**Inner Classes – Part-05**

* **Various Combinations of Nested Classes & Interfaces:**

**Case\_01: Class Inside A Class:**

Without existing one type of object, if there is no chance of existing another type of object, then we can declare class inside a class.

Example:

University consists of several departments. Without out existing University, there is no chance of existing department. Hence, we have to declare Department class inside University class.

class University{

class Department{

}

}

**Case\_02: Interface Inside A Class:**

Inside a class if we require multiple implementations of an interface and all these implementations are related to a particular class. Then, we can define interface inside a class.

Example:

class VehicleTypes{

interface Vehicle{

public int getNoOfWheels();

}

class Bus implements Vehicle{

public int getNoOfWheels(){

return 6;

}

}

class Auto implements Vehicle{

public int getNoOfWheels(){

return 3;

}

}

}

**Case\_03: Interface Inside Interface:**

We can declare interface inside interface.

Example:

A Map is a group of key value pair, and each key-value pair is called an Entry.

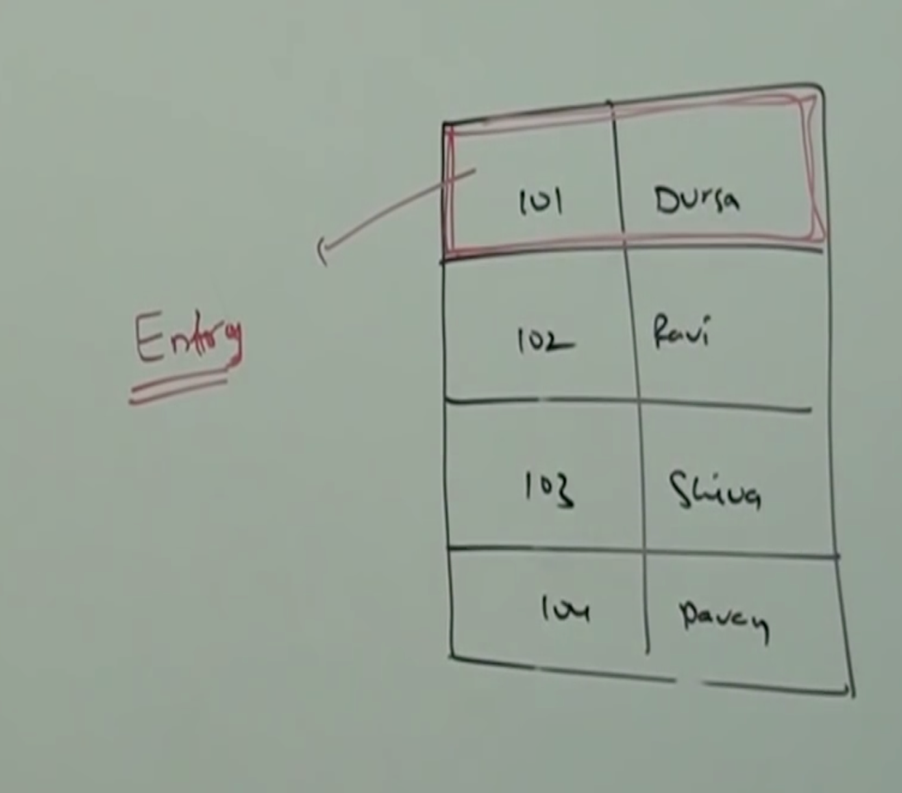
Without existing Map object there is no chance of existing Entry object. Hence, interface Entry is defined inside Map interface.

interface Map{

interface Entry{

}

}



Every interface present inside interface is always public and static whether we are declaring or not. Hence, we can implement inner interface directly without implementing outer interface.

Similarly, whenever we are implementing outer interface we are not required to implement inner interface, that is we can implement outer and inner interfaces independently.

Example:

interface Outer{

public void m1();

interface Inner{

public void m2();

}

}

class Test1 implements Outer{

public void m1(){

System.out.println(“Outer interface method implementation”);

}

}

class Test2 implements Outer.Inner{

public void m2(){

System.out.println(“Inner interface method implementation”);

}

}

class Test{

public static void main(String[] args){

Test1 t1 = new Test1();

t1.m1();

Test1 t2 = new Test1();

t2.m2();

}

}

**Case\_04: Class Inside Interface:**

If functionality of a class is closely associated with interface then it is highly recommended to declare class inside interface.

Example:

interface EmailService{

public void senMail(EmailDetails emailDetails);

class EmailDetails{

String toList;

String ccList;

String subject;

}

}

In the above example EmailDetails is required only for EmailService and we are not using anywhere else. Hence, EmailDetails class is recommended to declare inside EmailService interface.

We can also define a class inside interface to provide default implementation for that interface.

Example:

interface Vehicle{

public int getNoOfWheels();

class DefaultValue implements Vehicle{

public int getNoOfWheels(){

return 2;

}

}

}

class Bus implements Vehicle{

public int getNoOfWheels(){

return 6;

}

}

class Test{

public static void main(String[] args){

Vehicle.DefaultVehicle d = new Vehicle.DefaultVehicle();

d.getNoOfWheels();

Bus b = new Bus();

b.getNoOfWheels();

}

}

In the above example, default Vehicle is the default implementation of Vehicle interface, whereas Bus is customized implementation of Vehicle interface.

Note:

The class which is declared inside interface is always “public, static” whether we are declaring or not. Hence, we can create class object directly without having outer interface type object.

* **Conclusions:**

1. Among classes and interfaces we can declare anything inside anything.

class A{

class B{

}

}

class A{

interface B{

}

}

interface A{

interface B{

}

}

interface A{

class B{

}

}

1. The interface which is declared inside interface is always “public & static” whether we are declaring or not.

The class which is declared inside interface is always “public and static”, whether we are declaring or not.

1. The interface which is declared inside a class is always static, but need not be public.