



BNCS Custom Controls

BBC_CC Base Library

Author: Simon Dowson

Changes & Updates: Steve Jensen

© Copyright 1993-2004 BBC
© Copyright Siemens IT Solutions & Services 2005-2011
© Copyright Atos IT Solutions & Services 2011-2015

Contents

Contents	2
Custom Control Base Library	3
1. Introduction	3
2. Controls	4
3. Application Notes	8
4. Version History	9

Custom Control Base Library

1. Introduction

This library was developed primarily for use with ApplCore, a generic control panel core application. The existing control libraries provided by Microsoft and Borland at the time did not contain sufficient controls to exploit the highly flexible Windows environment when used in control panel applications. This library contains controls which are aimed at use in a broadcast environment as well as permitting colour and multiline text.

1. Standard button
2. Text box
3. LED button
4. Destination button
5. Browse button
6. Ident box
7. Bitmap button
8. Touch button
9. Flat button
10. Multi window button
11. Vertical fader control
12. Horizontal fader control

With the exception of the Ident Boxes and Bitmap Buttons all controls have the functionality of ordinary Windows controls but with added features. The background colour and text colour can be changed. Text can be split over many lines and automatically positioned within the control.

2. Controls

- **Standard Button**

The standard button can contain any number of rows of automatically centred text. The text colour and button colour can be changed by sending it a valid attribute change command. The format of the attribute change command is the same for all BBC custom controls and is described later.

- **Text Box**

The text block colour and text colour can be changed. This control is also useful for creating thin coloured borders etc.

- **LED Button**

The LED button has a small LED in the top left hand corner. The colour and brightness of the LED can be changed. The button and text colour can also be changed. The default LED colour is black or 'Off'.

- **Destination Button**

The Destination button has two independent text blocks. They can be written to independently and the colour of all texts and backgrounds may be changed independently.

- **Browse Button**

This is similar to the standard button except that the button activates as soon as it is pressed instead of when it is released. As you drag the mouse pointer across the face of browse buttons they will depress and activate. Unlike other buttons the browse button has a guard band of 4 pixels around its perimeter. Touching the button on the guard band will not activate it. This is to prevent a stream of commands being generated when the cursor hovers on the border between two buttons. This is particularly noticeable with touch screens which introduce more cursor 'jitter' than a mouse. These buttons are useful for monitoring panels.

- **Ident Box**

The ident box is intended for use as a caption or vision ident. The borders of the box are more sharply defined than an ordinary text box. The text placed in an ident box is automatically centred and sized to fit the available space. The colour attributes are the same as for the text box.

- **Bitmap Buttons**

This button will display any bitmap resource within the application. It is intended for use with the default button size of 64 x 40, but will work with any size button or bitmap. The bitmap is not scaled to fit the control. The bitmap button provides the highlighted borders for both the raised and depressed states of the button so to place the bitmap exactly within the button it should have a height and width 6 pixels less than the height and width of the button. For the default button size of 64 x 40 the correct bitmap size is 58 x 34.

The name bitmap to be displayed on the button is the name placed in the button text.

If a different bitmap name is sent to the button then the displayed bitmap will change immediately. When a button is pressed the bitmap is automatically displaced to the left and down. The shadow low light is done by the control itself. Bitmap buttons are not displayed in the dialogue box resource editor, but will appear in your application when it is executed.

- **Touch Button**

A touch button behaves in the same manner as a standard button except that the button activates when it is pressed instead of when it is released.

- **Flat Buttons**

Flat buttons look like text boxes, but act like buttons. When pressed a flat button will change colour and respond with its control ID. When released it will return to its previous colour and send its control ID + 1000. You can browse from one flat button to another. Control ID's are generated for each button.

The colour of the button and its text is controlled via the LABEL attributes and the pressed colours via the BUTTON attributes. The border of the button is controlled via the LED colour attribute. See the Control Attributes section for more information.

• Multi Button

The Multi button is similar to a destination button and is controlled the same way. It has two additional text fields added beneath the control. The text block switch is used to access these extra fields. Block 2 is the bottom left and Block 3 is the bottom right.

Once the desired text block has been switched in the background colour and the text colour can be changed.

• Vertical Fader Control

The Vertical Fader is primarily intended as a continuously variable control for analogue or pseudo-analogue devices. The control consists of a box split horizontally into two sections. Where the sections meet is called the 'Level' line. The line can be moved up or down. The colour of the top and bottom sections can be changed as can the colour of level line itself.

The fader control is accessed in a similar way to a button control. The control has 3 Sub-ID's which are accessed using Text Block Switches of the TP command. Text Block Switches are made by using the 6th parameter of the TP `'/_ _ _ _ _' <PnlID> <CtlID>` command and effectively divert all subsequent instructions to that Sub-ID of that Control. Unless switched the default Text Block for a Control is 0.

Text block 0 holds the current position or level of the fader. To obtain this information you need to switch the Text Block to 0, (TP `'/_ _ _ _ _ 0' <PnlID> <CtlID>`), if it's not already and then do a Text Get for that CtlID. The button background colour position of the TP `'/_ _ _ _ _' <PnlID> <CtlID>` command sets the colour of the level line itself.

Text block 1 holds the minimum range value of the Fader Control and can be set by first switching to Text Block 1 (TP `'/_ _ _ _ _ 1' <PnlID> <CtlID>`), and then writing the required minimum value to the Ctl by using the TP `<MinVal%> <PnlID> <CtlID>` command. The background label colour for text block one sets the colour of the fader below the level line.

Text block 2 holds the maximum range value and the background label colour for text block two sets the colour of the fader above the level line and is accessed as above but using Text Block Switch 2.

So with a fader range set to, for example, 20 and 80 the level offered by the Control's Switch Block 0 will fall proportionally between these two limits.

For further information on Custom Control Attributes review document: CC_Atribs.pdf, and the ApplCore Text Put command in document: v3_applcore_commands.pdf.

- **Horizontal Fader Control**

This is the horizontal version of the vertical fader. It works exactly as described above except that the operation of the control is left to right instead of up and down.

3. Application Notes

For further information review:

Custom Control Attributes document: **CC_Atribs.pdf**

ApplCore Text Put command in document:
v3_applcore_commands.pdf

Normally the library is used in support of ApplCore.

Please note when the library is used with a Windows program other than ApplCore it can be useful to know what Windows messages the library responds to and utilises.

The Windows messages used / responded to include:

- WM_PARENTNOTIFY
- WM_STYLECHANGING
- WM_STYLECHANGED
- WM_CREATE
- WM_SIZE
- WM_SETTEXT
- WM_GETTEXT
- WM_PAINT
- WM_LBUTTONDOWN
- WM_LBUTTONUP
- WM_MOUSEMOVE
- WM_ENABLE
- WM_CLOSE

4. Version History

- 24-November-1999 - Original version BBC_CC, this document was available only as a PDF page (source document missing), dated: 24-November-1999. SPJ
- 19-September-2013 - Word based document created from PDF. SPJ
- July-2013 - BNCS_CC and BNCS_CC09 custom control library release. BNCS_CC Version: 3.01.01, Dated: 16-July-2013, BNCS_CC09 Version: 3.27.00, Dated: 16-July-2013. Corrections made to BNCS_CC library for bug 2373 have been applied to the BNCS_CC09 library as well. The change is to move to a "double buffer" rendering scheme and then BitBlt to the GDI to correct "flashing" graphics issues noted on Windows 7. SPJ
- May-2015: BNCS_CC Version: 3.01.02, Dated: 25-Feb-2015. Added support for 4.5 environment. Bugs 2637
- August-2015: BNCS_CC Version: 3.01.03, Dated: 13-Aug-2015. 4.5 environment full Win32 path length. Bugs 2637

Atos IT Solutions and Services Limited
Faraday House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
Tel.: +44 (0)1276 696000