Core Java

Lesson 05 : Extending Classes

Capgemini

Lesson Objectives

- In this Lesson you will learn-
  - Inheritance
  - Using protected Keyword
  - Constructor in extended classes
  - Overriding Methods
  - Polymorphism
  - Making Methods and Classes Final

What is Inheritance?

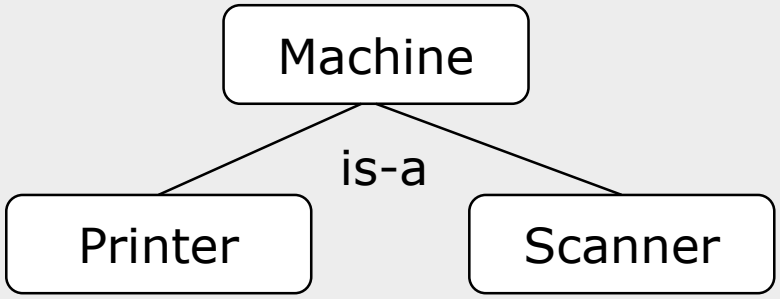
Basic TV

Smart TV

Smart TV's are inherited from basic television which apart from multimedia functionality of TV allows us to do more like streaming video contents from Internet.

What is Inheritance?

- Inheritance allows programmers to reuse of existing classes and make them extendible either for enhancement or alteration
- Allows creation of hierarchical classification
- Advantage is reusability of the code:
  - A class, once defined and debugged, can be used to create further derived classes
- Extend existing code to adapt to different situations
- Inheritance is ideal for those classes which has “is-a” relationship
- “Object” class is the ultimate superclass in Java



Protected Keyword

- The main purpose of **protected** keyword is to have the method or variable can be inherited from sub classes.
- ***protected*** members can be accessed from the class itself, subclasses of the class and also all classes in the same package of the class (doesn't matter if those are subclasses or not),by subclasses even if they are in another packages.

Constructors in extended classes

- In Java, constructor of base class with no argument gets automatically called in derived class constructor. For example, output of following program is:
- *Base Class Constructor Called*
  Derived Class Constructor Called

```
// filename: Main.java
class Base {
  Base() {
    System.out.println("Base Class Constructor Called ");
  }
}

class Derived extends Base {
  Derived() {
    System.out.println("Derived Class Constructor Called ");
  }
}

public class Main {
  public static void main(String[] args) {
    Derived d = new Derived();
  }
}
```

Constructors in extended classes

- If we want to call parameterized contructor of base class, then we can call it using super(). The point to note is **base class constructor call must be the first line in derived class constructor**. For example, in the following program, super(_x) is first line derived class constructor.

```
// filename: Main.java
class Base {
  int x;
  Base(int _x) {
    x = _x;
  }
}

class Derived extends Base {
  int y;
  Derived(int _x, int _y) {
    super(_x);
    y = _y;
  }
  void Display() {
    System.out.println("x = "+x+", y = "+y);
  }
}

public class Main {
  public static void main(String[] args) {
    Derived d = new Derived(10, 20);
    d.Display();
  }
}
```

Method Overriding

- In a class hierarchy, when a method in a subclass has the same *name* and *type signature* as a method in its super class, then the subclass method overrides the super class method
- Overridden methods allow Java to support run-time polymorphism

Normal Swap Machine

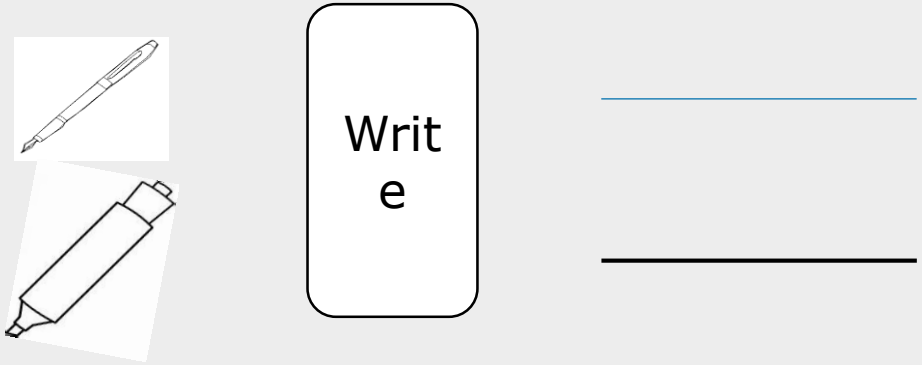
Chip card Machine which **overrides** the card reading for better security

Demo

- Polymorphism
  - Inheritance
  - Method Overriding

What is Polymorphism?

- Poly meaning “many” and morph means “forms”
- It’s capability of method to do different things based on the object used for invoking method



- Polymorphism also enables an object to determine which method implementation to invoke upon receiving a method call
- Java implements polymorphism in two ways
  - Method Overloading
  - Method Overriding

Final Modifier

- Final Modifier : Can be applied to variables, methods and classes
- Final variable:
  - Behaves like a constant; i.e. once initialized, it's value cannot be changed
  - Example: final int i = 10;
- Final Method:
  - Method declared as final cannot be overridden in subclasses
  - Their values cannot change their value once initialized
  - Example:

```
class A  {
   public final int add (int a, int b)  { return a
}
```

A class or metho… cannot be abstract & final at the same time.

- Final class:
  - Cannot be sub-classed at all
  - Examples: *String* and *StringBuffer* class

Lab

- Lab 5: Extending Classes

Summary

- In this lesson, you have learnt about:
  - Inheritance
  - Method overriding
  - Using final keyword
  - Best Practices

Review Question

- Question 1: Which of the following options enable parent class to avoid overriding of its methods.
  - extends
  - Override
  - Final
- Question 2: When you want to invoke parent class method from child, it should be written as first statement in child class method
  - True/False
- Question 3: Which of the following access specifier enables child class residing in different package to access parent class methods?
  - private
  - public
  - Final
  - Protected