java8 Interface inside methods

untill 1.7 onwords

Every method present inside interface is always: public and abstract whether we declared or not.

Ex:
```
 void m1( );
public void m1( );
abstract void m1( );
public abstract void m1( );
```

from 1.8 version

<u>Related to methods</u>

from 1.8 version inside interface default methods + static methods are allowed.
from 1.9 version inside interface private method are allowed.

<u>Related to Variables</u>

Every variable present inside interface is always

public static final

java 1.7
1.8
1.9 **<u>there is no Enhancements related to the variables on new version of java.</u>**

1)Default Method | Virtual Extension Method | Defender Method

Ex:
```
interface interf{

public void m1();

public void m2();

// public void m3();

}
```

if we are trying to add a new method inside interface all the implemented classes are affected.

problem::::

if we want change the functionality of this interface by adding a new method, all the implemented classes are affected.

without effecting the implemented classes if we are trying to add a new method inside interface then we should go for Default methods.

"Default methods already having implementation"

Ex:

```
interface inf{

public void m1( );

public void m2( );

default void m3( ) {

System.out.println("default implementation");

}
```

Note:
if we are not satisfy with the default implementation then we can override the default method implementation.

Q)Why the word is Default ?

it is not a modifier because this default method having default implementation that's way the word default came.
if we are not declare any modifier then it is default modifier

Q)While overriding default method compulsary we should use public only ?

because of default methods can't declare inside class ,we can declare inside interface only
default ----> public only

Inside class------modifier default not allowed here.

NOTE::
Object class methods we can't declare as default methods.

Class vs interface

Class is a heavy weight because of Object creation and it contain Constructor, static block, instance block
interface is light weight because of never contain constructor, static block, instance black

Q)When we should go for class ?

Q)Why interface support static methods ?

because of this method no way related to Object.

Q)How many ways we can call interface static methods?

by default interface static method not available for the implementation class

Ex:

```
public interface inter {
public static void m1() {
System.out.println("interface static method");
}
}

public class InterfaceStatic implements inter{

public static void main(String[] args) {

inter.m1(); //1   TRUE

m1( );//2  FALSE

InterfaceStatic t=new InterfaceStatic( );
t.m1( );//3 FALSE

InterfaceStatic.m1( );//4 TRUE


}
}
```

NOTE: Interface static methods by default not available to the implementation class,we can call interface static method by using interface name only.

By mistake if we are using implementation class name or implementation Object reference or if we are call directly we will get compile time error.

Ex:

```
Interface  interf{
public static void main(String[ ] args ){

System.out.println("inside interface static methods are allowed");
 }
}
```