# Abbreviation and Definition:

|  |  |
| --- | --- |
| .Net Framework | The .NET Framework is a major feature of Visual Studio that is shared by Visual Basic, Microsoft Visual C++, Microsoft Visual C#, Microsoft F#, and other tools in Visual Studio. It’s an underlying interface that becomes part of the Windows operating system itself, and it is installed on each computer that runs Visual Studio programs. The key components in the .NET Framework are the common language runtime (CLR) and the .NET Framework class library, which includes ADO.NET, ASP.NET, Windows Forms, and Windows Presentation Foundation (WPF). |
| argument | A value or an expression passed to a procedure or a function. |
| build | Build an executable file; debug build(slower)/release build |
| carriage return character | In programmer terms, a carriage return character is the equivalent of pressing the ENTER key on the keyboard. |
| Class | A blueprint or template for one or more objects that defines what the object does. |
| compile | To change (computer programming instructions) into a form the computer can understand and use |
| event procedure | A block of code that’s executed when an object is manipulated in a program. |
| event-driven | Code remains idle until called upon to respond to some event (button pressing, menu selection, ...) |
| Focus | Insertion point, blinking sign with the shape of “I” when editing text |
| GUI | Graphic User Interface |
| HTML | Hypertext Markup Language |
| IDE | Integrated Development Environment |
| italic | Inclined font |
| Method | [Format: Object.Method(Value)]; where *Object* is the name of the object you want to work with, *Method* is the action you want to perform, and *Value* is zero or more arguments to be used by the method. |
| namespace | Or called class libraries; A hierarchical library of classes organized under a unique name, such as System.Windows or System.Diagnostics. |
| Projects/  Solutions | A project file contains information specific to a single programming task. A solution file contains information about one or more projects. |
| Property | [Format: Object.Property=Value]; where Object is the name of the object you’re customizing, Property is the characteristic you want to change, and Value is the new property setting. |
| Scope | Value or memory for a variable/constant |
| syntax | The rules of construction that must be used when you build a programming statement |
| TOC | Table Of Contents |
| UDT | User-defined type |
| UI/form | User Interface |
| URL | Uniform Resource Locator |

# Signs, Symbols and Operators:

|  |  |
| --- | --- |
| ‘ | Used to add comments, not be executed by program |
| & | ampersand; string concatenation operator (combination)/define access key for menu |
| \_ | line continuation character |
| = | assignment operator |
| () | parenthesis (plural: parentheses) or bracket |
| ^ | exponentiation (raising to a power) |
| Mod | Remainder division |
| / | Division |
| \ | Integer (whole number) division |
| \* | Multiplication |
| + | Addition |
| – | Subtraction |
| = | Equal to |
| <> | Not equal to |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal to |
| <= | Less than or equal to |
| And | If both conditional expressions are True, then the result is True. (Short-Circuiting: AndAlso) |
| Or | If either conditional expression is True, then the result is True. (Short-Circuiting: OrElse) |
| Not | If the conditional expression is False, then the result is True. If the conditional expression is True, then the result is False. |
| Xor | If one and only one of the conditional expressions is True, then the result is True. If both are True or both are False, then the result is False. (Xor stands for exclusive Or.) |

• (Shortcut Syntax: for the operators like “+, -, \*, /, \, ^, &”, “X += 6” equals to “X = X + 6”.)

• (Order of Precedence of Operators: from high to low, (), ^, - (negative), \*/, \, Mod, +-)

# Variable Type:

|  |  |  |  |
| --- | --- | --- | --- |
| Data Type | Size | Range | Sample Usage |
| Short | 16-bit | –32,768 through 32,767 | Dim Birds As Short  Birds = 12500 |
| UShort | 16-bit | 0 through 65,535 | Dim Days As UShort  Days = 55000 |
| Integer | 32-bit | –2,147,483,648 through  2,147,483,647 | Dim Insects As Integer  Insects = 37500000 |
| UInteger | 32-bit | 0 through 4,294,967,295 | Dim Joys As UInteger  Joys = 3000000000 |
| Long | 64-bit | –9,223,372,036,854,775,808 to  9,223,372,036,854,775,807 | Dim WorldPop As Long  WorldPop = 4800000004 |
| ULong | 64-bit | 0 through  18,446,744,073,709,551,615 | Dim Stars As ULong  Stars = \_  1800000000000000000 |
| Single | 32-bit  floating point | –3.4028235E38 through  3.4028235E38 | Dim Price As Single  Price = 899.99 |
| Double | 64-bit  floating point | –1.79769313486231E308 through  1.79769313486231E308 | Dim Pi As Double  Pi = 3.1415926535 |
| Decimal | 128-bit | 0 through +/–79,228,162,514,264,  337,593,543,950,335  (+/–7.9 . . . E+28) with no  decimal point; 0 through +/–  7.9228162514264337593543950335  with 28 places to the right of the  decimal. Append “D” if you want  to force Visual Basic to initialize a  Decimal. | Dim Debt As Decimal  Debt = 7600300.5D |
| Byte | 8-bit | 0 through 255 (no negative numbers) | Dim RetKey As Byte  RetKey = 13 |
| SByte | 8-bit | –128 through 127 | Dim NegVal As SByte  NegVal = –20 |
| Char | 16-bit | Any Unicode symbol in the range  0–65,535. Append “c” when  initializing a Char. | Dim UnicodeChar As Char  UnicodeChar = "Ä"c |
| String | Usually 16-bits  per character | 0 to approximately 2 billion  16-bit Unicode characters | Dim Dog As String  Dog = "pointer" |
| Boolean | 16-bit | True or False. (During conversions,  0 is converted to False, other values  to True.) | Dim Flag as Boolean  Flag = True |
| Date | 64-bit | January 1, 0001, through  December 31, 9999 | Dim Birthday as Date  Birthday = #3/1/1963# |
| Object | 32-bit | Any type can be stored in a variable of type Object. | Dim MyApp As Object  MyApp = CreateObject \_  ("Word.Application") |

(Naming Conventions:

exp: strName; making the first letter of a variable name lowercase) to distinguish variable names from functions and module names, which usually begin with uppercase letters; begin each variable name with a two-character or three-character abbreviation corresponding to the type of data that’s stored in the variable.)

# Special Variables/Properties:

|  |  |  |
| --- | --- | --- |
| DateString | | same as TimeString |
| System.DateTime.Now | | This property returns an encoded value representing the current date and time. This property is most useful as an argument for other system clock functions. |
| TimeString | | A property that contains the current time formatted for display or printing. |
| ColorDialog | SolidColorOnly | Set to True if you want the user to select only solid colors (dithered colors—those that are made up of pixels of different colors—are disabled). |
| ShowHelp | Set to True if you want to enable the Help button in the dialog box. |
| AllowFullOpen | Set to True to enable the Define Custom Colors button in the dialog box. |
| AnyColor | Set to True if the user can select any color shown in the dialog box. |
| FullOpen | Set to True if you want to display the Custom Colors area when the dialog box first opens. |
| MaskedTextBox | PasswordChar | The PasswordChar property can be used to display a placeholder character, such as an asterisk (\*), when the user types. |

# Keywords/Directive References:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Converting Variable Types** | | | | | |
| CStr(X) | Convert X to string type text, not necessary for numbers/variables sometimes | | | | |
| CSng(X) | Convert String X to single type, not necessary for numbers sometimes | | | | |
| Int(X) | Returns the integer portion of X | | | | |
|  |  | | | | |
| **Random** | | | | | |
| Rnd() | Creates a random number between 0 and 1 | | | | |
| Randomize | Uses the system clock to create a truly random starting point, or seed, for the Rnd() statement | | | | |
|  |  | | | | |
| **Statement** | | | | | |
| Dim X1, X2 As Y;  Dim X1, X2 As Y = Z | | | | X: variable names; Y: variable type; Z: value of Y | |
| Structure X  Dim Y1 As Z1  Dim Y2 As Z2  End Structure | | | | UDT statement; X as the name of the type; Y as the name of the component variables in X, Z as variable type; X.Y1, X.Y2 are full names of the variables in this UDT | |
| Const X As Y = Z;  Public Const X As Y = Z | | | | Similar to Dim statement | |
| Imports X.Y | | | | This statement references the X.Y class so that you can use its methods without being preceded by namespace in your program (must be the very first statement in code) | |
| Decision Structure | | If … Then | | If condition1 Then  statements executed if condition1 is True  ElseIf condition2 Then  statements executed if condition2 is True  [Additional ElseIf conditions and statements]  Else  statements executed if no conditions is True  End If | |
| Select Case | | Select Case X ‘ X as variable  Case value11, value12, value13  statements executed  Case value21 To value22  statements executed  Case Is > value3  statements executed  ...  Case Else  statements executed if no match is found  End Select | |
| Loops | | For … Next | | For X = value11 To value12 Step value2  ‘ X as variable, can be Integer/Long/Decimal  ‘ value11 and value 12 are start and end points of X  ‘ value2 is the increment of X  statements to be repeated  Next X ‘ X after Next is optional | |
| Exit For | | Statement to exit For…Next loop anytime inside the loop | |
| Do | | Do While/Until condition  block of statements to be executed  Loop | |
| Do  block of statements to be executed  Loop While/Until condition | |
|  | |  | |  | |
|  | |  | |  | |
| **Function** | | | | | |
| InputBox(X, Y) | | | | | X is shown by prompt as instruction, Y is the title of the inputbox; the function returns the input value |
| MsgBox(X, Y, Z) | | | | | X is shown by prompt as instruction, Y is a number that specifies the buttons, icons, and other options to display for the msgbox, Z is the title of the msgbox; the function returns results indicating which button the user clicked |
| Beep() | | | | | Make the computer speaker sound beep for one time |
| **Methods** | | | | | |
| Add(X) | | | | | Adds items to list boxes or other items |
| Process.Start(“X”, “Y”) | | | | | X as the name of .exe to run Y; Y as URL/path of target file |
| ShowDialog() | | | | | Use to open any form as a dialog box and returns a result named DialogResult |
| ToString() | | | | | converts the result to a textual value |
| System.Drawing.Image | | | FromFile(“X”) | | load document from the hard disk with “X” as path |
| System Clock | | | Hour(date) | | This method extracts the hour portion of the specified date/time value (0 through 23). |
| Minute(date) | | See Hour(date) |
| Month(date) | | See Hour(date) |
| Second(date) | | See Hour(date) |
| Weekday(date) | | See Hour(date) |
| Year(date) | | See Hour(date) |
| System.Math | | | Abs(n) | | Returns the absolute value of *n*. |
| Atan(n) | | Returns the arctangent, in radians, of n. |
| Cos(n) | | Returns the cosine of the angle n. The angle n is expressed in radians. |
| Exp(n) | | Returns the constant e raised to the power n. |
| Sign(n) | | Returns –1 if n is less than 0, 0 if n equals 0, and +1 if n is greater than 0. |
| Sin(n) | | Returns the sine of the angle n. The angle n is expressed in radians. |
| Sqrt(n) | | Returns the square root of n. |
| Tan(n) | | Returns the tangent of the angle n. The angle n is expressed in radians. |

# Tips:

• By default, a green jagged line indicates a warning, a red jagged line indicates a syntax error, a blue jagged line indicates a compiler error, and a purple jagged line indicates some other error.