26/3/2024

Interface

⇒ Just like classes

⇒ One interface multiple implementation.

⇒ For using multiple inheritance in java.

Access
- modifiers
- Others

```
modifiers interface Interface_Name {
    modifier datatype var1 = value1;
    modifiers return_type method_name1 (arg list);
    modifiers return_type method_name2 (arg list);
}
```

- Access modifiers with Interface (Top-Level) ⇒
  - Public
  - Default

  } Just like top-level Class

- Other modifiers ⇒
  - abstract
    {- By default all the interfaces are abstract.}

Methods of Interfaces ⇒

⇒ By default all the methods of interface are public & abstract.

⇒ from JDK-8 onwards you can provide implementation of methods in interface as default implementation.

⇒ from JDK-8 onwards we can define Static methods in interfaces.

⇒ from JDK-9, we can have private methods in interfaces.

Data members of Interfaces ⇒

⇒ By default, all data members of Interface are public, static & final.

⇒ Data members must be initialised at the time of declaration

Other Points

⇒ A class which inherits interface, have to use 'Implements' keyword.

⇒ An interface can inherit another interface by using 'extends' keywords

⇒ A class can inherit more than one interface at a time.

⇒ You cannot create object of an interface but you can declare reference of type of that interface.

---

default & abstract

```
interface MyInt {
    Public abstract {
        Void m1();
        Void m2();
    }
}

Class A implements MyInt {
    Public Void m1() {
        SoP("M1 of A");
    }
    Public Void m2() {
        SoP("M2 of A");
    }
}
```

as both methods in interface m1, m2 are public we have to make these Overridden methods Public as well.

```
interface MyInt {
    Void m1();
    Void m2();
}

abstract Class A implements MyInt {
    Public Void m1() {
        ~~void~~
    }
    Void m3() {
    }
}
```

This class must be abstract as we have not defined m2().

```
Class B extends A {
    Public Void m2() {
    }
}
```

28/3/2024

```
interface I1{
    void m1();
}
interface I2{
    void m2();
}
```

```
interface I1{
    void m1();
}
interface I2 extends I1{
    void m2();
}
```

```
interface I1{
    void m1();
}
interface I2{
    void m2();
}
interface I3 extends I1, I2{
    void m3();
}
```

```
Class A implements I1, I2{
    Public Void m1(){
    }
    Public Void m2(){
    }
    Void m3(){
    }
}
```

```
Class A implements I2{
    Public Void m1(){
    }
    Public Void m2(){
    }
}
```

```
Class A implements I3{ /
Class A implements I3, I1 /
Class A implements I3, I2
```

will need to implement all three same methods in class A.

```
interface I1{
    void m1();
    void m2();
}
interface I2{
    void m1();
    void m3();
}
```

```
interface I1{
    void m1();
    void m2();
}
interface I2{
    void m1();
    void m2(int i);
}
```

```
interface I1{
    void m1();
}
Class A implements I1{
    P.V. m1(){
    }
}
```

```
Class A implements I1, I2{
    P.V. m1(){
    }
    P.V. m2(){
    }
    P.V. m3(){
    }
}
```

Three methods needed to be implemented in class.

m1
m2
m1(int i)

```
Class B implements I1{
    P.V. m1(){
        SoP("M1 of B");
    }
}
Class C{
    PSVM(_){
        I1 i = new A();
        i.m1();
        i = new B();
        i.m2();
    }
}
```

Output
// M1 of A
// M1 of B

{Same code as Previous example}

```
Class C {
    PSVmain(){
        I1 i = new A();
        *A a2;
        *a2 = i;  ×
        a2 = (A)i;  ✓
    }
}
```

```
Interface I1{
    void m1();
}

Class A implement I1{
    Pvm1(){
        SOP("M1 of A");
    }
    Void m2(){
        SoP("M2 of A");
    }
}
```

```
Class B {
    PSVMC_){
        I1 i = new A();
        i.m1();
        i.m2(); //error
    }
}
```

{ as M2 is not declared in I1 }

```
interface I1{
    int a1 = 1;
    Void m1();
}
Class A implements I1{
    P V m1(){ }
```

```
PS VM(){
    SoP(a);
}
}
```

```
Interface I1{
    int a = 1;
    void m1();
}
interface I2{
    int a = 3;
    int b = 2;
}
```

```
Class A implements I1, I2{
    P V m1(){ }
    PSVM(_){
        SOP(a);  ×  //error
        SoP(b);
    }
}
```

{ SoP(I1.a); SoP(I2.a); }

Q1 Which one of following is true about interfaces :->

① All the methods of interfaces are abstract.

② All the methods of interfaces are static

③ All the methods of interfaces are final

✓④ All the methods of interfaces of public by default.

Q => Which of the following is ~~false~~ false about interfaces :->

✓⑤ implementation of method can be provided by in interface

② All the methods of an interface are ~~static~~ abstract by default

⑤ All the data members of an interface are public, static & final

④ we cannot create an object of an interface.