Chapter 3 : Objects and Classes

Defining Your Own Classes

However, all those classes had just asingle main method. Now the time has come to show you how to write the kind of “workhorseclasses” that are needed for more sophisticated applications. These classes typicallydo not have a main method. Instead, they have their own instance fields andmethods.

To build a complete program, you combine several classes, one of which has amain method.

An Employee Class

The simplest form for a class definition in Java is

```
classClassName
 {  constructor1constructor2
  . . .method1method2
  . . .field1field2
  . . .
  }
```

NOTE : We adopt the style that the methods for the class come first and the fields come at the end. Perhaps this, in a small way, encourages the notion of looking at the interface first and paying less attention to the implementation.

```
 class Employee
  {
  // constructor
public Employee(String n, double s, int year, int month, int day)
  {
  name = n;
  salary = s;
  GregorianCalendar calendar = new  GregorianCalendar(year, month - 1, day);
  hireDay = calendar.getTime();
  }
  // a method
```

```
public String getName()
  {
  return name;
  }
  // more methods
  . . .
  // instance fields
  private String name;
  private double salary;
  private Date hireDay;
  }
```

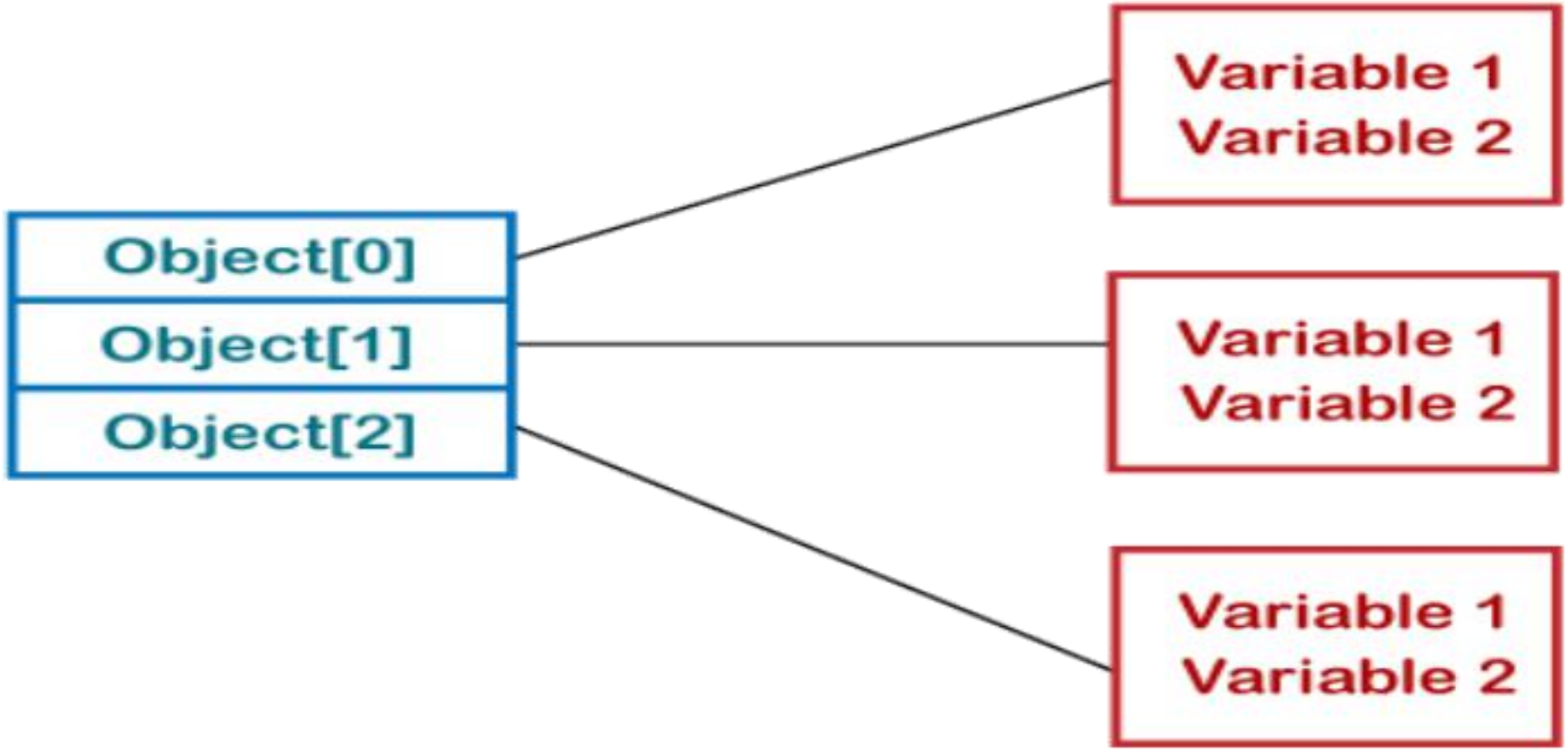
Array of Objects

Java is an object-oriented programming language. Most of the work done with the help of objects. We know that an array is a collection of the same data type that dynamically creates objects and can have elements of primitive types.

Java allows us to store objects in an array. In Java, the class is also a user-defined data type. An array that conations class type elements are known as an array of objects.

It stores the reference variable of the object.

Arrays of Objects



Creating an Array of Objects

Before creating an array of objects, we must create an instance of the class by using the new keyword. We can use any of the following statements to create an array of objects.

Syntax :

ClassName obj[]=new ClassName[array_length]; //declare and instantiate an array of objects

Or

ClassName[] objArray;

Or

ClassName objeArray[];

Access Specifiers (public, protected, private,default)

Specifier where a property / method is accesible there are more four types of access modifiers in java :

1. Public
2. Private
3. Default
4. Protected

Getter and Setters

Getter => Returns the value [accessors]

Setter => Sets / Updates the value [mutators]

Example :

```
public class Employee {
        private int id;
        private String name;
        public String getName() {
                        return name;
}   public void setName() {
                        this.name = “Your-name”;
}    public void setName(String n) {
                        this.name = n;
}
```

Constructors in Java

A member function used to initialize an object while creating it.

Employee codelab = new Employee();

codelab setName(“CodeLab”);

In order to write our own constructor, we define a method with name same as class name.

```
public Employee(){
        name = “Your Name”;
 }
```

Constructor Overloading in Java

Constructors can be overloaded just like other methods in java. We can overload the Employee Constructor like below :

```
public Employee(String n) {
    name = n;
  }
```

Note : 1. Constructors can the parameters without being overloaded.

2. There can be more than two overloaded Constructor.

Use of ‘this’ Keyword

The this keyword refers to the current object in a method or constructor.

The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter). If you omit the keyword in the example above, the output would be "0" instead of "5".

this can also be used to :

Invoke current class constructor

Invoke current class method

Return the current class object

Pass an argument in the method call

Pass an argument in the constructor call

Usage of static block, static fields and static methods

Static Block

Static block is used for initializing the static variables.This block gets executed when the class is loaded in the memory. A class can have multiple Static blocks, which will execute in the same sequence in which they have been written into the program.

Static Fields or Static Variable

If you declare any variable as static, it is known as a static variable.

The static variable can be used to refer to the common property of all objects (which is not unique for each object), for example, the company name of employees, college name of students, etc.

The static variable gets memory only once in the class area at the time of class loading.

Static Method :

If you apply static keyword with any method, it is known as static method.

A static method belongs to the class rather than the object of a class.

A static method can be invoked without the need for creating an instance of a class.

A static method can access static data member and can change the value of it.

Predefined class : Object class and its methods getClass(), clone()

In java, the Object class is the super most class of any class hierarchy. The Object class in the java programming language is present inside the java.lang package.

Every class in the java programming language is a subclass of Object class by default.

The Object class is useful when you want to refer to any object whose type you don't know. Because it is the superclass of all other classes in java, it can refer to any type of object.

Object clone() : Creates and returns a copy of the existing class object.

Class getClass():Returns a hash code value (numeric value) of the object of a class.

Method	Description	Return Value
getClass()	Returns Class class object	object
hashCode()	returns the hashcode number for object being used.	int
equals(Object obj)	compares the argument object to calling object.	boolean
clone()	Compares two strings, ignoring case	int
concat(String)	Creates copy of invoking object	object
toString()	eturns the string representation of invoking object.	String
notify()	wakes up a thread, waiting on invoking object's monitor.	void
notifyAll()	wakes up all the threads, waiting on invoking object's monitor.	void
wait()	causes the current thread to wait, until another thread notifies.	void
wait(long,int)	causes the current thread to wait for the specified milliseconds and nanoseconds, until another thread notifies.	void
finalize()	It is invoked by the garbage collector before an object is being garbage collected.	void

Usage of built-in string functions and mathematical functions

Method	Description
equals()	Compares two strings. Returns true if the strings are equal, and false if not
toString()	Returns the value of a String object

Method	Description
sqrt()	It is used to return the square root of a number.
Math.pow()	It returns the value of first argument raised to the power to second argument.
round()	It is used to round of the decimal numbers to the nearest value.

Inner Class, Anonymous Class

Inner Class

Creating an inner class is quite simple. You just need to write a class within a class. Unlike a class, an inner class can be private and once you declare an inner class private, it cannot be accessed from an object outside the class.

Anonymous Class

An anonymous class in Java is a class not given a name and is both declared and instantiated in a single statement. You should consider using an anonymous class whenever you need to create a class that will be instantiated only once.

Creating a JAR File

The basic format of the command for creating a JAR file is:

jar cf jar-file input-file(s)

The options and arguments used in this command are :

The c option indicates that you want to create a JAR file.

The f option indicates that you want the output to go to a file rather than to stdout.

jar-file is the name that you want the resulting JAR file to have. You can use any filename for a JAR file. By convention, JAR filenames are given a .jar extension, though this is not required.

The input-file(s) argument is a space-separated list of one or more files that you want to include in your JAR file. The input-file(s) argument can contain the wildcard * symbol. If any of the "input-files" are directories, the contents of those directories are added to the JAR archive recursively.

Creating a Manifest File

To create manifest file, you need to write Main-Class, then colon, then space, then classname then enter.

For example :

myfile.mf

Main-Class: First

As you can see, the mf file starts with Main-Class colon space class name. Here, class name is First.

Garbage Collection (System.gc(), finalize() Method)

In java, garbage means unreferenced objects.

Garbage Collection is process of reclaiming the runtime unused memory automatically. In other words, it is a way to destroy the unused objects.

To do so, we were using free() function in C language and delete() in C++. But, in java it is performed automatically. So, java provides better memory management.

finalize() Method

The finalize() method is invoked each time before the object is garbage collected. This method can be used to perform cleanup processing.

This method is defined in Object class as:

protected void finalize(){}

gc() method

The gc() method is used to invoke the garbage collector to perform cleanup processing.

The gc() is found in System and Runtime classes.

public static void gc(){}