STATIC FIELDS, METHODS AND BLOCKS

Mr. S. G. Lakhdive

STATIC FIELDS

- When a number of objects are created from the same class, they each have their own distinct copies of *instance variables*.
- For Rectangle class, the instance variables are length and breadth.
- Sometimes, you want to have variables that are common to all objects
- They don’t belong to each object but belong to the class.
- These fields are called static fields or class variables.
- Every object shares a class variable, which is at one fixed location in memory.

STATIC FIELDS

- Suppose we want to keep a track of all Rectangle objects that have been created.
- Such a field is not related to any individual object, but to the class as a whole.
- For this, we need a class variable, **numberOfObjects** which will be a static field of the class.
- This field will be incremented when an object is created i.e. in the constructor, we can increment this field.

STATIC FIELDS

```
class Rectangle
{
        int length;
        int breadth;
        static int numberOfObjects = 0;
    ……
}
```

- Since static variables do not belong to any object, they are accessed using the class name itself.
- ***Syntax:*** ClassName.staticMember
- ***Example:*** Rectangle.numberOfObjects

Static Methods

- Java supports Static methods as well as static variables.
- Static methods are methods which are declared static.
- Static methods do not belong to an object but belong to the class.
- They exist even when there are no objects.
- We have seen that main is a static method.
- A static method can be invoked using the class name.
- ***Syntax:*** ClassName.methodName(args)

Static Methods

- Static methods usually access static data members of the class. *For example*, we can define a static method **getNumberOfObjects** which returns the value of static member numberOfObjects.
- static int getNumberOfObjects()

```
{
  return numberOfObjects;
}
```

- This method can be invoked in main using the class name:
- Rectangle.getNumberOfObjects();

Rules for Static Variables and Methods

- Instance methods can access instance variables and instance methods directly.
- Instance methods can access class variables and class methods directly.
- Class methods can access class variables and class methods directly.
- Class methods cannot access instance variables or instance methods directly—they must use an object reference.
- Class methods cannot use the this keyword as there is no object.

Static Blocks

- In addition to static fields and methods, a class can have a static block which belongs to the class.
- This block will be executed even before main is invoked.
- This block is executed only once – when the class is loaded.
- Its main purpose is to initialize static variables.

Static Blocks

```
class StaticBlock
        {
                static int a;
                static int b;
        static {
                System.out.println("Static block initialized.");
                a = 10; b = 20;
        }
        static void method() {
                        System.out.println("a = " + a);
                        System.out.println("b = " + b);
        }
        public static void main(String args[]) {
                method();
        }
}
```