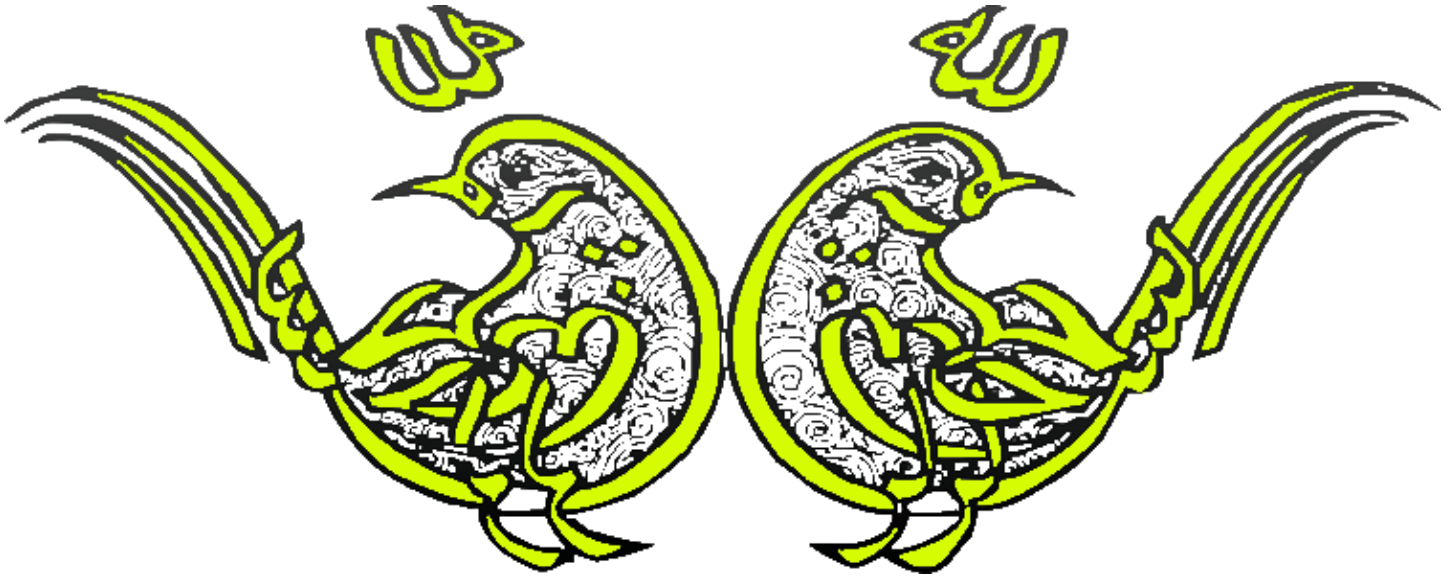

تقدیم به همه هموطنان عزیزم

مقالات آموزشی جاوا

تفاوت بین کلاس Abstract و Interface در جاوا

نویسنده : رحمان زارعی

جاوا را ساده،آسان و شیرین بنوشید!!!!

ما در سلسله آموزش های ساده جاوا در جلسه بیست و پنجم مبحث **Abstraction(انتزاع)** و در جلسه بیست و هفتم مبحث **Interface** را بررسی کردیم. قبل از مطالعه این مقاله دو مبحث مذکور را مطالعه کنید.

۱. **جلسه بیست و پنجم مبحث Abstraction(انتزاع)**
۲. **جلسه بیست و هفتم مبحث Interface**

در این مقاله قصد داریم با هم در مورد تفاوت های میان کلاس Abstract و Interface با مثال بحث کنیم.

کلاس Abstract	Interface
۱. کلاس Abstract می تواند **متدهای انتزاعی (abstract)** و **غیرانتزاعی (non-abstract)** داشته باشد.	۱. Interface تنها می تواند **متدهای انتزاعی(abstract)** داشته باشد.
۲. کلاس Abstract وراثت چندگانه را ساپورت نمی کند.	۲. Interface از **وراثت چندگانه** حمایت می کند.
۳. کلاس Abstract می **تواند متغیرهای final و غیر final ، static و غیر static** داشته باشد.	۳. Interface تنها می تواند **متغیرهای final و static** داشته باشد.
۴. کلاس Abstract می تواند **به interface ، implements شود.**	۴. Interface **نمی تواند** به کلاس **abstract ، implements** شود.
۵. کلمه کلیدی abstract برای اعلان کلاس abstract استفاده می شود.	۵. **کلمه کلیدی interface** برای اعلان یک interface استفاده می شود.
۶. کلاس Abstract می تواند **یک کلاس یا یک کلاس abstract را به ارث ببرد.** به عبارت دیگر کلاس Abstract **تنها می تواند به یک کلاس یا کلاس Abstract دیگر extends شود.**	**۶.** Interface **می تواند هر تعداد Interface که دوست را به ارث ببرد.** **به عبارت دیگر** Interface **می تواند به هر تعداد Interface** دیگر **extends** شود.
۷. **یک کلاس می تواند تنها یک کلاس abstract را به ارث ببرد.**	۷. یک کلاس می تواند به **هر تعداد** Interface را **implements** کند.

مثال:
`public abstract class Shape{`
`public abstract void draw();`
`}`
	مثال:
	`public interface Drawable{`
	`void draw();`
	`}`

- هر یک از تفاوت های میان کلاس Abstract و Interface در جاوا را که در جدول بالا بهش پرداختیم با مثال بررسی می کنیم:

مورد۱ :

```
abstract class Cat {

        // کلاس Abstract می تواند متدهای انتزاعی (abstract) و غیرانتزاعی (non-abstract) داشته باشد.

      public abstract void sleep();

      public void run() {
            System.out.println("Cat is runing...");
      }

}

interface Dog {

                                    // Interface تنها می تواند متدهای انتزاعی(abstract) داشته باشد.

      public void sleep();

      public void run();
}
```

مورد۲:

```
// کلاس Abstract وراثت چندگانه را ساپورت نمی کند.

abstract class A {

}

abstract class B extends A {

}


                                    //Interface از وراثت چندگانه حمایت می کند
```

```
interface C {

}

interface D {

}

interface E extends C, D {

}
```

مورد۳:

```
// کلاس Abstract می تواند متغیرهای final و غیر final ، static و غیر static داشته باشد.

abstract class Example1{
   private int numOne=10;
   protected final int numTwo=20;
   public static final int numThree=500;
   public void display1(){
      System.out.println("Num1="+numOne);
   }
}
class Example2 extends Example1{
   public void display2(){
      System.out.println("Num2="+numTwo);
      System.out.println("Num2="+numThree);
   }
}
class Demo{
   public static void main(String args[]){
      Example2 obj=new Example2();
      obj.display1();
      obj.display2();
   }
}
```

مورد۳:

```
//Interface تنها می تواند متغیرهای final و static داشته باشد.


interface Example1{
```

```
   int numOne=10;
}
class Example2 implements Example1{
   public void display1(){
      System.out.println("Num1="+numOne);
   }
}
class Demo{
   public static void main(String args[]){
      Example2 obj=new Example2();
      obj.display1();
   }
}
```

مورد۴:

```
interface A {

}
```

// کلاس Abstract می تواند به **interface** ، **implements** شود.

// Interface **نمی تواند** به کلاس **abstract** ، **implements** شود.

```
abstract class B {

}
```

مورد۵:

```
interface A {

}
```

// **کلمه کلیدی abstract برای اعلان کلاس abstract استفاده می شود.**

// **کلمه کلیدی interface** برای اعلان یک **interface** استفاده می شود.

```
abstract class B {

}
```

- مورد ۶ همان مورد ۲ می باشد که به گونه دیگر بیان کردیم.

مورد۷:

```
interface A {

}

interface B {

}

interface C {

}

interface D {

}

abstract class E {

}

// یک کلاس می تواند تنها یک کلاس abstract را به ارث ببرد.
// یک کلاس می تواند به هر تعداد Interface را implements کند.
class F extends E implements A, B, C, D {

}
```

پیروز و موفق باشید

سایت آموزش زبان جاوا به زبان ساده،آسان و شیرین!!!

www.JAVAPro.ir

آموزش جاوا SE را با تجربه شخصی و به زبان خودمونی یاد بگیرید!!!!!

بازدید ازکانال

بازدید از سایت

هر روز مفاهیم و مثال های جدید به سایت اضافه می شود برای اطلاع از مطالب جدید روی سایت عضو کانال شوید.

دخل و تصرف ، ویرایش و کپی زدن تمامی آموزش های جاوالایک به دور از اخلاق حرفه ای ست و حرام می باشد.