# Lesson 2: Introduction to Object-Oriented Programming

**Object-Oriented Programming**

Object-Oriented Programming (OOP) is similar to procedural programming. It is different, however, in that it envisions the components of a program as real world objects. In this class, we will create both objects and applications that use those objects.

Objects are made up of states and methods. The **states** or data items of an object are the components that define what the object is. These states describe the object's **attributes**. An automobile object would have attributes such as make, model, year, color, price etc. All automobiles contain these attributes although the specific attributes would be different for different automobiles. One automobile object may have the following specific attributes: make (Dodge), model (Stratus), year (2000), color (red), and price ($15,000). Another automobile object may have the following: make (Chevrolet), model (Impala), year (1966), color (blue), and price ($2,000). States are very similar to variables in procedural programming.

Objects also can have **methods** to accomplish a task. Just like procedures in procedural programming may perform a certain action, methods in OOP perform certain actions. An automobile object can move, be washed etc. Methods describe how these actions are accomplished. While states of an object could be called adjectives that describe the components of the object, methods are **verbs** that describe the actions that can be applied to an object. Object-Oriented Programming binds the states (variables) and the methods (procedures) together in one package. This binding of the two is called **encapsulation**.

A method should be so well written that the user is unaware of the details of how the methods are executed. The user must simply understand the **interface** of the method to interact with the object. For example, it is not necessary for someone to understand how a television remote control works in order to use the remote to change the stations or the volume. The user of the remote could be called a **client** that only knows how to use the remote to accomplish a certain task. The details of how the remote control performs the task are not necessary for the user to use the remote. Likewise an automobile is a complex mechanical machine with a simple interface that allows users without any (or very little) mechanical knowledge to start, drive and use it for a variety of functions. Drivers do not need to know what goes on under the hood. In the same way a user of an object does not have to understand how the objects methods are implemented.

Classes are the definitions from which objects are created. Classes and objects are often confused with one another; however, there is a subtle but important difference best explained by the following example. A plaster of Paris mold consists of the design of a particular figurine. When the plaster is poured into the mold and hardened, we have the creation of the figurine itself. A class is analogous to the mold, for it holds the definition of an object. The object is analogous to the figurine, for it is an **instance** of the class. Examples and further explanations are given later in the course.

## Java

Java was designed in the early 1990s by Sun Microsystems. It was designed as a compact object-oriented language to be used for cellular phone applications. In just a few years, however, it provided the interactivity for the World Wide Web and has since become a popular development language. It is an **architecturally neutral** language which means that a Java program can run on any operating system on any computer (**platform**) that has a Java interpreter. A Java program is compiled into what is called **bytecode** rather than to the machine language of the computer. A special program called an **interpreter** translates the bytecode to the machine language of the particular computer. Any compiled Java program (the bytecode) will run on any machine that has a Java programming language interpreter. This makes Java very **portable**.

$$\begin{tikzcd}\text{Java program (source code)} \arrow[r, "\text{Compiler}"'] & \text{bytecode} \arrow[r, "\text{Interpreter}"'] & \text{Machine code}\end{tikzcd}$$

Java is also simpler than most other object-oriented languages and is patterned after C++.

**Applets** are mini-Java programs that can be downloaded and executed as part of a Web page. Although the lessons in this manual concentrate on console (non-graphical, non-Web page) applications, we do cover and look at and develop applets later in the course.

## First Program in Java

Look at the following code:

```
// This is the first program that just writes out a simple message

public class First {
      public static void main(String[] args) {
            System.out.println("Now is the time for all good men");
            System.out.println("to come to the aid of their party");
      }
}
```

At first glance, the program looks a bit complex. We do not have to understand everything about the program to be able to run it and get results. We can, however, note a few things about this program.

The code above defines a class named `First`[1] that has a single method called `main`. The main method consists of two instructions. (specifically, two System.out.println statements).

Let's examine in more detail the statement System.out.println("Now is the time for all good men");

System.out.println represents three different components each separated from the others by a dot.

`System` is a class that provides general capabilities for Java programs, `out` is a field of the System class that references a PrintStream object that is defined automatically by Java, and `println` is a method of that object that will print a message to the screen and then move the cursor to the beginning of the next line.[2] The message that it will print is provided as an argument to the method. This argument is contained in the parentheses that follow the method name. A System.out.println statement outputs to the screen whatever is inside the double quotes within the parentheses that follow. The text "Now is the time for all good men" is a **literal string** of characters. All literal strings in Java appear between double quotes. The string appears within parentheses because the string is an **argument** to the method. Arguments are information that a method (i.e., procedure) needs to perform a task. The `println` method is called twice, each with a different message (literal string).

[1]Programmers have established the convention of beginning a class name with an uppercase letter.

[2]There is a method called print() that does the same thing as println() except that the cursor remains on the same line as the output.

# Lesson 2 Summary Outline

I. Object-Oriented Programming (OOP)

   A. Components of a program are viewed as **objects**
   B. **States** of an object describes the attributes of the object. Attributes describe what makes the object the object. Attributes of a car (such as tires, engine etc.) makes a car a car.
   C. **Method**– A method is a logical set of instructions that perform a certain task. Methods are verbs!!
   D. **Encapsulation**-The binding of states(variables, nouns) and methods together.
   E. **Interfacing**-The ability to use something without understanding the details of its operation.
   F. **Class**- the generic definition of a group of objects. The mold.
   G. An object is an **instance** of a class. Instance is a creation of an object from a class.

II. Java

   A. **Architecturally neutral**–Java can run on any platform.
   B. **bytecode**–the code that a Java compiler generates
   C. **Interpreter**-a program on a particular platform that translates the bytecode to machine code.
   D. **Applets** are mini-Java programs that can be downloaded and executed as part of a Web page.

III. First Java Program

```
public class First {       // Definition of a class called First
      public static void main(String[] args) { // Definition of the main method
            System.out.println("Now is the time for all good men");
            System.out.println("to come to the aid of their party");
      }
}
```