Classes in Java

By Rahul Barve

Objectives

- Class Basics
- Object Creation
- Access Modifiers
- Working with Methods
- Method Overloading
- Working with Constructors
- Understanding `this`
- Understanding Static Members
- Variable Types

What is Class

By Rahul Barve

What is Class

- A class is at the core of Java.
- Any concept to be implemented, must be encapsulated within a class.

What is Class

- A Class is a template that decides a structure for an object.
- Using classes and objects, one can map 2 major pillars of OOP: Abstraction and Encapsulation.

Class Syntax

```
class <class-name> {
  <variable-declarations>
  <method-definitions>
}
```

A Simple Class

```
class Planet{
  String name;
  int moons;
}
```

Creating Object

By Rahul Barve

Creating Object

- Once a class is created, it can be further used by creating its object.
- Syntax:

```
<class-name> <ref-var-name> =
                    new <class-name>();
```

Access Modifiers

By Rahul Barve

Access Modifiers

- Access modifiers are used to specify the scope of class members.
- These are `private`, `public`, `protected` and `default`.

Adding Methods

By Rahul Barve

Adding Methods

- Once an object is created, at any time it can be manipulated with the help of methods.
- Methods act like behaviors or operations.

Adding Methods

- Syntax:

```
<return-type> <method-name> ([param-list]){
      //Some Code
}
```

Accessor and Mutator Methods

By Rahul Barve

Accessor and Mutator Methods

- Used to retrieve or modify the values of the fields for a specific object.
- These methods follow the convention:

  `getXXX()` and `setXXX()`

Method Overloading

By Rahul Barve

Method Overloading

- If 2 or multiple methods have same name but different signatures, then those methods are called as overloaded methods.
- It’s a compile time polymorphism.

Method Overloading

- E.g.

```
class Test {
    void test(){ }
    void test(int a){ }
    void test(int a, int b} { }
    void test(int a, String b){ }
    void test(String a, int b){ }
 }
```

Method Overloading

```
Test t = new Test();
t.test();
t.test(10);
t.test(10,20);
t.test(10, “Hello”);
t.test(“Welcome”, 20);
t.test(“Hello”, “Welcome”);→Error
```

Constructor

By Rahul Barve

Constructor

- Constructor is a special member within a class having same name as that of a class name.
- It is invoked implicitly as soon as an object is created.

Constructor

- Does not have any return type.
- Constructors are used for object initialization.

Constructor

- A constructor without any parameter is known as no-argument constructor.
- Constructors can be overloaded.

Constructor

```
class Box {
      int length, width, height;
      Box(){      //No-Argument
            length = 10;
            width = 8;
            height = 5;
      }
      Box(int l, int w, int h){   //Parameterized
            length = l;
            width = w;
            height = h;
      }
}
```

Constructor

```
Box box1 = new Box();
//Invokes no-argument constructor

Box box2 = new Box(20,15,12);
//Invokes parameterized constructor

Box box3 = new Box(12);
//Error
```