to this property. No bitmap is needed for the predefined Spin Buttons and the Color Buttons.

This is a design-time only property.

BM2_Bitmap — See the bitmap assignment chart to determine which bitmap to assign to this property. No bitmap is needed for the predefined Spin Buttons and the Color Buttons.

This is a design-time only property.

BM3_Bitmap — See the bitmap assignment chart to determine which bitmap to assign to this property. No bitmap is needed for the predefined Spin Buttons and the Color Buttons.

This is a design-time only property.

BM4_Bitmap — See the bitmap assignment chart to determine which bitmap to assign to this property. No bitmap is needed for the predefined Spin Buttons and the Color Buttons.

This is a design-time only property.

Size_to_Bitmap — By setting this property to TRUE, the button is forced to size itself to the same size as the bitmap used for the button. When set to FALSE, the bitmap is stretch to the selected size. This property is ignored for Color Buttons since they don’t use bitmaps.

This is a design-time only property.

Mouse_Pointer

With this property, either the arrow or the hand mouse pointer can be selected.

This is a design-time and run-time controllable property.

Type

With this property, the Button type can be selected. See the Button Type Details section for a description of all the button types.

This is a design-time only property.

Orientation

With this property you select whether the button operates vertically or horizontally. This is ignored by Color, Momentary, and 2-State Push Buttons.

This is a design-time only property.

Wrap

With this property, the WRAP feature for Spin Buttons is to either TRUE or FALSE.

This is a design-time and run-time controllable property.

Min

For Spin Buttons, this is minimum range value. For 2-State Toggle/Push Buttons, this is the State 1 value   This property is ignored for Color Buttons and Momentary Buttons.

This is a design-time and run-time controllable property.

Max

For Spin Buttons this is the maximum range value. For 2-State Toggle/Push Buttons, this is the State 2 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

This is a design-time and run-time controllable property.

Value

The Button initial value. For Spin Buttons, this is any value within its range. For 2-State Buttons, this must be either the State1 or State 2 value. If the neither State 1 or State 2 is specified, the value defaults to the State 1 value. This pararameter is ignored for Color Buttons and Momentary Buttons.

This is a design-time and run-time controllable property.

Delay

The delay value of a Spin Button. This is the time between the moment a Spin Button is pushed and when the Spin Button begins to continue to change value. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value, the longer the delay. A value of 1000 is about 1 second.

This is a design-time and run-time controllable property.

Speed

The speed value specifies how fast a Spin Button changes its value while it is pushed. Ignored by all other button types. Valid value is between 1 and 1000. The larger the value the slower the speed. A value of 1000 is about 1 second.

This is a design-time and run-time controllable property.

Step

The step value for a Spin Button control. This is the value by which a Spin Button increments and decrements. This pararameter is ignored by all other button types.

This is a design-time and run-time controllable property.

Enable

This is a run-time only property which does not appear in the properties window. In programm code, set this to TRUE to Enable the Button or FALSE to disable the Button.

Name

This is the standard Name property that specifies the name of the Button. Use this name when referencing the Button in program code.

The default name is SRCEntButton1.

Top

This is the standard Top property that specifies the vertical location of the upper left corner of the Button control within the form.

Left

This is the standard Left property that specifies the horizontal location of the upper left corner of the Button control within the form.

Width

This is the standard Width property that specifies the width of the Button control.

Height

This is the standard Height property that specifies the height of the Button control.

Button VBX Event

The Button control has only one event called **`NewVal`**. When a Button value changes, the Button control fires the **`NewVal`** event. It is in this **`NewVal`** event that your program code should reference the Value property of the Button and assign the Value to other variables or controls depending on what your application program is designed to do.

There is no value for the Momentary and Color Button types, so it is in the **`NewVal`** event where you program code processes the desired action when these buttons are clicked.