MICROSOFT VISUAL BASIC.NET

[1]Aarti Singh, [2]Ananya Anikesh

[1,2] *Maharshi Dyanand University, (India)*

ABSTRACT

This paper assesses the utility of the Microsoft Visual Basic.Net programming language for teaching programming at a first level. Comparisons with Java are made.

I INTRODUCTION

It was the first version that could create 32-bit as well as 16-bit Windows programs. It has three editions; Standard, Professional, and Enterprise. It also introduced the ability to write non-GUI classes in Visual Basic. Incompatibilities between different releases of VB4 caused installation and operation problems. While previous versions of Visual Basic had Visual Basic is a third-generation event-driven programming language and integrated development environment (IDE) fromMicrosoft for its COM programming model first released in 1991. Microsoft intended Visual Basic to be relatively easy to learn and use.[1][2] Visual Basic was derived from BASIC and enables the rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using Data Access Objects, Remote Data Objects, or ActiveX Data Objects, and creation of ActiveX controls and objects.

The final release was version 6 in 1998 (now known simply as Visual Basic). Though Visual Basic 6.0 IDE is unsupported as of April 8, 2008, the Visual Basic team is committed to "It Just Works" compatibility for Visual Basic 6.0 applications on Windows Vista, Windows Server 2008 including R2, Windows 7, and Windows 8 In 2014 there are hundreds of thousands of developers who still prefer Visual Basic 6.0 over Visual Basic .NET Moreover, in recent years some developers lobbied aggressively for a new version of Visual Basic 6.0 A dialect of Visual Basic, Visual Basic for Applications (VBA), is used as a macro or scripting language within several Microsoft applications, including Microsoft Office

II HISTORY

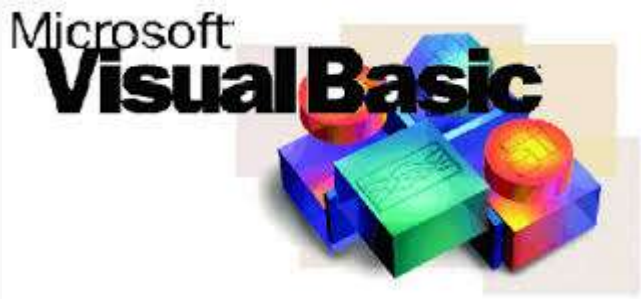
VB 1.0 was introduced in 1991. The drag and drop design for creating the user interface is derived from a prototype form generator developed by Alan Cooper and his company called *Tripod*.Microsoft contracted with Cooper and his associates to develop Tripod into a programmable form system for Windows 3.0, under the code name *Ruby* (no relation to the later Ruby programming language).

Tripod did not include a programming language at all. Microsoft decided to combine Ruby with the Basic language to create Visual Basic.

The Ruby interface generator provided the "visual" part of Visual Basic and this was combined with the "EB" Embedded BASIC engine designed for Microsoft's abandoned "Omega" database system. Ruby also provided the ability to load dynamic link libraries containing additional controls (then called "gizmos"), which later became the VBX interface.

III TIMELINE DESCRIPTION

VB DOS icon



- Project 'basic Thunder' was initiated in 1990.[20]
- Visual Basic 1.0 (May 1991) was released for Windows at the Comdex/Windows World trade show in Atlanta, Georgia.
- Visual Basic 1.0 for DOS was released in September 1992. The language itself was not quite compatible with Visual Basic for Windows, as it was actually the next version of Microsoft's DOS-based BASIC compilers, QuickBASIC and BASIC Professional Development System. The interface used a Text user interface, using extended ASCII characters to simulate the appearance of a GUI.
- Visual Basic 2.0 was released in November 1992. The programming environment was easier to use, and its speed was improved. Notably, forms became instantiable objects, thus laying the foundational concepts of class modules as were later offered in VB4.
- Visual Basic 3.0 was released in the summer of 1993 and came in Standard and Professional versions. VB3 included version 1.1 of theMicrosoft Jet Database Engine that could read and write Jet (or Access) 1.x databases.
- Visual Basic 4.0 (August 1995)used VBX controls, Visual Basic now used OLE controls (with files names ending in .OCX) instead. These were later to be named ActiveX controls.
- With version 5.0 (February 1997), Microsoft released Visual Basic exclusively for 32-bit versions of Windows. Programmers who preferred to write 16-bit programs were able to import programs written in Visual Basic 4.0 to Visual Basic 5.0, and Visual Basic 5.0 programs can easily be converted with Visual

Basic 4.0. Visual Basic 5.0 also introduced the ability to create custom user controls, as well as the ability to compile to native Windows executable code, speeding up calculation-intensive code execution. A free, downloadable Control Creation Edition was also released for creation of ActiveX controls. It was also used as an introductory form of Visual Basic: a regular .exe project could be created and run in the IDE, but not compiled.

- Visual Basic 6.0 (Mid-1998) improved in a number of areas[21] including the ability to create web-based applications. Visual Basic 6.0 has entered Microsoft's "non-supported phase" as of March 2008. Windows Vista, Windows Server 2008 and Windows 7, no longer support the Visual Basic 6.0 development environment, but still support the runtime.[22] Microsoft announced in February 2012 that they support the runtime in Windows 8.[23]
- Mainstream Support for Microsoft Visual Basic 6.0 ended on March 31, 2005. Extended support ended in March 2008.[24] In response, the Visual Basic user community expressed its grave concern and lobbied users to sign a petition to keep the product alive.[25] Microsoft has so far refused to change their position on the matter.[26] Ironically, around this time (2005), it was exposed that Microsoft's new anti-spyware offering, Microsoft AntiSpyware (part of the GIANT Company Software purchase), was coded in Visual Basic 6.0.[27] Its replacement, Windows Defender, was rewritten in C++.

IV PERFORMANCE

Earlier versions of Visual Basic (prior to version 5) compiled the code to P-Code only. The P-Code is interpreted by the language runtime. The benefits of P-Code include portability and smaller binary file sizes, but it usually slows down the execution, since having a runtime adds an additional layer of interpretation.

Visual Basic applications require Microsoft Visual Basic runtime MSVBVMxx.DLL, where xx is the relevant version number, either 50 or 60. MSVBVM60.dll comes as standard with Windows in all editions after Windows 98 while MSVBVM50.dll comes with all editions after Windows 95. A Windows 95 machine would however require inclusion with the installer of whichever dll was needed by the program.

Visual Basic 5 and 6 can compile code to either native or P-Code but in either case the runtime is still required for built in functions and forms management.

Criticisms levelled at Visual Basic editions prior to VB.NET include:[31]

- Versioning problems associated with various runtime DLLs, known as DLL hell
- Poor support for object-oriented programming[32]
- Inability to create multi-threaded applications, without resorting to Windows API calls
- Variant types have a greater performance and storage overhead than strongly typed programming languages

- Dependency on complex and fragile COM Registry entries[33]
- The development environment is no longer supported by MicrosofT.

V ADVANTAGE OF VISUAL BASIC

Quite a number of programming languages are text based and text based languages which do not allow user to work directly with graphics but visual basic is a graphical based language which allows user to work directly with graphic. Graphical based language can be used to develop windows program quickly.

Visual Basic gives a disciplined approach to writing programs that are clearer than unstructured programs, easier to test, debug and can be easily modify.

It allows for the creation of powerful and professional looking application with less time and coding. It all s for strong typing i.e. has wide variety of input data types and support Rapid Application Development (RAD).It has a complete edifying and debugging facilities and has the ability to generate a Dynamic Link Libraries (DLL`S), it allows for easier management of document and it is easy to learn.

Visual Basic is a complete form of package for building user interface

VI WHAT'S NEW ABOUT VB.NET?

VB.Net is a fully-fledged OO language, supporting encapsulation, inheritance and polymorphism. In contrast to C++ (but like Java) single inheritance is provided. Although VB.Net is sometimes referred to as VB7, the successor to VB6, there are marked changes and very little compatibility. VB.Net is part of a complete suite of tools associated with the .Net architecture. However, it can easily be regarded as a free-standing programming language that just happens to create components that fit within the .Net architecture.

In creating VB.Net, Microsoft have made VB into an elegant and consistent language. Some examples of this are:

·The syntax is cleaned up. For example, keywords End While are used to complete a While statement, rather than Wend the VB6 variant data type has gone. Instead the programmer must explicitly use the type of the variable parameters are, by default, call by value a VB.Net declaration such as Dim x, y, z As Integer means that the variables are all Integer there are no defaults for properties. For example, the programmer must write TextBox1.Text to describe the Text property of a text box. features such as graphics and file handling have been moved from the language itself to the libraries, which now present a coherent and OO interface to the programmer, a clean exception-handling mechanism has been provided in VB.Net in contrast to the clumsy VB6 mechanism

Provided that the appropriate compilation option is selected, VB.Net is a strongly-typed language.

VII WHAT'S THE SAME ABOUT VB.NET?

The Integrated Development Environment (IDE) for VB.Net is similar to that of older versions of VB. It displays 4 main windows - the tool box, the code window, the screen designer and a properties window. Like earlier versions

of VB, it allows the user to create a user interface by visually selecting GUI components from a tool box. Compilation and linking errors are clearly shown immediately as blue underlines.

VB.Net features

Vb.Net provides all the features necessary to teach either the Imperative-First or the Objects-First model of the ACM programming curricula (ACM 2001). It provides all the usual primitive data types, plus arrays. Strings are well-supported via library methods. The well-known control structures for iteration and selection are supported. Procedure and function methods are available, with reference or value arguments. GUIs can be built either visually or by hand coding. (Console input-output is also available for those who prefer that approach.) Writing classes is straightforward with overloading, single inheritance, polymorphism and interfaces all provided using meaningful keywords. A distinctive feature of VB.Net is properties. The exception-handling feature is structured and elegant.

The libraries provide access to a vast collection of facilities including data structures, graphics, file handling, database access. Alternatively, data structures can be implemented by the programmer. In common with other languages in which functionality has migrated from the language into the libraries, there is an increased need for good documentation on the extensive libraries.

The IDE automatically indents code as it is typed in. Both compilation and linking errors are immediately displayed and the debugger is excellent. Thus there is every support for the novice.

From the novice point of view, the biggest obstacle to understanding VB.Net is the sight of the code that is automatically created by the IDE and mentioned above. The only other conceptual difficulty is that while all objects are accessed by reference, the simpler primitive types such as Integer are not. This necessitates some clear thinking when arguments are passed.

Java versus VB.Net

Java is becoming widely adopted as a first programming language (Hosch 1996), so it makes sense to compare it with VB. VB.Net is uncannily similar to Java. Even the syntax of creating a new object is similar, so that for example the Java:

Button button = New Button("press here");

is equivalent to the VB.Net:

Dim myButton As Button = New Button("press here")

The VB.Net libraries are also markedly similar to those provided with Java.

Thus in terms of functionality of the language and of the libraries, Java and VB.Net are similar. However,VB.Net wins at the level of syntax, where semicolons are absent and Java curly brackets are replaced by keywords such as End If. This means that VB.Net is far easier for the novice.

One major difference between Java and VB.Net is in the areas of GUI programming and event handling. A GUI must be hand coded in Java, whereas the standard VB.Net IDE allows the user to create a GUI using a convenient drag and drop interface. In Java, the programmer must explicitly write the code to distinguish between different events, whereas in VB.Net, the concept of an event is built into the language. Thus event handling is shorter and less error-prone than in Java.

We have translated the programs presented in a well-known Java textbook into VB.Net. We found that the VB.Net programs were, on average, 60% shorter (lines of code), and, in addition, the VB.Net programs were markedly easier to develop, debug and test. Most of the difference is attributable to the convenience of GUI programming and event handling in VB.Net.

VIII CONCLUSION

Our conclusion is that VB.Net is worthy of consideration as a language for first level programming. It supports all the features required for learning imperative and OO programming and it provides every functionality via its libraries. The drawback is that even the simplest of programs incorporates some aspects of an advanced program (for example inheritance).

As compared with Java, VB.Net is probably easier to use, with no need to sacrifice programming principles.