Inheritance in Java

Prepared by Harish Patnaik

School of Computer Engineering, KIIT Deemed to be University

Content

Introduction

- It is a concept of OOP which allows creation of hirarchical classification of objects.
- It is the process by which one object acquires the properties of another object.

$$\begin{tikzcd}
\text{superclass} \arrow[d, "\text{extends}"] \\
\text{subclass}
\end{tikzcd}$$

Example - X.java, Y.java
inhdemo.java

box.java
woodbox.java
boxinh.java

Multi-level inheritance

- Java supports multi-level inhertance .

```
class X{
        X(){
        System.out.println(“From X const -”);
        } }

class Y extends X{
        Y(){
        System.out.println(“From Y const -”);
        } }

class Z extends Y{
        Z(){
        System.out.println(“From Z const -”);
        } }
```

Multi-level inheritance

```
class demo{
        public static void main (String ar[]){
        Z  ob1= new Z();
        } }
```

Output-

```
        From X const -
        From Y const -
        From Z const -
```

Note - Constructors are called in order of derivation, from superclass to subclass.

Example - supercon.java

Multiple inheritance

- Java does not support multiple inheritance.

```
class X{
        X(){
        System.out.println(“From X const -”);
        } }

class Y{
        Y(){
        System.out.println(“From Y const -”);
        } }

class Z extends Y, X{                    /Error
        Z(){
        System.out.println(“From Z const -”);
        } }
```

Method overriding

- ✓ We can have method with same signature in both super class and subclass

- ✓ If we call that method with an object of subclass then the subclass method will override on super class method.

Example - methover.java

'final' keyword

Uses of final keyword

- ✓ final data member - constant
- ✓ final method - prevents method overriding
- ✓ final class - prevents inheritance

Example - finals.java methover.java
finalC.java
infinalC.java

Handler

Example - base.java
superconst.java

Dynamic method dispatch

- It is a mechanism by which a call to an overriden method is resolved at run time rather than compile time.

Example -

Write a program in java to create a class Bank having ROI (Rate of Interest) data member and find_ROI() member function. Derive two classes HDFC, ICICI with find_ROI() function . The ROI of HDFC bank is calculated as ROI = (Last year annual profit / 1.5 crore) where the annual profit is an user entered value. The ROI of ICICI bank is calculated as ROI = Fund supported by RBI / 1 Crore where Fund supported by RBI is an user entered value. So find the rate of interest of all the Banks using dynamic method dispatch concept.

Illustration - dynamethod.java

Thank you