Introduction to VB

The evolution of VB took place while developing & designing windows based application. Programming windows based application used to require hours of hard work for creating interface only. Visual Basic was introduced in the year 1991. Basic idea of introducing VB to the world was to provide a rapid application development environment to the developers. This environment was facilitated so that developers could put in all their concentration in Application's design rather than how to design. Where other languages take hours of work for creating an interface, the same application can be created in few hours using Visual Basic.

Visual Basic (VB) is an ideal programming language for developing sophisticated professional applications for Microsoft Windows. It makes use of Graphical User Interface for creating robust and powerful applications. The Graphical User Interface as the name suggests, uses illustrations for text, which enable users to interact with an application. This feature makes it easier to comprehend things in a quicker and easier way.

Coding in GUI environment is quite a transition to traditional, linear programming methods where the user is guided through a linear path of execution and is limited to small set of operations. In GUI environment, the number of options open to the user is much greater, allowing more freedom to the user and developer. Features such as easier comprehension, user-friendliness, faster application development and many other aspects such as introduction to ActiveX technology and Internet features make Visual Basic an interesting tool to work with.

Visual Basic (VB) is an event-driven programming language. This is called so, because programming is done in a graphical environment unlike the previous version BASIC where programming is done in a text only environment and executed sequentially in order to control the user interface. Visual Basic enables a user to design the user interface quickly by drawing and arranging the user elements. Due to this spent time is saved for the repetitive task.

An Introduction to VB- As the name suggest, programming on Visual Basic is programming with the vision, that is while writing the program you can see how your program will look during run time. This is an advantage over other programming language. Micro soft VB is a part of micro soft visual studio 6.0 that include a variety of development tools. These development tools are integrated together to provide programmer an easy application development environment. This environment develops application quickly which is called rapid application development.

VB was introduced in 1991 to provide a G U I base application development. Being a G U I based application development application platform, it minimizes a large amount of programming works and make application development easy.

The two main themes in developing applications through VB are:
1 Visual design– The basic drawing capabilities are used to design the user inter face of the application. If you have used drawing application you can design users interface with VB.

2 **Event driven programming-** When you program in VB, you must first decide how the application interact with user. In other words you must decide how each control reacts to user action such as mouse click etc. and you must program for the reaction. This is called event driven programming because the application does actions on the basis of events caused by the user determining the flow the program execution.

Important Features of Visual Basic (VB)

Full set of objects - you can 'draw' the application
Lots of icons and pictures for your use
Response to mouse and keyboard actions
Clipboard and printer access
Full array of mathematical, string handling and graphics functions
Can handle fixed and dynamic variable and control arrays
Sequential and random access files support
Useful debugger and error-handling facilities
Powerful database access tools
ActiveX support
Package & Deployment Wizard makes distributing your applications simple

Hardware & Software requirement for Visual Basic:
Processor:
Any processor of Pentium series with at least 150 MHz processing speed can run VB 5.0/6.0. Generally processor with 300 MHz speed is recommended.
Memory Size:
At least 2 GB of disk storage space is required for VB5.0 with 16 MB RAM. Recommended is 64 MB RAM with 4.3 GB Disk space for VB 6.0.

Operating System:
Windows 95 or above version of windows family is required for running VB 5.0/6.0.
Visual Basic terminology is very vast. It is difficult to select few terms to explain because all terms are quite often used & have their own importance.

Knowledge of few terms like:
Objects & controls
Objects in any language supporting OOPs are the variables belonging to a particular user defined data types.
These user defined data types are known as class. In VB controls on the tool box represent individual classes and are called objects, when placed on a form. A command button, textbox, etc & even the form itself are instances of

some class. These are called as an object as soon as they are included in the project.

Properties
A property is a changeable characteristic of an object e.g. color, size & caption, etc. various valid values can be set to properties of an object.
Properties are named using nouns e.g. Text, Back color, etc.
Properties can be categorized into 3 categories:
Design-time
Run-time
Design & Run-time

Methods
Methods are the actions that can be performed on any object or by the object. They may or may not change the state of the object. Generally they are named using verbs, such as open, hide, move, etc. properties of an object can be accessed in any order while methods must usually be executed in a specific order.

Events
Events in VB can be called as the happenings with the objects while an application is running. These happenings can be caused due to the user's action on keyboard, mouse or even by the application itself

Forms
Form in VB is actually the workspace where you design the visual layout of the form & the controls that lie on it. This is actually the window of your application in run-time. VB IDE displays forms & code window separately. The code window is actually the space where you write your codes & can visually classify between different procedure & functions.
Functions:
Procedures:
is important to begin the process of learning this language.

Element of IDE of Visual Basic:
It includes following elements
Tool bar, menu bar, tool box, project explorer windows, properties window, form lay out window form designer

Constant:
Variable that do not change their values during program execution are called constant. Constants evaluate the same value each time. By the followings reasons constant are preferred over variables.
Syntax - Const <Constant Name> [As type] = <value>
e.g. Const dimension as single = 22.41

Variable: in a programming language variable store value and during a program execution that is temporary store data. A variable contains a certain type of data that can be modified during program execution

Variable naming rule in Visual Basic:
It must start with a letter. Its maximum length can go up to 255 characters. It can not contain special character. A variable name can not be same as the Visual Basic key word.

Declaring variables: variable are declared in a programming language to assign a name to the variable of type of data you want to store in it VB variables are declared by dim statement.
Syntax –dim <variable name > [as <type>]
Ex. Dim num as integer
Din num, name as string.

Print statement:
It is used to display information within currently active form starting from the upper left corner. This statement consists of a key word print followed by a list of out put items. The out put item can be numeric, string, constant or expression

Syntax – Print [“message “] [<variable>]
Ex. print “abc”

Print with comma:
It allows printing of variable or message column wise on the screen numeric item are printed left most position and strings are right most position

Print with semicolon:
It prints data items in the same line with no spaces between them
Ex print 2; 3; 4; 5;

Print with tab: it is used to make columns in tabular forms. Tab allows the computer to insert desert desired no of spaces before printing a matter.

Print tab(20) “we”; tab (30) “love”; tab (40) ”India”;

Operator in Visual Basic:
An operator is used to transform one or more values into a resultant single value. As other programming value Visual Basic also provide a lot of operators
Arithmetic operators-- +, - , /, *, \

Num1=1000.234
Num2=10.2364
Result =num1\num2

^-exponent—2^3=8

Mod (modules)—12 mod 5=2

Assignment -- =

Comparison operator -- <, >, >=, <=, <>.

Logical operator –
And operator 1<2 and 3>4 =>0
Or operator 1>2 or 2<4 => 1
Not operator not 2<4 => 0
Concatenation & and + are used to join two string expression.

```
A= "34"
B= "54"
Print A+B
Or
Print A&B
```

Understanding advantage of Visual Basic:
VB applications are event-driven. Event-driven means the user is in control of application. The user generates a stream of events each time he or she clicks with the mouse or presses a key on the keyboard.
VB supports the principles of object-oriented design. This means that you can compartmentalize different aspects of your application as objects and develop and test those objects independently of the rest of the application.
Microsoft has designed VB to be a complete Windows application development system. This means that your windows applications will look and behave like other windows programs.

Visual Basic Development Overview:
Design & build user interface
The basic component of user interface is form. A form is a window that contains several different objects. All the objects in the form are called controls. The labels, text boxes, command, buttons, and other controls have been placed on this form by its designer.

Writing code that responds to events:
VB supports principle of event-driven programming. Events are generated by several different sources. One of the nice things about event-driven development with VB is that you need not respond to every possible event that might occur. You write code only for meaningful events in the application you are creating.

Creating and calling other procedures as needed:

In addition to event-driven code behind the forms, most applications include a certain amount of code in freestanding, public modules that are part of the VB project. These modules are not connected to any of the forms in the application. By default, the code in public modules is available throughout the application and can be called from any procedure behind a form or contained in some other freestanding code module.

Testing and debugging:
VB code tends to be complex. Even a simple project might have several thousand lines of code behind its forms and in public modules. This code might support dozens of different features, retrieve data, and perform complicated calculations. Problems with completed applications go far beyond oblivious syntax and logical errors in the VBA code. The last step in creating a VB application is to thoroughly test and debug the user interface and code behind it. Even the most carefully designed and implemented code can contain errors and problems. If you are lucky, the errors in your code will be obvious and easily fixed.

Converting to runtime version:
As you work with VB to produce your windows application, you create several different files. An .FRM file contains the specification and code associated with a single form with the applications, and .BAS file contains only VB code. After you satisfied with the design and operation of your VB application, you compile it into a distributable version. During the compilation process VB combines the information contained in the form files and code modules into a single executable file with an .EXE extension. The .FRM and .BAS files remain on the disk to enable you to continue making changes and improvements to the application.

Preparing a distributable set of files:
VB includes the Package and Development Wizard, which eases the process of building distributable set of files. It produces and installation set that looks and behaves like the installation routines for any other windows application. The Package and Development Wizard is smart and includes all the support DLLs and other files required by your application.

Basic controls:
Labels:
Label control is used to display the information which is read only. A label control is similar to text box in the sense that both display text. The only difference being that label control provides a read-only interface where as you can change text in the textbox.
Textbox:
Generally almost each and every application uses a textbox to accept an input from the user. Textbox has all the required properties and events that can be used in an application. The textbox control has the text property as the default property. You can add scrollbars to it, set font for the text, set its back color. Multiline property enables you to allow typing in multi lines.

List box & Combo box:
A list box is an ideal way of presenting users with a list of data. A list box control displays a list of items, and allows to select one of them.
A scroll bar is automatically added to a list box control if the number of items exceeds its size. The user can't edit the data in the list box.
A combo box is simply a drop down list box. The combo box combines the features of a textbox and a listbox, both. A combo box allows you to select an item fro the list and even add a new item.
Radio buttons & checkboxes:
Option buttons are also called as radio button. Option buttons always work together. When user chooses one button, all the other buttons in the group are turned off. Thus if one form contains more than one group, then it should be kept in different frame control, to identify the group of option buttons separately.
A checkbox control is rather similar to an option button. The value property of both the controls is tested to check the current state. Check boxes are valid as a single control whereas a single option button is probably counts – intuitive. Checkboxes are not mutually exclusive.

Timer control:
This control is similar to the time bomb which explodes after a certain time. Only difference is that it explodes (execute the code) continuously with a fixed time gap unless it is stopped. Timer controls should not be overused, as a window restricts all the applications running at one time to 16 items.

There are 26 basic tools (including the above stated) in the tool box:

Pointer
Picturebox
Label
Textbox
Frame
Command button
Checkbox
Option button
Combobox
Listbox
Hscrollbar (horizontal scroll bar)
Vscrollbar (vertical scroll bar)
Timer
Drivelistbox
Dirlistbox
Filelistbox
Shape
Line
Image
Data
Commondialog
Grid

Dblist (data-bound list box)
Dbcombo (data-bound combo box)
Dbgrid (data-bound grid)
Ole container

Adding /Removing custom controls to the toolbox:
Custom controls are basically third party controls you buy from an outside vendor, e.g., Sheridan tabbed controls. You can create your own custom controls in Visual Basic.

Coding in Visual Basic:
For each and every event, we have to provide a code on the entire object. Double click on the object and a new window called code window opens up. Now we have to provide the code in proper syntax.
The start up form— an application form has more than one form. When an application starts the main form is loaded you can select the form displayed when your application start. For this purpose, choose
Project>> Properties.
In start up form list select the desired form name and click ok
Loading showing hiding forms: there may be three statement and method for displaying forms

1- Not loaded – the form loads from disk and does not take any resources.
2- Loaded but not showing—the form is loaded into memory take a required resources and is ready to display.
3- Loaded and showing – the form is shown and the user can interact with it.
4- Loading and unloading a form-Load and unload statement are used to load and unload a form respectively.

Syntax – To load a form
Load < form name>
Syntax- To unload a form
Unload <form name>
Ex. Unload form1
Unload me

The form name variable is the name of form to be loaded or unloaded
Load statement does not show the form you have to call the form's show method to display it on the desktop.
Once a form is loaded it take over the required resources so you should always unload a form that is not longer needed.
When a form is unloaded the resources it occupies are returned to the system and can be used by other forms and or application.

Showing Forms:
To show a form the show method is used. If a form is loaded but not shown the show method brings the specified forms on the top of every open window. If the form is not loaded show method loads and then displays it.

Syntax- <Form Name>. Show [Mode]
0 modules (default), 1- Module,
The <Form Name> Variable is forms name and optional argument mode determined the form will be modeless or model. The modeless forms are normal forms they interact with the user and allow the user to switch to any other form of the application.

A model form takes control of the application and could not load the application process unless the form is closed. A model form must have close button something like that to close it.

Hiding Forms – If your application usage many forms you may want to hide some of them to make space on the desktop for other.
Syntax <Form Name>.Hide
(me.hide)
Scope of Variable- The variables also has a scope. The Scope of a variable is the sections of the application that can see any manipulation of the variables. The Scope of a variable determines the section of an application where the variable can be use. A variable can be defined with following any of the three scopes.

Local- Variable declaration within a function or procedure or local to that function or procedure is called local variable. Private keyword is used to declare local Variable. By default the scope of all variable is local.

Form Wise/Module Wise- These variables are available to all the functions present in a form or module. They also considered as private variable.

Global- Variable that can be accessed by the entire application comes under global Scope. To declare a global Scope variable use public keyword in place of dim.
e.g.- Public num as integer.

These are defined in general declaration area of the code. A run time error is generated if you try to redefine public variable in a procedure.

User defined data type- VB allows you to create your own data type containing one or more elements such as structure.
Syntax: type <Variable Name>
<Var 1> as type
<Var 2> as type.
End type.

Decision making statements:
Control Structure: VB provides basically two control structures
1. If Structure 2. Select Structure

If Structure- It can be in any form
a. If …..then
b. If …….then else
c. Nisted if ……….then……..else
d. Multiple else if…

If-then-else:
VB provides if then else statements for decision making. E.g. If condition is true then if part execute otherwise else part will be executed. If the test fails and else part is not mentioned then the processing skips to the next statement.
Example: if x =0 then msgbox "zero" else msgbox "non-zero"
More often multiple statements' are to be processed if a certain condition is true or false. All such statements are thus blocked in if…end if statements.
Multiple conditions can be checked in one if statement using AND keyword.
One out of many conditions can be checked for true using OR keyword.
You can also use nested if structure i.e. if-then block within if – then blocks.
Nested if statements can be replaced by using if – else if – end if structure.
IF condition then
[Statement1]
[Statement2]
.
.
.
End If

IF-ELSE-IF Structure
If [condition1] Then
[Statement1]
[Statement2]
.
.
.
Else If [Condition2]
[Statement1]
[Statement2]
.
.
.
End If

Immediate if:
The iif() function provides immediate if for quick decisions. The iif() function returns one part out of two, after evaluating the given expression. The entire expression consists of two parts the if part and else part.
Example:
Result = iif div >=60, "first", "second"

Select case statement:

Execute one out of several groups of statements, depending upon the value of an expression.

Syntax:

```
Select case <test expression>
        Case expression list
                [statements]
        Case expression list
                [statements]
        ----------------------
        ----------------------
        Case else
                Statements
        End select
```

The case else clause is used to indicate the statements to be executed if no match is found between the test expression and in expression list in any of the other case selections. Execution continues at the statement following end select. Case Else statement should be added in the select case block, in order to handle unforeseen test expression value.

Comparison can be done using logical operators while testing the expression in each clause.

Iterations: loop structure

Loop Statement- Loop statements allow you to execute one or more lines of code repetitively VB Supports following loop structures

Do……loop

While…….wild

For……next

Do……Loop- The do……..loop executes a block of statements for as long as a condition is true VB evaluate on expression if it is true statement on executed there are two variations of Do……..loop statement. A loop can be executed either while condition is true of until the condition becomes true. These two variations use key words while & until. There syntaxes are

1) Do while <Condition
Statement ……..
Body …….
Loop

2) Do……..
Loop while <Condition

3) Do Untill <Conditions
Body
Loop

4) DO
Loop While <condition>

```
Dim num as integer num = text1 text
Label1-visible = false
Text1- visible = false
I = 1
Do while i<=10
t = uum
```

For –Next- Unlike do…….loop, the for …..next loops required that you have to vow how many times the statements in the loop will be executed.
The for next loop user go a variable that in creases & decreases in value during lath repetition of loop
for <variable> = <start> to <end> statement

While……..Word:- The loop execute a back of statements while a condition is true
Syntax- while <Condition> statement block word

Nested Loops:- It is common to intend the bodies of nested loop structures to make the program larier to read the outer loop scans each row of the array & the liner loop scans each coldams of that row the outer loop prints.

VB provides loop structures to perform iterative operations. Repeating action can be implemented in two ways;

- Determinate loops
- Indeterminate loops

Determinate loops:
Repeat the operation for a fixed number of times.

Indeterminate loops:
Continues until you reach a specific predetermined goal, or continue until certain initial conditions have changed.

For—next:
Using this loop structure enables the program to execute a block of statements a fixed number of times. At the beginning of the loop, specify the starting value and ending value for the counter. The program executes till the counter reaches the ending value.
e.g.

```
for x = 1 to 10
text1.text = x
next
```

Do—loops:
This loop structure enables the program to execute a block of statements an indefinite number of times based on a condition. Each time the code loops, the condition is tested again. When the condition finally becomes false, the loop terminates.

Do-while loop:
This executes a code indefinitely till some condition is True or False.
General syntax:
Do while some condition is true
Execute these statements

Loop
e.g.
var = 10
Do while var <> 0
print "it is rainging"
var = var + 1
Loop

While-wend statements:
Execute a series of statements as long as a given condition is true.
Syntax:
While condition
[Statement]
Wend

For each – next loops:
For each—next loops are the loops that iterate through all the elements of an object's collection. A collection is a group of objects. Each object has its own identify in the collection. For instance, you might want to hide all the controls on a form then you can use the following code.
Dim control as object
For each control in controls
Control.visible = false
Next control
In the above example the collection controls contains all the controls placed on a form. If there are no objects in the collection, for each—next loop is automatically terminated.

With – end with statements:
With – end with statements are used to avoid writing the name of the object again and again while referencing more than one time. Instead of referring repeatedly to the same object, you can identify the object and then use with – end with statement to perform a series of action on it.

Example:
All the properties of textbox can be referenced as
With text1
.text = "howard"
.height=1400
.width=1800
.forecolor=rgb(0,256,0)
End with

Arrays:
Variables that can hold more than one values, are called as arrays. Generally arrays hold the values of similar data types but can hold different values of different data types if they're declared as variant type. The lowest subscript of arrays depends upon the option base statement. However it is 0 by

default. In such a case the last element of an array of four elements will have index number 3. Arrays can be declared using all the keywords that can be used to declare variables.
Syntax:

Dim arrayname(subscript) as type
Example:
Dim name(10) as integer

Double-dimensional or multidimensional arrays:
You can declare arrays with more than one dimension. Arrays with two dimensions are called as double dimensional arrays and the arrays with more than two dimensions are called as multidimensional arrays.
Syntax:

Dim arryname(lowest subscript to highest subscript, ----)
Example:
Dim grid (1 to 10, 1 to 10)
grid will have 10 rows and 10 columns and total number of cells (elements) will be 100.

Dynamic array:
This has been noticed that generally you are not aware of the total number of elements to make an array while programming. For example your application stores the records of employees, but the number of employee is unknown. In such a case you must declare an empty array. An array that starts life with no elements is called as dynamic array.
Array or lists whose size can be changed on the fly are known as dynamic arrays or dynamic lists. You can resize an array using redim statement. VB6 has introduced the new concept of dynamically resizing the target array to the size of the source array. To declare a dynamic array you must ignore the size of the array in declaration statement.
Dim sizeunknown () as integer
The size of the array is not specified, and thus sizeunknown will be a dynamic array whose size can be changed later in the program.

Re-dimensioning an array:
The keyword used to re-dimension an array is Redim. The Redim statement should be accompanied with preserve keyword in order to retain the initial values of the array.
Example:
Redim preserve sizeunknown(10)
This re-dimension feature of VB can help you to develop a link-list without the use of pointers.

Collection:

The new keyword tells us to create a new collection. The collection objects have three methods (add method, remove method, item method) & one property (count property).

Adding to a collection: The add method is used to add new items to the collection

<Collection Name>. add <value>, key [before/after]

Removing an item for a collection: It removes a method and deletes an item from a collection.

Example: To remove student jaya from your collection
College.remove "Jaya"
College.remove 2

Returning items in a collection: The item methods return the value of an item in a collection. The index can either be items position or its key in the collection.

M=College.item("Jaya")
M=College.tem(4)
M=College("Jay")

Counting a collection: The count properties return no of items in the collection for example to find out the no. of items in the collection college count.

Processing a collection item: To scan all the item in the collection of VB provides for each……next structure

Syntax- For each <item> in <collection>
Statements
Next.

Procedure: In VB, the theme is to provide an easy environment to develop application by packing a complex program into small logical units. Hence an application is made up of small, self contained segments & logical units & each segment is responsible. It's a specific task. These segments are called procedure.

Hence a procedure is set off instructions that work as a unit to perform a specific task.

Types of procedure: Procedures are written in the code window of VB IDE VB provides basically two types of procedures-

- Subprocedure/Subroutries.
- Function Procedure.

Subprocedure: A subprocedure is a block of statement that carries a well define task. subprocedure doses not return any value. The block of statement is head between sub ……..endsub statement & can be invoked by its name. All the event procedures in Visual Basic are called as subdirectories. It is possible to exit a sub routine permanent ally with the exit statement.

Syntax: [Private/Public] [static] sub <procedure name>

([<arguments>])
Body of procedure
End Sub.

Visual Basic supports two types if subprocedure
1) Event Subprocedure 2) General Subprocedure

Event Subprocedure: These procedure are attached with event you can write event write procedure for controls and for forms.
Event procedure also called event handless. When an object in VB recognized that event it automatically invokes the event procedure corresponding to that event. It is a self contained place of code that is executed when needed.
Syntax :
Private Sub < Control Name>_<Evenet Name> ([<Argument List>])
Body of procedure
End Sub.

Control Subprocedure: A general procedure helps the application how to perform a specific task if is not related to any specific event. One's a general procedure defined. It must be specifically invoked by the application unlike the event procedure that remains ideal until they called upon the respond to event triggered by the user. General procedures are written to avoid that application of code in modules.
Example: If some place of code is common in more than one event procedure then you can put it in a separate procedure that is general procedure.

Access Specifier:
There are two access specifiers called private & public. These specifiers are used to determine the scope of subprocedure. By default, subprocedures are public means they can be called from any where in the application.

Static:
This keyword is used to ration value during the life time of subroutine irrespective of how many times that procedure is called.

Sub:
This keyword tells the compiler to define a procedure.

Procedure Name:
Name of the procedure follows following rules

- First character should be any alphabet or under score only.
- Special characters like + - * / . % spaces etc are not allowed anywhere.
- Procedure name should be indicative.
- Procedure name must be preceded by ().

Arguments:
It includes list of arguments based on the events when a procedure is called.

That general procedure can be called by any event procedure in your application for example-

```
Sub show _ date ( )
        Msg Box  _ date ( )
End sub
Private sub command 1- Click ( )
        Show- data
End sub.
```

Function Procedure:
A function is similar to sub procedure but it returns a result.
The function statement are written function ……..End function.

Creating Procedure:
A procedure can be created by just typing it in the code window you can also create procedure by
tools >> add procedure

Statement:
Code written within or outside of the procedure or function without comment character is called a statement.

System defined function:
VB provides a large set of built in function such as scr, chr, date, abs, etc. that can be used to perform a specified task.

```
Syntax-
[Private/Public] [Static]
Function <Function Name> ([<Argument>])
[As type]
Statement
End Function.
```

Example –

```
Function   absl (num or double) as double
If num>=0 then
Absl = num
Else
Absl=-num
End if
End function.
```

Input box Function: Input box used when user asks for input by clicking a button

Syntax- input box (prompt [title]
[default] [x position] [y position]