Core Java 8

Lesson 04 : Classes and Objects

Capgemini

Lesson Objectives

- After completing this lesson, participants will be able to:
  - Define classes and objects
  - Work with Access Controls
  - Define Constructors
  - Overloading
  - Static Method and fields
  - Garbage Collection-finalize() method
  - Extended Parameters for JVM-xms,-xmx
  - Memory Leakage, Stack overflow, Out of Memory
  - The toString Method

Classes and Objects

- Class:
  - A template for multiple objects with similar features
  - A blueprint or the definition of objects
- Object:
  - Instance of a class
  - Concrete representation of class

```
class < class_name>
{
    type var1; …
    Type method_name(arguments )
     {
        body
      } …
} //class ends
```

Introduction to Classes

- A class may consist the following elements:
  - Fields
  - Methods
  - Constructors
  - Initializers

Introduction to Classes

```
class Box{
        double dblWidth;
        double dblHeight;
        double dblDepth;
        double calcVolume(){
                return dblWidth * dblHeight * dblDepth;
        } //method calcVolume ends.
}//class Box ends.
```

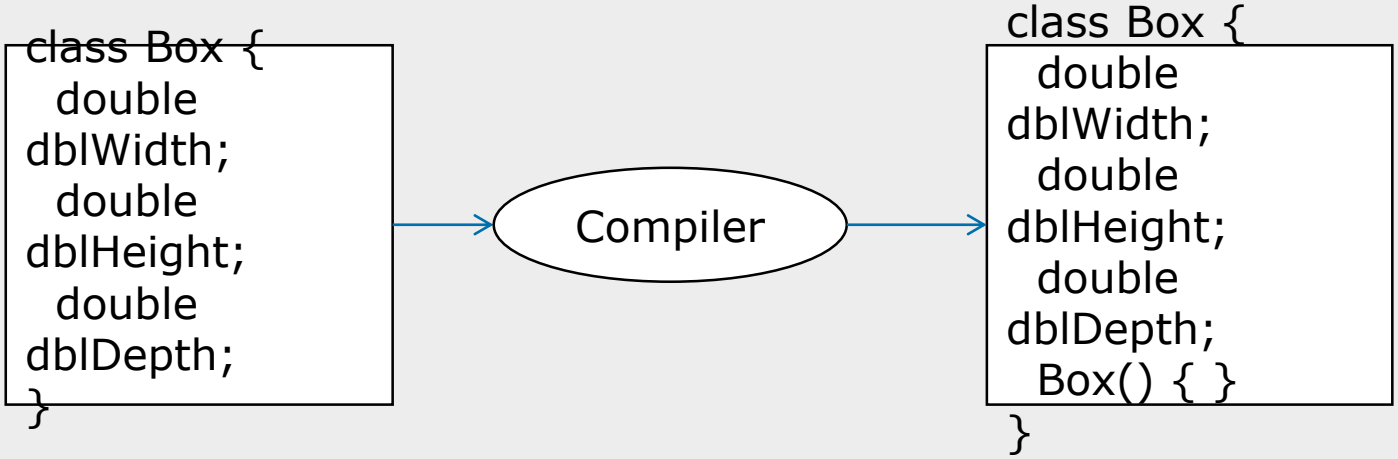
```
class BoxDemo{
    public static void main(String a[])
    {
        Box box;                        //declare a reference to object
        box = new Box();                //allocate a memory for box object.
        box.calcVolume();               // call a method on that object.
} }
```

Types of Access Controls

- Default
- Private
- Public
- Protected

Default Constructors

- All Java classes have *constructors*
  - Constructors initialize a new object of that type
- Default no-argument constructor is provided if program has no constructors
- Constructors:
  - Same name as the class
  - No return type, not even void



Overloading

- Two or more methods within the same class share the *same* name. Parameter declarations are different
- You can overload Constructors and Normal Methods

```
class Box {
    Box(){
        //1. default no-argument constructor
    }
    Box(dbl dblValue){
        // 2. constructor with 1 arg
    }
   public static void main(String[] args){
Box  boxObj1 = new Box(); // calls constructor 1
Box  boxObj2 = new Box(30);  // calls constructor 2
    } }
```

Demo

- Execute the BoxDemo.java program.
  - This uses the Box.java

Static modifier

- Static modifier can be used in conjunction with:
  - A variable
  - A method
- Static members can be accessed before an object of a class is created, by using the class name
- Static variable :
  - Is shared by all the class members
  - Used independently of objects of that class
  - Example: static int intMinBalance = 500;

Static modifier

- Static methods:
  - Can only call other static methods
  - Must only access other static data
  - Cannot refer to this or super in any way
  - Cannot access non-static variables and m
- Static constructor:
  - used to initialize static variables

Method main() is a static method. It is called by JVM.

Static modifier

```
// Demonstrate static variables, methods, and blocks.
public class UseStatic {
    static int intNum1 = 3;                    // static variable
    static int intNum2;
static {                                  //static constructor
          System.out.println("Static block initialized.");
          intNum2 = intNum1 * 4;
      }
    static void myMethod(int intNum3) {     // static method
           System.out.println("Number3 = " + intNum3);
           System.out.println("Number1 = " + intNum1);
           System.out.println("Number2 = " + int
     }
public static void main(String args[]) {
    myMethod(42);
   }}
```

Output:
Static block initialized.
Number3 = 42
Number1 = 3
Number2 = 12

Demo

- Execute UsingStatic.java Program

Enhancement in Garbage Collector

- Garbage Collector:
  - Lowest Priority Daemon Thread
  - Runs in the background when JVM starts
  - Collects all the unreferenced objects
  - Frees the space occupied by these objects
  - Call *System.gc()* method to “hint” the JVM to invoke the garbage collector
    - There is no guarantee that it would be invoked. It is implementation dependent

Finalize() Method

- Memory is automatically de-allocated in Java
- Invoke *finalize()* to perform some housekeeping tasks before an object is garbage collected
- Invoked just before the garbage collector runs:
  - protected void finalize()

JVM Parameters

- The flag Xmx specifies the maximum memory allocation pool for a Java Virtual Machine (JVM), while Xms specifies the initial memory allocation pool.

- This means that your JVM will be started with Xms amount of memory and will be able to use a maximum of Xmx amount of memory. For example, starting a JVM like below will start it with 256MB of memory, and will allow the process to use up to 2048MB of memory:
- The Xms flag has no default value, and Xmx typically has a default value of 256MB. A common use for these flags is when you encounter a java.lang.OutOfMemoryError

Memory Leakage ,Overflow and Out of Memory

- A memory leak occurs when memory acquired by a program for execution is never freed-up to be used by other programs and applications.

- A stack is the part of the memory. The local automatic variable is created on this stack and method arguments are passed. When a process starts, it get a default stack size which is fixed for each process. In today's operating system, generally, the default stack size is 1 Mb, which is enough for most of the process. Under abnormal condition, the stack limit exceeds. This is known as stack overflow.
- Usually, Out of Memory error is thrown when there is insufficient space to allocate an object in the Java heap. ... When a java.lang.OutOfMemoryError exception is thrown, a stack trace is also printed.

toString() Method in Java

- The method is used to get a String object representing the value of the Number Object.
- If the method takes a primitive data type as an argument, then the String object representing the primitive data type value is returned.

Lab

- Lab 4:

Summary

- In this lesson you have learnt:
  - Classes and Objects
  - Access Controls
  - Constructors - Default and Parameterized
  - Overloading
  - Using static keyword
  - Garbage Collection

Review Questions

- Which method is used to perform some housekeeping work in Java program?
  - final()
  - finalize()
  - release()
  - final keyword
- Static keyword is used with?
  - Constructor
  - Class
  - Fields
  - Method