**Declarations and Access Modifiers – Part-13- Interfaces-03**

* Marker Interface:
  + If an interface doesn’t contain any methods and by implementing that interface, if our objects will get some ability such type of interfaces are called Marker interface or Ability interface or Tag interface.
  + Example:

Serializable

Clonable

RandomAccess

SingleThreadModel

These are marked for some ability.

* + Example\_1:

By implementing serializable interface, our objects can be saved to the file and can travel across the network.

* + Example\_2:

By implementing cloneable interface, our objects are in a position to produce exactly duplicate cloned objects.

* Without having any methods, how the objects will get some ability in marker interfaces?
  + Internally JVM is responsible to provide required ability.
  + Why JVM is providing required ability in marker interfaces?

To reduce complexity of programming and to make Java language as simple.

Note: Recall about searching job in abroad.

Hint: Searching job our own or with the help of another person.

* Is it possible to create our own interface?
  + Yes, but customization of JVM is required.
* Adapter Classes:
  + Adapter class is a simple Java class that implements an interface with only empty implementation.
  + Example:

interface X{

m1()

m2()

m3()

m4()

.

.

.

M1000()

}

class AdapterX implements X{

m1(){}

m2(){}

m3(){}

m1000(){}

}

* + If we implement an interface for each and every method of that interface, compulsory we should provide implementation. Whether it is required or not required.

class Test implements X{

m3(){

// 10 lines of code

}

m1(){}

m2(){}

m4(){}

m1000(){}

}

The problem with this approach is it increases length of the code and reduces readability. We can solve this problem by using Adapter classes.

Instead of implementing interface if we extend adapter class we have to provide implementation only for required methods, and we are not responsible to provide implementation for each and every method of the interface. So that length of the code will be reduced.

class Test extends Adapter{

m3(){

}

}

class Sample extends Adapter{

m7(){

}

}

class Demo extends Adapter{

m10(){

}

}

* + Example:

We can develop a servlet in the following three ways.

1. By implementing Servlet interface.
2. By extending GenericServlet.
3. By extending HttpSevlet

Servlet

GenericServlet

HttpServlet

If we implement Servlet interface for each and every of that

Interface we should provide implementation. It increases

Length of the code and reduces readability.

Instead of implementing Servlet interface directly, if we extend GenericServlet, we have to provide implementation only for service method and for all remaining methods we are not required to provide implementation. Hence more or les GenericServlet acts as Adapter class for servlet for interface.

* + Note:

Marker interface and Adapter classes simplifies complexity of programming and these are best utilities to the programmer, and programmers life will become simple.