Dt : 14/10/2022

Concrete methods in Interfaces:(Java8 - new feature)

=>From Java8 version onwards the interfaces can be declared with Concrete methods.

=>The following are list of Concrete methods can be declared in Interfaces:

(a)static concrete methods(Java8)

(b)default concrete methods(Java8)

(c)private concrete methods(Java9)

(a)static concrete methods(Java8):

=>The concrete methods which are declared in interfaces with "static" keyword are known as Static concrete methods.

=>These static concrete methods will get the memory within the interface while interface loading and can be accessed with interface name.

Coding Rule:

=>Static concrete methods of Interface are not available to implementation classes,which means Implementation classes cannot access static concrete methods of Interface.

(b)default concrete methods(Java8):

=>The concrete methods which are declared in Interfaces with "default" keyword are known as default concrete methods.

=>These "default" concrete methods are only NonStatic methods.

Coding Rule:

=>These "default" concrete methods are available to implementation classes and which can be accessed with implementation_class object_reference.

(c)private concrete methods(Java9):

=>The concrete methods which are declared in interfaces using "private" keyword are known as private concrete methods.

=>These private concrete methods are introduced by Java9 version.

=>These private concrete methods are categorized into two types:

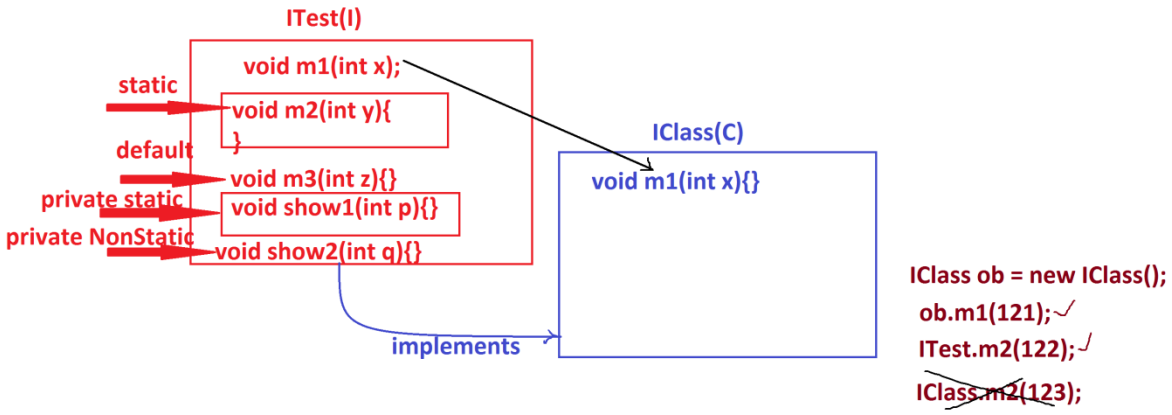
(i)static private concrete methods

(ii)NonStatic private concrete methods

Coding Rule:

=>private concrete methods are accessed only inside the interface, which means private concrete methods are accessed by the NonPrivate methods of same interface.

Diagram:



Ex:

ITest.java

```
package test;
public interface ITest {
    public abstract void m1(int x);
    public static void m2(int y) {
     System.out.println("====static concrete m2(y)====");
     System.out.println("The value y:"+y);
    }
    default void m3(int z,int p,int q) {
     System.out.println("====default concrete m3(z)====");
     System.out.println("The value z:"+z);
     ITest.show1(p);
     this.show2(q);
    }
    private static void show1(int p) {
     System.out.println("====private static show1(p)====");
     System.out.println("The value p:"+p);
    }
    private void show2(int q) {
     System.out.println("====private Non-static show2(q)====");
     System.out.println("The value q:"+q);
    }
}
```

IClass.java

```
package test;
public class IClass implements ITest{
    public void m1(int x) {
     System.out.println("====method m1(x)====");
     System.out.println("The value x:"+x);
    }
}
```

DemoInterface4.java(MainClass)

```
package maccess;
import test.*;
public class DemoInterface4 {
     public static void main(String[] args) {
         IClass ob = new IClass();
         ob.m1(121);
         ITest.m2(122);
         //IClass.m2(123);//Error
         ob.m3(124,125,126);
     }
}
```

o/p:

====method m1(x)====

The value x:121

====static concrete m2(y)====

The value y:122

====default concrete m3(z)====

The value z:124

====private static show1(p)====

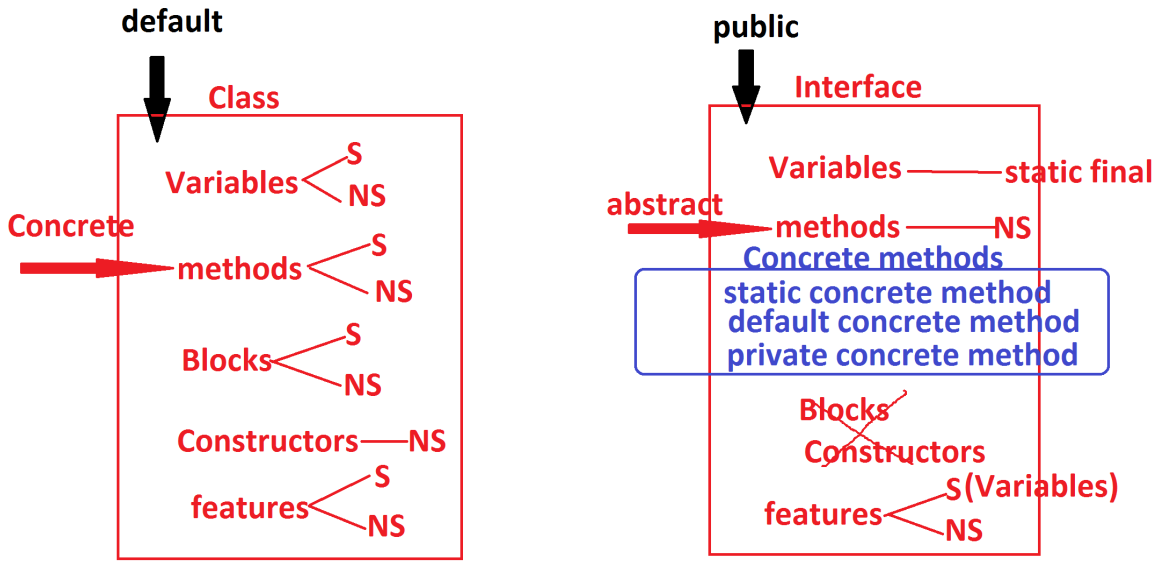
The value p:125

====private Non-static show2(q)====

The value q:126

==========

Comparision Diagram:



==========