Event

Event

VISUAL DEVELOPMENT AND EVENT DRIVEN PROGRAMMING

With visual programming the programmer has the ability to create graphical user interface by pointing and clicking with the mouse. The programmer creates the GUI and writes the code to describe what happens when user interacts (click, press a key, double click etc) with the GUI. These notifications called "EVENT". The events are passed into the program by Microsoft's Windows Operating System. We programming the code that respond to these events is called "EVENT DRIVEN PROGRAMMING". With event driven program the user dictates the order of program execution - not the programmer, so instead of the program driving the user, the user drives the program.

We specify sub-routine for a particular event the sub-routine determines how control reacts to an event is called an "EVENT HANDLER"

Events are the things that we can do to our control that it will recognize and we able to respond to. Each control has a set of events it understands. We know that visual basic programming is done in two steps -

1) User interface designing
2) Event Driven Programming

The events that are used in programming are of following types -

1) Mouse related events.
2) Keyboard related events.
3) General events.

1) Mouse Related Events → Visual Basic controls are capable of recognizing when the mouse button is single click, double click as well as other. Mouse events are such as - moving the mouse, pressing the mouse button and releasing the mouse button. Like other events the programmer writes the code or perform some action when the event is triggered (or raised). The programmer can test which mouse button (left, middle or right) was pressed. Visual Basic also supports for drag and drop. Like dragging an item on the screen and dropping it on another item.

2) Keyboard Related Events → Keyboard is one of the most fundamental device for any program. We can easily determine the users key-stroke by a manner in which Visual Basic implements Event Driven programming as the user press and release key or group of keys on the keyboard VB generates events for object that support time, the text control in Visual Basic can handle keyboard

input in most situations, but sometimes we need more direct control over keyboard input such as when handling special keys and key combinations. Keyboard event along with mouse events are the primary elements of a user's interactions, with our program. When a user interacts with keyboard the key events are generated, VB provides three event procedures for handling key events ie,- Keypress, Keyup, Keydown. Event procedure Keypress and Keydown are called when Key is pressed and Keyup is called when pressed key is realesed.

3) General Events → Besides the mouse and keyboard related events others are called general events, like- Gotfocus, Lostfocus, Activate, Deactivate, Change, Load, Unload etc.

The mouse related events are described below -

(a) Click Event → This event occurs when user clicks anywhere on the form. Event procedure click is called when a mouse button is clicked once. User can click on a list box item, command button etc. then click event occurs.

(b) Double Click → The DblClick() event takes place when the user double clicks the left mouse button. The Dblclick event is define in a sub-procedure ie,

named starting with the name of control on the form, followed by an underscore (_) and Dblclick() If the reference control is a part of control array, the Dblclick() is followed by an index variable in parenthesis. The Dblclick event occurs after we press and release the left ~~and~~ or right mouse button twice in quick succession.

General syntax -

Private sub object_dblclick ([index as integer])

(c) Mouse Move → The MouseMove event occurs when the user moves the mouse.

Syntax -

Private sub object_MouseMove ([index as Integer], shift as integer, x as single, y as single)

Private sub form_MouseMove (button as integer, shift as integer, x as single, y as single)

Here -

Part	Description
Object	An object expression that evaluates to an object in the applies to list.

Part.	Description.
Index	An integer that uniquely identifies a control if it is a control array.
Button	An integer that corresponds to the state of the mouse button in which a bit is set if the button is down. The button argument is a bit field with bits corresponding to the left button (bit 0), right button (bit 1), middle button (bit 2) These bits corresponds to the value 1, 2, 4 respectively.
Shift	An integer that corresponds to the state of shift, ctrl, Alt, keys. A bit is set if the key is down. The shift argument is a bit field with least significant bit correspond to the shift key (bit 0), the ctrl (bit 1) and the alt key (bit 2). These bits corresponds to the value 1, 2, 4 respectively.
(x, y)	Specifies the current location of the mouse pointer, the x and y values are always expressed in terms of co-ordinate system set by scalewidth, scaleheight and scaletop properties of the object.

The mousemove event is generated continuously as the mouse pointer moves across the object. An object recognize a mousemove event whenever the mouse position is within its border. We can use constants listed in the VB library object.

Constant Button	Value	Description
VbLeftButton	1	Left Button is pressed.
VbRightButton	2	Right Button is pressed
VbMidButton	3	Middle Button is pressed.

Constant for the shift -

Constant Shift	Value	Description.
VbShiftMask	1	Shift Key is pressed
VbCtrlMask	2	Ctrl key is pressed.
VbAltMask	3	Alt Key is pressed.

(d) MouseUp -> MouseUp event occurs when user releases any mouse button.

(c) MouseDown → MouseDown event occurs when user press any mouse button.

Following are some common keyboard events –

(a) KeyDown → KeyDown event or low level keyboard handling. This event reports the current status of the keyboard when a key is pressed and this event's control has the focus. Means, the keyDown event Triggered when a key is pressed. Syntax –

Private sub object_KeyDown ([Index as Integer], Keycode as integer, Shift as integer)

(b) KeyUp → The keyup event is used for low level keyboard handling. It reports the current status of the keyboard when key is released, and this event's control has the focus. Syntax –

Private sub object_KeyUp ([Index as Integer], Keycode as integer, shift as integer)

Part	Description
Object	An object expression that evaluates to an object in the applies to list

Part	Description.
Index	An integer that uniquely identifies a control if its in a control Array.
KeyCode	A keycode such as VbKeyF1 (The f1 key) or VbKeyHome (The home key). To specify keycode use the constants in VB object library in the object between.
Shift	An integer that corresponds to the state of the shift, ctrl and Alt key at the time of the event. The shift argument is a bit field with least significant bit corresponding to the shift key (bit 0), Ctrl Key (bit 1) and Alt Key (bit 2). These bits corresponds to the values 1, 2 and 4 respectively. If both Ctrl and Alt are pressed then the value of shift argument is 6.

(c) Keypress → The keypress event is used frequently to write keyboard handlers for textboxes because this event takes place before the character is pressed is displayed in the textboxes. Keypress is differ from Keydown event keypress can not detects keys such as Shift, Ctrl, Alt. Depending on which key is pressed all three event procedure may be called. Common Key event constants are listed below -

Constants	Ascii Values	Description
VbKeyA - VbkeyZ	65-90	A key through Z Key
VbkeynumpadO - Vbkeynumpad 9	96-105	Keypad Numeric Key 0 to 9.
VbKey0 - VbKey 9	48-57	Numeric Key 0 to 9
VbKeyF1 - VbKey F16	112-127	Function Key F1 - F16
VbKeyDecimal	110	Decimal Point Key (Period key)
VbKeyBack	8	Back Space Key
VbKeyTab	9	Tab Key
VbKeyReturn	13	Return Key (Enter Key)
VbKeyShift	16	Shift Key
VbKeyControl	17	Ctrl Key
VbKeyCapital	20	Caps Lock Key
VbKeyEscape	27	Escape Key
VbKeySpace	32	Space Bar

VbKeyInsert	45	Insert Key
VbKeyDelete	46	Delete Key

Syntax –

Private sub object_KeyPress ([Index as Integer], KeyAscii as integer)

Here,

Object → An object expression that evaluates to an object in the Applies to list.

Index → An integer that uniquely identifies a control if it a control array.

KeyAscii → An integer that returns standard numeric Ascii Keycode. KeyAscii is pass by reference.

General Events.

Following are some general events described below –

(a) Activate → This event occurs when a form gets focus. If an application contains multiple forms, the activate event occurs when a user changes to a different form by clicking on the form.

(b) Deactivate → This event occurs when another form gets the focus. therefore both the activate and deactivate event occurs when the user selects a different forms.

(c) Load → This event occurs right as the form is loaded into active memory and appears on the screen. The form will normally come into view at some point after the end of form's load event. B'coz the code in the load event could go off and do a hundred or more other things such as setting up variables to data, or doing some calculations in loops.

(d) Unload → This event occurs when the application removes a form from the window using code: When the application is terminated all loaded forms are first unloaded, so we must write an unload event procedure for each form.

(e) Resize → This event occurs when the user change size of the form. A resize event happens and we can put code in our control to react to that event that is exactly what we have done here.

Writting event procedure

The event procedure are section of code that handle a particular control's event. One control might have several event procedure. if we want to respond to several different kinds of event for that control, Visual basic uses an event procedure's name to determine these two things about the procedure -

* which control will trigger the procedure.
* which event will trigger the procedure.

Here, is the formate of all event procedure name -

controlname_eventname()

The underscore seperate the control name from the event name. All event procedures are named this way. Therefore, an event procedure cmdadd_click() executes if and only if the command buttons name is cmdadd and click event is performed in that.