Controls

Controls are the means by which a user communicates with an application program.
Traditionally, implementing a GUI interface, using the inbuilt features of the Mac Operation System, involved a great deal of work in design and programming.
The Crimson Basic Development System automates GUI programming so that the programmer can concentrate on the logic of the application at hand.
The Form Designer is used to design Forms and extensions to the Basic language are used to communicate with them (see Properties, Methods and Events). Several standard controls are available for use within the Form Designer and they are described below.

Form (Window)

A Form is used as a container for other controls and is implemented as a standard Macintosh Window (or dialog box).
The Form is implemented as a modeless Dialog Box which may optionally have a 'Close Box'.
The Background Colour of a Form may be set at Design time or Run Time (using the UseColour, ColourRed,ColourGreen and ColourBlue properties).
A Form may be opened and closed using the OpenWindow and CloseWindow methods and can be made visible or invisible by use of the ShowWindow and HideWindow methods.
Forms respond to user events by coding the following event procedures :

Click
e.g.

```
Procedure Form1.Click()
......
Return
```

The Click procedure is called whenever the user clicks the mouse and the mouse is positioned over the Form (and not over a control within the Form).

DClick
e.g.

```
Procedure Form1.DClick()
......
Return
```

The DClick procedure is called when the user double clicks the mouse within

the Form. Note that the Click procedure is also called in this case.

MouseDown
e.g.

```
Procedure Form1.MouseDown(x,y)
Parameter x As Integer
Parameter y As Integer
......
Return
```

The MouseDown procedure is called when the user presses the mouse within a Form. The two parameters supplied to the procedure give the x and y coordinates of the mouse within the Form at the time it was pressed.

MouseUp
e.g.

```
Procedure Form1.MouseUp(x,y)
Parameter x As Integer
Parameter y As Integer
......
Return
```

The MouseUp procedure is called when the user releases the mouse and is positioned over the Form. The x and y coordinates are supplied as in MouseDown above.

Onmenu
e.g.

```
Procedure Form1.OnMenu(menu_id, menu_item)
Parameter menu_id As Integer
Parameter menu_item As Integer
......
Return
```

The OnMenu procedure is called whenever the user selects a menu item and this form (Form 1 in this case) is the active one. The menu_id and menu_item parameters supplied allow the program to determine which menu item has been selected.

GotFocus
e.g.

```
Procedure Form1.GotFocus()
......
Return
```

The GotFocus procedure is called whenever the Form is activated, either by

being opened (using the OpenWindow method) or by the user clicking on the Form and bringing it to the front.
The GotFocus event is often used to change menu settings to suit the currently active Form.

Button

A Button is a simple control which responds to user mouse events.
A button may be displayed in its standard form or with a 3D effect, controlled by the Appearance property. The button is positioned at design time using the Form editor or at run time using the Height, Width, PositionX and PositionY properties.
The Fill Colour and Border Colour of the button are set at design time.
A Button responds to the Click, DClick, MouseDown and MouseUp events.

List Box

The List Box is the most complex control provided with Crimson Basic, however, interacting with a List Box is made easy with the supplied Properties, Methods and Events.
A List Box may have a Flat or 3D appearance, controlled by the Appearance property.
The MultiSel property controls the highlighting in a List Box. If MultiSel is set to False, only a single row can be selected, if set to True, then multiple rows can be selected.
The size and position of the List Box can be manipulated using the Height, Width, PositionX and PositionY properties.
A complex control like the List Box needs many methods to control its operation. These are :
SetRow - Sets a row in the List Box to a given value
InsertRow - Inserts a new row into the List Box
GetRow - gets the value of a given row in the List Box
IsSelected - used to check if a given row has been selected
GetSelected - Searches for selected rows.
DeleteRow - Deletes a given row or rows
SetSelected - Sets the selected state of a given row to On.

The standard Click, DClick, MouseDown and MouseUp events are supported.

The OnKey event is fired when a key is pressed and the List Box is the currently active control (the last one to be clicked). It returns as parameters, the ascii value of the key pressed and an indicator which is set to True if the event was fired in response to a key being held down.

Rectangle

The Rectangle control draws a rectangle shape on the form which may be : Flat, have a 3D border, be filled with a Colour or have a Colour border depending on the Appearance property.
Its size and position on the Form are controlled with the height, Width, PositionX and PositionY properties.
The colour of the rectangle (where appropriate) may be changed by setting the ColourRed, ColourGreen and ColourBlue properties.
The Rectangle control supports the Click, DClick, MouseDown and MouseUp events.

Static Text

The Static Text control is used for providing text within a Form, usually in the form of a label.
The TextJustify property may be set to justify the text Left, Right or Centre within the area in which it is drawn.
The Value property may be used to set or retrieve the value of its text.

Colour Icon Button

The Colour Icon Button is of a fixed size (32 * 32 pixels) containing an Icon (resource type 'cicn') which may be designed using the ResEdit utility.
Two properties, Off_Cicn_id and On_Cicn_id determine which 'cicn' resources are used to render the icon on the button in its Off and On state.
If the AutoState property is set to True then the button will automatically display the 'On' icon when the button is pressed, otherwise, the button can be manually set On and Off using the State property.

To design an Icon using ResEdit, follow this procedure :

Build your application, thus creating an object file.
Drag the object file onto the ResEdit utility.
Select Resource / Create new Resource from the menu.
Select resource type 'cicn' from the displayed List Box.
The Icon design screen is now displayed for 'cicn' number 128 (this number will increment for each new icon designed. use this number for the Off_Cicn_id or On_Cicn_id properties.
After designing the icon (its easy), select File/Save from the menu to save your work.

Check Box

The Check Box control allows the user to toggle the state of the Check between On and Off by clicking upon it.

The caption to the right of the Check may be changed by setting the Caption property.
Its size and position are changed using the Height, Width, PositionX and PositionY properties.
The state of the Check box (its On/Off state) may set or retrieved by using its Value property.
The MouseDown event is supported.

Editable Text

The Editable Text control allows a user to enter a string of text into the position on the Form allocated to the control.
The text within the control can be set and retrieved using the Value property.
The DataType property allows some control over the data which the user is allowed to type in to the Editable Text field:
Any - any characters
Integer - numeric characters only (preceded by + or -)
float - numbers in floating point form, e.g. 1.234, 33E+2.

Radio Button

The Radio Button control allows a use to select one choice from many.
Clicking on one Radio Button, within a group, will set that button on, and set all other Radio Buttons within that group off.
The state of an individual button can be set or retrieved using the Value property.
The Radio_group property can be used at design time to assign a Radio Button to a Group.

Picture

The Picture control renders a picture with the bounds of the control.
The PICT_id property contains the resource id of the PICT resource in the object code file.
To import a picture into the object code file from a pict file :

Compile and build your application.
Drag the pict file onto the SimpleText application.
Select Edit/Select all.
Select Edit / Copy.
Drag the object code of the application onto RedEdit.
Select Resource/Create new Resource.
Select PICT from the list box. A window will be opened for the picture. The resource id of the picture can be found in the Window title.
Select Edit/Paste.

Select File/Save.

Pop Up Menu

The Pop Up Menu control allows the user to choose a value from many pre-set choices. It performs a similar function to the Radio Button control in that the user will always select one value from a list of many.
The Radio Button control is normally used to select one of up to 4 choices. Where more than 4 choices exist, the Pop Up Menu should be used.
The Value property may be used to set which entry in the Pop Up Menu is current and to retrieve the number of the currently selected entry.
The MouseDown and MouseUp events are supported.