CHAPTER 1

Getting Started with VB

1.1 The IDE

IDE (Integrated Development Environment) or Integrative development environment is a software application that provides comprehensive facilities to computer programmers for software development.
An IDE normally consists of a source code editor, build automation tools and a debugger.
Some IDEs contain a compiler, interpreter or both such as Microsoft Visual Studio and Eclipse.
Sometimes a version control system and various tools are integrated to simplify the construction of GUI.
IDEs are designed to maximize programmer productivity by providing integrated components with similar user interface.
IDE present a single program in which all development is done.
This program typically provides many features for authoring, modifying, compiling, deploying and debugging software.
IDEs are having a set of features that most closely matches the programming paradigms of the language.

IDE must contain following features :

1. Code completion or code insight : The ability of an IDE to know a language, knowledge and function names is crucial. The IDE may are knowledge to do such thing as highlight errors, suggest a list of available function based on the appropriate satiation or offer a function definition.
2. Resource Management : When creating application, language often rely on certain resources like library or header file to be act specific locations. IDE should be able to manage these resources.
3. Debugging tools : In an IDE, you should be able to thoroughly test your application before release. The IDE must be able to give variable values at certain points, connect to different data repositioneries or accept different run-time parameter.
4. Compile & build for language that require a compile or build stage, IDEs translate code from high level language to the object code of the targeted platform.

Advantages of using IDE :

1. Less time and effort : The entire purpose of an IDE is to make developing faster and easier its tools and features are supposed to help you organize resources, prevent mistakes and provide shortcuts.
2. Enforce project or Company Standard : Simply by working in the some development environment, a group of programmers will adhere to a standard way of doing things.
3. Project Management : This can be two fold,
    a. first, many IDEs have documentation tools that either automate the entry of developer comments, or may actually force developers to write comments in different areas.
    b. Second, simply by having a visual presentation of resources. It should be a lot easier to know how an application is laid out as opposed to traversing the file system.

Disadvantages of IDE :

1. Learning Curve - IDEs are complicated tools. Maximizing their benefit will require time and patience.
2. A sophisticated IDE may not be a good tool for beginning programmers. If you skip the learning curve of an IDE on top of learning how to program. It can be quite hard but the features and shortcut for experienced programmers often enjoy this programming.
3. Will not fir bad code, practices or design. An IDE will not eliminate efficiency or performance problems in

your application.

Integrated Development Environment Elements

The Visual Basic integrated development environment (IDE) consists of the following elements.

Menu Bar

Displays the commands you use to work with Visual Basic. Besides the standard File, Edit, View, Window, and Help menus, menus are provided to access functions specific to programming such as Project, Format, or Debug.

Shortcut Menus

Contain shortcuts to frequently performed actions. To open a shortcut menu, click the right mouse button on the object you're using.

Toolbars

Provide quick access to commonly used commands in the programming environment. You click a button on the toolbar once to carry out the action represented by that button. By default, the Standard toolbar is displayed when you start Visual Basic. Additional toolbars for editing, form design and debugging can be opened.
Toolbars can be docked beneath the menu bar or can "float" if you select the vertical bar on the left edge and drag it away from the menu bar.

Toolbox

Provides a set of tools that you use at design time to place controls on a form.

Project Explorer Window

Lists the forms and modules in your current project. A *project* is the collection of files you use to build an application.

Properties Window

Lists the property settings for the selected form or control. A *property* is a characteristic of an object, such as size, caption, or color.

Object Browser

Lists objects available for use in your project and gives you a quick way to navigate through your code

Form Designer

A window that is used to design the interface of your application. You add controls, graphics, and pictures to a form to create the look you want. Each form in your application has its own form designer window.

Code Editor Window

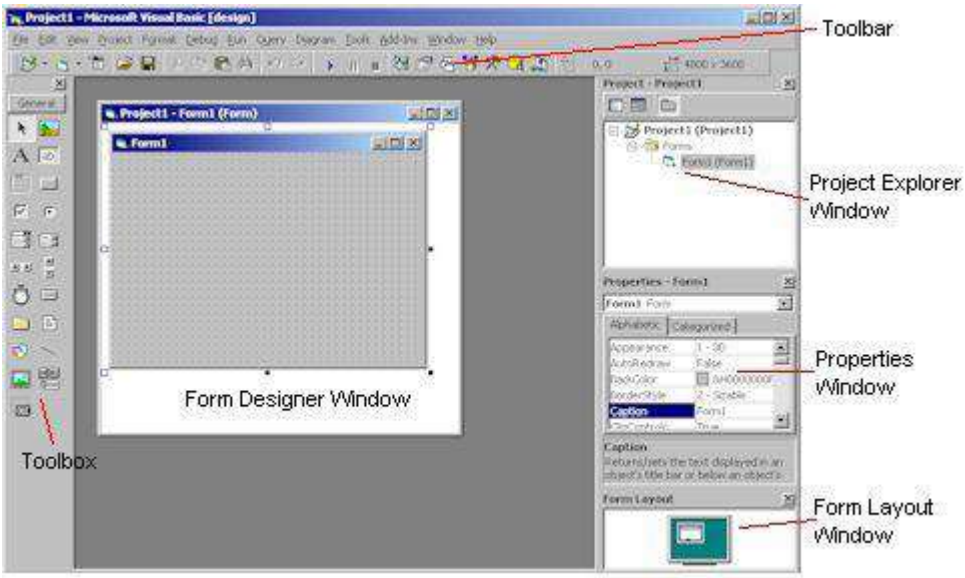
A Window used for entering application code. A separate code editor window is created for each form or code module in your application.

Form Layout Window

The Form Layout window is used to position the forms in your application using a small graphical representation of the screen.

Immediate, Locals, and Watch Windows

These additional windows are provided for use in debugging your application. They are only available when you are running your application within the IDE.



2.1 Menu Bar

The menu bar contains the commands you need to work with Visual Basic. The basic menus are:
File : - contains the commands for opening and saving projects and creating executable files and a list of recent projects.
Edit : - contains editing commands (e.g., Undo, Copy, Paste) plus a number of commands for formatting and editing your code (e.g., Find, Replace).
View : -contains commands for showing or hiding components of the IDE.
Project : - contains commands that add components to the current project, references to Windows objects, and new tools to the Toolbox.
Format : - contains commands for aligning the controls on the Form.
Debug : - contains the usual debugging commands.
Run : - contains the commands that start, break, and end execution of the current application.
Tools : - contains tools you need in building
Add-Ins : - contains add-ins that you can add and remove as needed. By default, only the Visual Data Manager Add-In is installed in this menu. Use the Add-In Manager command to add and remove add-ins.
Window : - is the standard Window menu of an application that contains commands to arrange windows on the screen.
Help : - contains information to help you as you work.
Query : - contains commands that simplify the design of Stuctured Query Language (SQL) queries. This menu is available when building database applications.
Diagram : - contains commands for editing database diagrams. This menu is available when building database applications.

Toolbars

The toolbars give you quick access to commonly used menu commands. Besides the main toolbar, which is displayed by default below the menu bar, the Visual Basic IDE provides additional toolbars for specific purposes, such as editing, Form design, and debugging. To view the additional toolbars, choose **View ➢ Toolbars**.

1. The Standard toolbar

The Standard toolbar is just below the menu bar and is displayed by default. The commands which are commonly used by user are present on this toolbar.

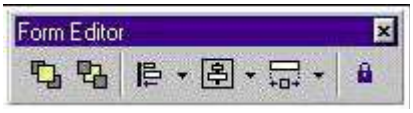
2. The Edit toolbar

The Edit toolbar contains the commands of the Edit menu. The commands in this toolbar are accessible only when user is in code window.

3. The Debug toolbar

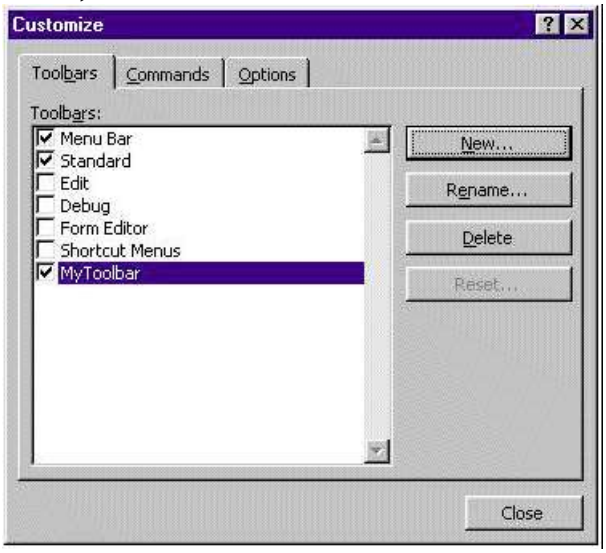
The Debug toolbar contains the commands of the Debug menu.

4. The Form Editor toolbar



The Form Editor toolbar contains the commands of the Format menu. The commands in this toolbar are accessible only when user is in design window.

To open and close toolbars, choose **View ➢ Toolbars** to display a submenu containing the names of the toolbars. These names are toggles and turn the corresponding toolbars on and off.
Choose the Customize command to customize the appearance and contents of menus. To customize default toolbars, choose **View ➢ Toolbars ➢ Customize** to open the Customize dialog box, as shown in following figure.

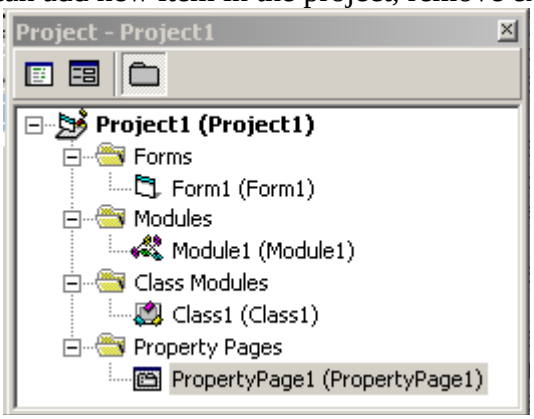


The Customize dialog box has three tabs:

Toolbars On this tab you can specify which toolbars will be visible, rename toolbars, delete them, and even create new ones. To create a new toolbar, click the New button and enter the new toolbar's name when prompted.
Commands This tab contains a list of the main menu options and a list of the commands of the selected option. Use this tab to add commands to your custom toolbars.
Options On this tab you can specify some general options for all toolbars. You can switch between small and large icons, specify whether ScreenTips display, and specify whether menus animate.

Project Explorer

Project Explorer Window Displays a hierarchical list of the projects and all of the items contained in a project. We can add new item in the project, remove existing. The following fig shows the property window.



List Window

Window Elements are

View Code-This Option or button displays the **Code** window so you can write and edit code associated with the selected item.

View Object- This Option or button displays the **Object** window or form designer window for the selected item, an existing form, module, ActiveX object.

Toggle Folders - This Option or button Hides and shows the object folders while still showing the individual items contained within them.

List window- List Window Lists the all loaded projects and the items included in each project.
The project and items contained within it are.

- Forms -All .frm files associated with the project.
- Modules -All .bas modules for the project.
- Class Modules -All .cls files for the project.
- User Controls -All user controls for the project.
- User Documents -All document objects, .dob files, in the project.
- Property Pages -All property pages, .pag files, in the project.
- ActiveX Designers - All designers, .dsr files, in the project.
- Related Documents -Lists all documents to which you want a pointer.
- Resources- Lists all of the resources you have in your project.

A project can contain multiple files which can be opened, closed, and saved together. You can add new item in the project explorer window by simple right clicking on project explorer window and choosing add option. The new

item can be added in the project by using project window. You can remove existing item from the project by selected. You can also save the item available in the window by simple right clicking on item to save choosing save option.

The Elements of User Interface

The Toolbox contains the icons of the controls you can place on a Form to create the application's user interface. By default, the Toolbox contains the pointer icon and the icons of 20 ActiveX controls. To place a control (such as a Command button) on a Form, you first select it with the mouse and then move the mouse over the Form. When the mouse is over the Form, the cursor turns into a cross, and you can draw the control on the Form, just as you would draw a rectangle using a drawing application. The size of the rectangle determines the size of the control.

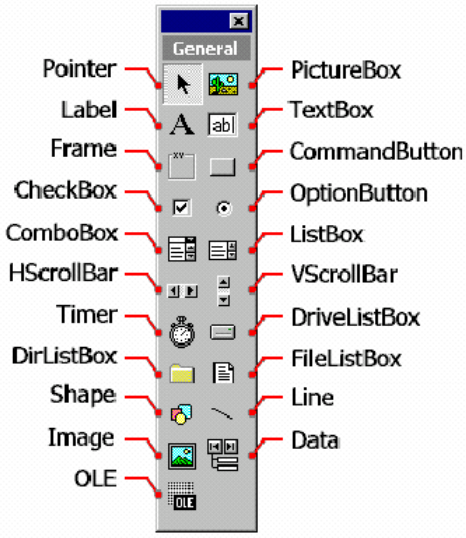


Figure 3: Toolbox Controls

Standard Toolbox Controls

Pointer -The only item in the Toolbox that doesn't draw a control. When you select the pointer, you can only resize or move a control that has already been drawn on a form.

Picture Box - Displays graphical images (either decorative or active

Label Allows you to have text that you don't want the user to change, such as a caption under a graphic.

TextBox Holds text that the user can either enter or change.

Frame -Allows you to create a graphical or functional grouping for controls. To group controls, draw the Frame first, and then draw controls inside the frame.

Command Button -Creates a button the user can choose to carry out a command.

CheckBox -Creates a box that the user can easily choose to indicate if something is true or false, or to display multiple choices when the user can choose more than one.

OptionButton Allows you to display multiple choices from which the user can choose only one.

ComboBox -Allows you to draw a combination list box and text box. The user can either choose an item from the list or enter a value in the text box.

ListBox -Used to display a list of items from which the user can choose one. The list can be scrolled if it has more items than can be displayed at one time.

HScrollBar (horizontal scroll bar) -Provides a graphical tool for quickly navigating through a long list of items or a large amount of information, for indicating the current position on a scale, or as an input device or indicator of speed or quantity.

VScrollBar (vertical scroll bar) -provides a graphical tool for quickly navigating through a long list of items

or a large amount of information, for indicating the current position on a scale, or as an input device or indicator of speed or quantity.

Timer Generates timer events at set intervals. This control is invisible at run time.

DriveListBox -Displays valid disk drives.

DirListBox (directory list box) Displays directories and paths.

FileListBox -Displays a list of files.

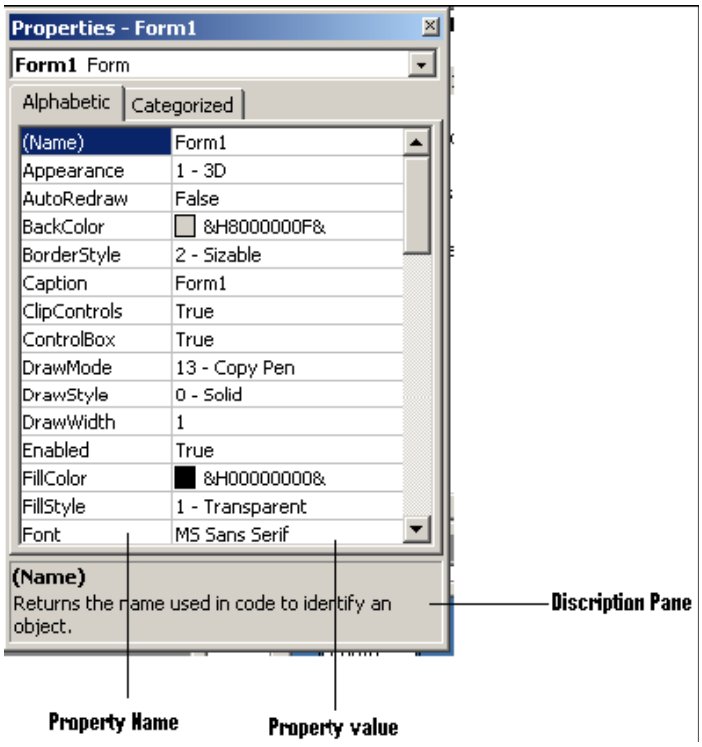
Shape -Allows you to draw a variety of shapes on your form at design time. You can choose a rectangle, rounded rectangle, square, rounded square, oval, or circle.

Line -Used to draw a variety of line styles on your form at design time.

Image -Displays a graphical image from a bitmap , icon , on your form. Images displayed in an **Image** control can only be decorative and use fewer resources than a **PictureBox**.

Data- Provides access to data in databases through bound controls on your form.

OLE -Allows you to link and embed objects from other applications in your Visual Basic application.



Properties Window

Lists the design-time properties for selected objects and their current settings. You can change these properties at design time. The following window displays the properties windows and it displays the form properties.

Elements

Object Box

Lists the currently selected object. Only objects from the active form or designer are visible. If you select multiple objects, the properties common to the objects and their settings, based on the first object selected, appear on the Properties List buttons.

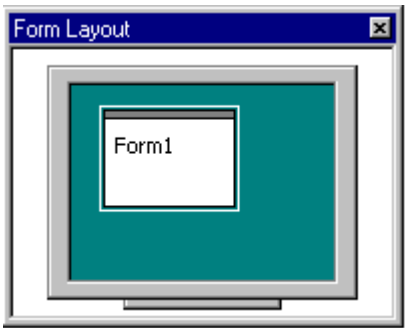
Property window Display two tabs these are **Alphabetic and Categorized**. **Alphabetic** tab - Alphabetically lists all properties for the selected object that can be changed at design time, as well as their current settings. You can change the property setting by selecting the property name and typing or selecting the new setting.

Categorized. tab -Lists all properties for the selected object by category. For example, BackColor, Caption, and ForeColor are in the Appearance category. You can collapse the list so that you see the categories, or you can expand a category to see the properties. When you expand or collapse the list, you see a plus (+) or minus (-) to the left of the category name.

Description Pane

Shows the property type and a short description of the property. You can turn the description of the property off and on using the Description command on the shortcut menu. You can move through the list of descriptions by pressing the ARROW keys.

Form Layout Window



Allows you to visually position your forms at design time.

All forms that are visible in the environment are displayed. When you place your cursor over a form, it changes to a. If you press the mouse button you can position the form where you want it to appear at run time.

When you re-size the Form Layout window, each form is sized relative to the size of your design window. The upper left corner of the client area represents the coordinates – 0, 0 – of the desktop. You can change the resolution using the Resolutions Guides command on the shortcut menu.

You can set the startup position of form as center, center owner manual etc using startup position command on the short cut menu. You can hide and display form layout window using toolbar short cut 

Project Types

Visual Basic is not just a language. It's an Integrated Development Environment in which you can develop, run, test, and debug your applications. Start Visual Basic, and you'll see the window shown in following figure. This is where you are prompted to select the type of project you want to create. With Visual Basic, you can create the following types of applications.

1. Standard EXE:

A Standard EXE project is a typical application. Most of the applications user creates are Standard EXE projects. These are the types of applications you developed with previous versions of Visual Basic.

2. ActiveX EXE, ActiveX DLL:

These types of projects are available with the Professional edition. ActiveX components are OLE automation servers. ActiveX components are basic code-building components that don't have a visible interface and that can add special functionality to your applications. The two types of projects are identical in functionality, but are packaged differently (as executable files or Dynamic Link Libraries).

3. ActiveX Control:

This type of project is also a feature of the Professional edition. Use it to develop your own ActiveX controls. An ActiveX control such as a TextBox or Command button control is a basic element of the user interface. If the ActiveX controls that come with Visual Basic (the ones that appear in the Toolbox by default) don't provide the functionality you need, you can build your own custom controls.

4. ActiveX Document EXE, ActiveX Document DLL:

ActiveX documents are in essence Visual Basic applications that can run in the environment of a container that supports hyperlinking (such as Internet Explorer). These types of documents are not discussed in this book.

5. VB Application Wizard, VB Wizard Manager:

The Application Wizard takes you through the steps of setting up the skeleton of a new application. Modifying the skeleton code created by the Wizard is just as difficult as developing your own application from scratch.
The Wizard Manager lets you build your own Wizard. A Wizard is a sequence of windows that collect information from the user. After the user fills out all the windows, the Wizard proceeds to build an application, install software, or carry out an automated operation for the end user.

6. Data Project:

This is a feature of the Enterprise edition, and it doesn't correspond to a new project type. It's identical to the Standard EXE project type, but it automatically adds the controls that are used in accessing databases to the Toolbox. It also adds the database ActiveX Designers to the Project Explorer window. The ActiveX Designers are visual tools for accessing and manipulating databases and generating reports.

7. DHTML Application:

VB6 allows you to build Dynamic HTML pages that can be displayed in the browser's window on a client computer.

8. IIS Application:

VB6 allows you to build applications that run on the Web server and interact with clients over the Internet with the Internet Information Server.

9. AddIn

You can create your own add-ins for the Visual Basic IDE. Add-ins are special commands you can add to Visual Basic's menus (they usually appear under the Add-Ins menu). To do so, select this type of project.

10. VB Enterprise Edition Controls

This is not a new type of project. It simply creates a new Standard EXE project and loads all the tools of the Enterprise edition of Visual Basic.

1.3 Designing the User Interface

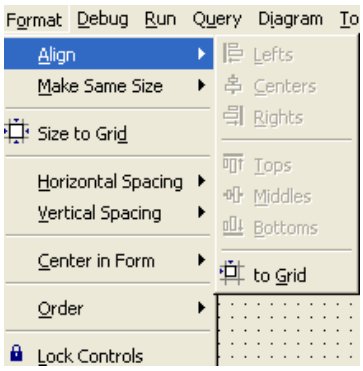
Designing the user interface consist of drawing the elements on forms. We are taking example of login form.

Procedure:

1. Open a new standard EXE project in VB 6.0
2. Insert 2 labels, 2 text box, 1 command button
3. Change the following property of label1 as
   Caption= “User Name”
   Font= customize
   Backcolor=customize
4. Change the following property of label2 as
   Caption= “Password”
   Font= customize
   Backcolor=customize
5. Set following property to textbox1 and textbox2
   Text= EMPTY
   Change the property of textbox 1
   Name=“username”
   Font= customize
   Backcolor=customize
   Change the property of textbox 2
   Name=“Password”
   Font= customize
   Backcolor=customize

6. Set following property to form
   Caption = Login Form
7. Set following property to Command Button and enter the following code on command button
   Caption = Login

Aligning the controls :



- After adding the elements (controls) on form we have to property organize the form because application user interface is the visible part.
- The visual basic IDE provides numerous commands for aligning controls, which are available on format menu.
- Align command align the edges or middles of the selected control to left, right or center, top, middles, bottom et.
- Make same Size command make all selected control same size.
- Size to grid : Corners of selected control will align with nearest grid point.
- Horizontal Spacing : It controls the horizontal spacing between selected controls.
- Vertical Spacing : It controls the vertical spacing between selected controls.
- Center in form : It centers the selected control horizontally or vertically in the form.
- Order : It changes the order of the selected control by moving them in front or behind other controls.
- Lock Control : This command allow you to select a number of controls on the form and lock them. You can change the properties of locked control but you cannot move them.

Running the application :

- Next process is to run and test the application. To run the application choose Run >> Start or Press F5.
- You may wondering that without entering a single line code, we create a application. It happens in Visual Basic because it uses visual tools to build a large section of the application.
- The user interface can be built almost entirely with point and click operation.
- To stop the application choose Run>>End.
- As you can see, the Visual Basic IDE is more than a program editor. It's an integrated environment in which you can design and run your application.

Programming an application :

Designing user interface consist of drawing the elements on forms, aligning it properly and formatting it as per the requirement.

Designing is done in design view. Design is the visible part to the user.

Programing is the another angle of the application development.

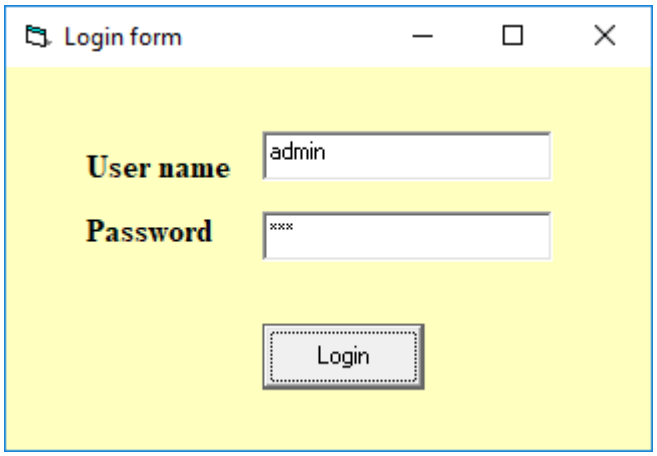
In the Designing user interface we take the example of login form. In the login form we add two labels, two text box and one common button.

We are going to create a code on command button. (login button).

Double click on the command button and insert the following code on command1_Click() event.

CODE:

```
Private Sub Command1_Click()
Dim user As String
Dim pass As String
user = "admin"
pass = "rvm"
If (user = username And pass = password)
Then
MsgBox "welcome"
Else
MsgBox "Sorry"
End If
End Sub
```



After completion of coding Run the program by pressing F5 key.
Login form is displays as given below :

Now enter the username as "Admin" and password as "rvm" and click on login.

If username and password is matched then following message box is displayed.

If user name and password is not matched, then message box is displays “sorry” message,

Visual Basic IDE is an integrated environment in which you can design a user interface form and enter a code for it and run your application.

Visual Development and Event-Driven Programming :

Forms and controls are the basic elements in the user interface of any windows application. In visual basic, these elements are called as *objects* because are manipulated like objects.

Objects have properties and methods, and they react to external events as does any physical object. Normally properties are set when an object is created. Most control properties are set when the object is created. But you can change a property later by assigning a new value to it. VB assigns a default properties to every new control you place on form. Like If you insert Command button it gives name as command1, commad2 etc.

Visual development and Event-Driven programming includes three basic things

1. properties of objects
2. Methods
3. Events

1. **Properties of objects:**

Following are the most common properties apply to most objects.

Name:

Name used to identify object. It is the name by which the object is referred to in your code. Visual Basic by default names the forms Form1, Form2, Form3, etc. three letter prefix for form names is ***frm***

Appearance:

Return or set whether or not an object is painted at run time with 3D effects. There are two values of this property 0-Flat & 1- 3D. the default vale is 3D.

BackColor:

This property is used to sets the form background color.

BorderStyle

The BorderStyle property determines the style of the form’s border and the appearance of the form; it takes one of the values shown in following Table .You can make the form’s title bar disappear altogether by setting the form’s BorderStyle property to FixedToolWindow, the ControlBox property to False, and the Text property to an empty string. However, a form like this can’t be moved around with the mouse and will probably disturb users.

Table 3.2: The FormBorderStyle Enumeration

Value Effect

None Borderless window that can't be resized; this setting should be avoided.

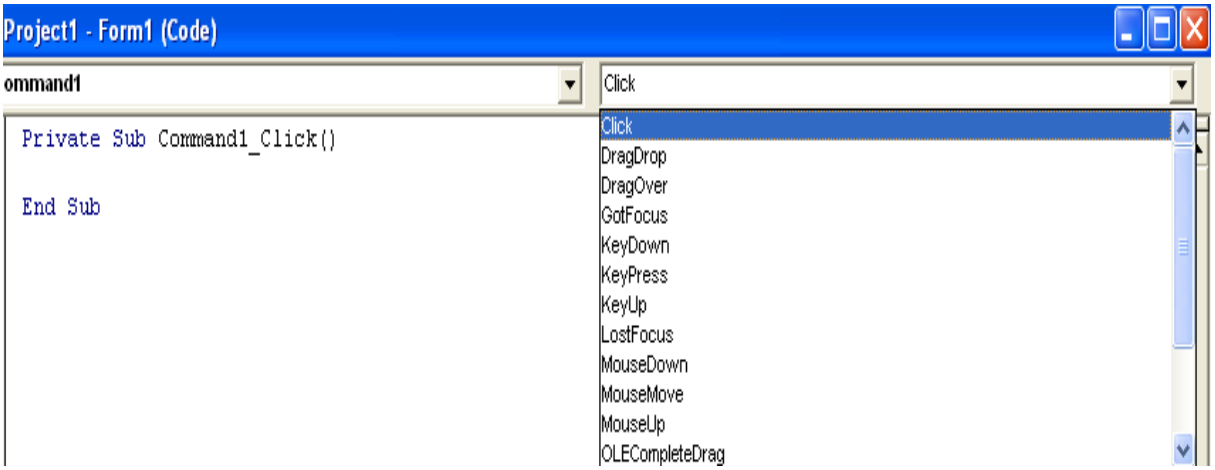
Sizable (default) :	Resizable window that's used for displaying regular forms.
FixedDialog:	A fixed window, used to create dialog boxes.
FixedSingle:	A fixed window with a single line border.
FixedToolWindow:	A fixed window with a Close button only. It looks like the toolbar
	displayed by the drawing and imaging applications.
SizableToolWindow:	Same as the FixedToolWindow but resizable. In addition, its caption font is smaller than the usual.

Caption:
It is the text that appears in the title bar of form. A caption can be changed at runtime. The caption must be informative and meaningful.
Font:
This property is used to sets font type, style, size.
ForeColor:
This property is used to sets color of text or graphics.
Height:
This property is used to sets the height of the form.
Left:
This property is used to sets the distance from left side of computer screen to left side of form.

Events

Event determines the control's reactions to external conditions. Controls recognize events, but your application handles them. A command button will recognize that it was clicked, but it won't react to the event unless you provide some code. It means you must tell visual basic what to do when the user clicks the specific command button. This subroutine or code executes each time the control is clicked.
To write an event handler for a control, follow these steps.

1. Switch to the code window or double click the control for which you wat to write the event handler.
2. Following is the view of code window. You will see two drop-down lists.
    a. The first contains the names of all control on the form. Select the control for which you want to write an event handler.
    b. The second list contains all the events the selected control can recognize.



3. The combination of the cotrol's name and event's name is unique and is the name of the event
4. Visual basic looks for the subroutine made up of the ame of the control on which the event took place and the name of the event. Two most common events are
    a. Mouse Event
    b. Keyboard Event.
    c. Focus

Mouse Event:

Most of the elements of the user in can be manipulated with the mouse, and programming mouse events is your number one job as a VB programmer. The form primarily acts as a container for other controls, but it does support events. That is it can respond to some user interactions. Some commonly used events are as follows:

Form Events:

Event Description

Click Event executed when user clicks on the form with the mouse.
DblClick Dblclick event triggered when user double clicks on form.
Activate Activate event is triggered when form becomes the active window.
Deactivate Deactivate event is triggered when form becomes the deactivate window.
Load Load event occurs when form is loaded. This is a good place to initialize variables and set any run time properties.
Unload Unload event occurs when user close the form.
Terminate This event is used to close or terminate the form. All the memory that was held for the form variables are released
Resize This event is occurred when the user resizes form.
MouseMove This event is occurred when the user mouse moves on the form.

Keyboard Events:

Keyboard events are generated by keystrokes, usually, you must program the keyboard events for the controls that can accept text. You must provide code for the keyboard events of controls that can be manipulated with both the mosue and keyboard.
Keypress Tis event is occurred when the user presses key from the keyboard.
Besides these events there are some events these are initialize keyup, keydown etc.

Focus:

A fundamental concept I developing user interface that comply with the windows standard is the focus. Focus is the ability of a control to receive user input via the keyboard . When an object has the focus, it can receive input from a user. Suppose you have a form with two textbox controls on it. At any time, only one of them can accept input. The control that can accept input is said to have the focus. **GetFocus** and **LostFocus** are two releated events. Whe the focus is moved from one control to the other, the first control receives the LostFocus event, and after that the second control receives the GotFocus event.

Methods:

Objects have methods too, which are the actions they can carry out You can think of methods as the actions of an object. For ex. Methods of your VCR are the play, Fast Forward, Rewind, Pause, and Record buttons. After you press oe of these buttons, your VCR can perform without any further assistance from you.

Similarly Form object, knows how clear itself, and you can invoke the CLS method to clear a form. A form also knows how to Hide itself, an action that you can invoke from within your code with the hide method.

CLEAR, MOVE, AddItem, RemoveItem are some methods.