Mohit Jain

CS 1501

June 22, 2018

Object Oriented Programming

**Introduction**

Object Oriented programming is the savior of all kinds of styles of programming. Object oriented programming is first and foremost about objects and developing the functions for any real word object. For example, let’s take a car object for example. A car has the following properties: the length of the car, the width of the car, the model, the license plate number, number of seats, etc. Using this kind of programming we can easily create such an object and model it as such. Then we can call instances of this object anywhere in our program and define its properties as we wish. This is amazing because now we do not have to keep rewriting the same kind of code over and over again. We just write it once, call it, and that’s it.

Now some of you may be wondering why it is used at all. We can use object oriented programming to prevent our code from looking like “spaghetti code.” Spaghetti code is defined as code that is very messy, unorganized, and extremely difficult to read. If every programmer were to write code as such, it would be very difficult for programmers to understand such code and develop on it. By using object oriented programming, it becomes a lot easier to divide up a project into multiple tasks. Otherwise, without it, we would be writing all the methods, functions, and classes within a single, huge class. This would become very difficult to read to and write to, and code like that is the kind of code we programmers want to avoid as much as possible.

**History**

The first object oriented programming language was SIMULA. As the name suggests, this programming language was used to create simulations. It was developed at the Norwegian Computing Center in Oslo by Ole-Johan Dahl and Kristen Nygaard. A person by the name of Alan Kay had a vision of a personal computer that would provide graphics-oriented applications and he felt like this language would provide a great way for non-specialists to create such applications. He then sold this vision to a company called Xerox Parc, and together with a team within the company in the early 1970s produced the first personal computer called the Dynabook. Smalltalk was the object-oriented language developed for programming the Dynabook. Using this invention, in the early 1980s, Bjorn Stroustrup integrated object-oriented programming into the C language, which is now famously known as C++, and this language became the first object-oriented programming language to be widely used. Then in the early 1990s, a group led by James Gosling decided to develop a more simpler form of C++, which is now known as Java. Initially was not very popular, but when it was marketed as a language for programming Internet applications, it gained widespread popularity as the Internet continued to boom.

**Features of Object Oriented Programming**

There are many features of object oriented programming that make it very useful. The first feature I want to take about are the classes. A class is essentially a factory that is able to mass produce objects. The programmer provides as class with a blueprint of the desired type of object, and this blueprint is composed of a declaration of a set of variables that the object will possess, which are also known as instance variables, the set of operations the object will provide, which are also known as methods, and a set of function definitions that implements each of these operations. When the user wants to create a new instance of an object, it asks a certain class to create the object for it. As such, when this object is created, the user is able to inherit all the methods and access the instance variables of that object.

With this, the next feature that makes object oriented programming very useful is what we call inheritance. In object oriented programming, this is referred to the inheritance of another object’s interface or extensions, so if let’s say an object called “Car” extends an object called “Driver,” and a class called Foo calls the object “Car,” then the class “Foo” also inherits the methods from “Driver” and can use them. This is amazing because it allows for much less code repetition. We do not have to insert the entire “Driver” class back inside the class “Foo.” We just call the object “Car,” and essentially kill two birds with one stone.

Another neat feature of object oriented programming concerns abstract data types. An abstract data type is a type of data and a set of operations(methods) for manipulating that data. Such examples of data types include tress and hash tables. An object is a great implementation for these abstract data types since the data stored by the object can represent the data of the abstract data type, so it is easily accessible.

**Objected Oriented Programming in Java vs. C++**

To do a comparison of these two programming languages, in my personal opinion I believe that C++ lacks key elements of a good object-oriented language. Even though with both languages you can still easily write the same kind of functionality for which you which to perform with the program, there are still a few key elements that I believe cause Java to reign in superiority over C++. First, in C++, there is no single object hierarchy. In Java, the class hierarchy forms a single tree, with the class Object at the root, whereas C++ had a multitude of classes, with zero “singled out” class like the Object class that Java provides. Java also never puts objects on the stack or inside array elements. It puts references of these object on them instead. C++ just dumps the objects straight on the stack. This can cause all sorts of problems. Let’s say for example, you have an object that is a superclass and then you have another object that is a subclass of that class. If you declare an array of the object that is a superclass, the subclass principle states that even though each instance is the superclass, each array element is really an instance of the subclass. Now this can cause problems only if the subclass is bigger than the superclass. In that case, all the array index operations will be wrong and when you try to access an element of that array, you may not get the element that you want. In Java, all of these are references, and this will not cause a problem because references are of the same size. To get past the problem C++ provides, you can just use pointers, but then again, that is just added work for the programmer, which is why I believe that Java is the better object-oriented programming language over C++. Not to mention the final detail, Java also has garbage collection so you do not have to worry about when you have used an object for the last time.

**Conclusion**

In conclusion, object oriented programming is very useful as it helps programmers write better, more clean, readable and professional code. One of the biggest use cases of object oriented programming is done in Android Development. Since in android development, a wide variety of classes exist for different pages and layouts, it is necessary to know how to use objects to reference these classes in order to freely send arguments from one class to another.

Sources

1. <http://web.eecs.utk.edu/~huangj/CS302S04/notes/oo-intro.html>
2. <http://www.eeng.dcu.ie/~ee553/ee402notes/html/ch05s06.html>
3. http://icarus.cs.weber.edu/~dab/cs1410/textbook/1.Basics/review.html