Comparison of Object-Oriented Programming Languages

Timothy Clark (488232)

April 26, 2008

Contents

1 Introduction

What is a good programming language is a common topic for debate, everyone has their own opinion and favourite. In this report I am going to look at object orientated programming and some example languages to try and find out what makes these languages good, and when they are the right tool for the job.

Object Oriented languages came into existence in the 1960s but didn't come into common use until the early 1990s. The concept is to have a group of abstract interacting objects compared to a more linear execution path. The ease of code reuse, structuring and abstraction have made it a popular attribute of modern languages.

Languages comparison can be complex due to the large number of variables that languages can be compared by, and differing opinions on the priority of these variables. I am going to look at some important fundamental attributes of languages and there adherence to the idea of object orientation.

I have had to select a few programming languages to compare out of the thousands that have been created. I decided to pick some of the common contenders that are used in language debates, and ones that I have used so that I have a feel for the language itself. I settled on comparing Visual Basic, Java and Python.

2 What is object orientation

Object Oriented programming is made up from several fundamental elements. These are discussed in “The quarks of object-oriented development” [1] which identifies the most discussed elements. 50% of all sources researched in this paper talked about: Inheritance, Objects, Classes, Encapsulation, Methods, Message Passing, Polymorphism and Abstraction.

Inheritance is the idea that an object can use and extend the attributes of its parent object.

Objects are instances of classes, they contain data and methods to interact with and process the data they contain.

A Class is a template of functions and placeholders for data that is used to create multiple similar objects.

Encapsulation is the process of containing data within objects so it is only visible and manipulateable through the classes methods.

Methods are functions that allow you to process, access and manipulate and object or classes data.

Message Passing is the process of objects communicating data and requesting actions of each other.

Polymorphism is the design process that allows objects to transparently do different actions from the same method call and return data in the same way but formed from a different process.

Abstraction is the process of hiding the complexity of a system within a more general object.

These fundamental ideas allow for more efficient use of code and abstraction to make programs smaller, easier to write and easier to understand. An object oriented programming language can be compared against these properties to check how well it conforms to the idea of object orientation.

3 Comparison Methods

To successfully compare languages I need to decide on how and what I'm going to compare them by. As I am comparing object oriented languages, one of the things that I will compare them by is how much they adhere to the common properties of object orientated languages.

Programming languages are often compared by their running speed and how they are executed. Methods of execution include ones fully compiled into machine code, languages that compile into interpreted byte code and languages that are fully interpreted.[2]

Whether or not a programming language is compiled or interpreted does not affect the speed, it also affects the ease and portability of the program. Programs that are interpreted or part interpreted are less likely to need modifying to run on different platforms, operating systems and architectures.

The strength of the typing of a programming language is often given as a reason why a specific language is good or bad. Strongly typed languages impose more restrictions on the programmer on how data types are intermixed, this means that errors show up more at compile time than at run time, but this is at the cost of a degree flexibility. Loosely typed programs also run slower as the variables have to be cast when run rather than at compile time.

Programming languages are often made because there is not already a language that does the job a programmer requires, so they make a new one to solve a problem. Languages can be compared by the problem they were designed to solve.[3]

People often say that a language is good or bad because of how many people use it, this is often a good sign of what makes a particular language good for a specific task.

4 Languages

4.1 Visual Basic

Visual Basic is a language for Microsoft's Visual Studio development environment. The language can only be compiled with the development environment meaning that the environment makes a difference to the program. The language is very focused around making quick and easy user interfaces. The fact that the user is encouraged to make the interface then write the code often leads to lots of quick very buggy and badly designed programs.

Visual basic supports Inheritance, Objects, Classes, Encapsulation(through access modifier), Methods, Message Passing(through method headers and events), Polymorphism and Abstraction.[4]

Visual basic is half-compiled, it is compiled into p-code and then the p-code is interpreted at run time. This means that it is faster to execute than a fully interpreted program, but slower than a program that is compiled into a binary file. Unfortunately the fact that it is interpreted does not mean that it runs on more systems as Microsoft only supply the interpreter for its Windows operating system.

The language can be compiled in both strong and loosely typed mode. This allows for quick development of prototypes, but also careful development of highly complex programs that need to run fast.[4]

This language provides a user interface design tool as part of its compiler, it is also a lot like natural language in its syntax[4] there are very few popular languages that have these advantages which makes this a good language for beginners, making quick prototypes and user interfaces for the Windows operating system.

4.2 Java

Java is an open source language owed by Sun. The language is designed with syntax similar to C to make it easier for C programmers to switch to Java.

Visual basic supports Inheritance, Objects, Classes, Encapsulation, Methods, Message Passing(through method headers and events), Polymorphism and Abstraction.[5]

Like Visual Basic, Java is also half-compiled, it is compiled into the Java byte code then the byte code is interpreted at run time. Java is reasonably platform independent as Sun provide the Virtual Machine interpreter for lots of platforms, this allows files to be compiled once and distributed to different architectures and operating systems.

Java is a strongly typed language[5] which means that all type conversions bust be done explicitly, this means that casting errors are shown more at compile time and less at run time, but this can limit the flexibility of the language.

This language comes with C like syntax, a large built in class library, is strongly typed and is available for most common desktop platforms, this makes it a good language to introduce people to programming as it forces them to adopt good programming practices in a language that is powerful, useful and is used widely.

4.3 Python

Python is an open source interpreted language owned by the Python Software Foundation. The language was designed for "Rapid Application Development" and with an emphasis on readability.[6].

Python supports Inheritance, Objects, Classes, some very basic Encapsulation, Methods, Message Passing(through method headers and events), Polymorphism and Abstraction.[6]

Python is fully interpreted, however it does run a syntax check on the code before it begins. The standard interpreter is available for most common platforms and the source code and a fully comprehensive language specification are available so it can be ported to more platforms easily.

The language is technically strongly typed but the type is not declared explicitly, its type is calculated by what it must be to work for the required functions. This adds flexibility to the language but also allows run time errors if a variable is defined as two types.

This language was designed to be easy to read and program, and comes with a very big default library. It is available for all common platforms as it has an open source compiler. It is commonly used as a scripting language or to connect components together[6].

5 Conclusion

After looking into all three languages they all fit the object oriented specification to varying degrees, python is probably the worst fit as it does not support encapsulation very well. They are also interpreted to varying levels, Visual Basic and Java being compiled into a midway stage and the interpreted from that and python being fully interpreted but has a syntax check before it is run.

The three languages are available on different ranges of platforms, Visual Basic is the most limited of the three only being available on Microsoft's Windows operating systems. Java is the next compatible because its virtual machine is available for most common platforms and the source code is available. Python is probably the most compatible as the standard interpreter is available for most common platforms and has its source code available, and there are several other open source python interpreters available which were created from the language specification documents.

Both Visual Basic and Python both support strong and weak typing to some extent, Visual Basic more then Python. Java only supporting strong typing limits the speed and ease which programs can be written, but does force good programming practices.

All three of these programming languages are designed and are good at different tasks. Visual Basic is good for quick prototyping, user interface design and integration with Microsoft products. Java is good for large structured programs and for teaching good programming practice so that learning other languages such as C are easy. Python is good as a quick prototyping, scripting and glue[6] language as it is quick and easy to program in.

All three of these languages also have their own downfalls. Visual Basic tends to encourage sloppy programming style and because of its development environment and compiler only working on Windows it is the least platform independent. Java is not as easy to write programs in as the other two languages due to its structure and the fact it is only strongly typed. Python is the slowest of the three languages and does not fully encourage strong types so can cause more run time errors.

I would use all three of these languages, but only for solving the right problem. I would use Visual Basic for rapid visual prototyping on Windows. I would use Python for scripting, prototyping on a non Windows, and cross platform prototyping and scripting. I would use Java for producing larger structured programs on any platform.

References

[1] D. J. Armstrong, *The quarks of object-oriented development.* Communications of the ACM, Volume 49, Issue 2 (February 2006), Next-generation cyber forensics, Pages: 123 - 128.

[2] David Bolton, *Comparing Popular Programming Languages.*
URL: “http://cplus.about.com/od/introductiontoprogramming/a/comparelangs.htm”
Date: 5th March 2008.

[3] Paul Graham, *What Languages Fix.*
URL: “http://www.paulgraham.com/fix.html”
Date: 5th March 2008.

[4] Paul Vick, *The Microsoft Visual Basic Language Specification (version 8.0).*
URL: “http://msdn2.microsoft.com/en-gb/library/ms234437(VS.80).aspx”
Date: 22nd November 2007.

[5] James Gosling, Bill Joy, Guy Steele, Gilad Bracha, *The Java Language Specification (Third Edition).*
Source: The LIS(2nd Ed) or “http://java.sun.com/docs/books/jls/index.html” (3rd Ed),
LIS Call Number: QA76.73.J38 ¿GOS2,
2005.

[6] Guido van Rossum, *What is Python? Executive Summary.*
URL: “http://www.python.org/doc/essays/blurb/”
Date: 5th March 2008