JAVA 中 extends 与 implements 有啥区别?

1. 在类的声明中,通过关键字 extends 来创建一个类的子类。一个类通过关键字 implements 声明自己使用一个或者多个接口。

extends 是继承某个类, 继承之后可以使用父类的方法, 也可以重写父类的方法;

implements 是实现多个接口, 接口的方法一般为空的, 必须重写才能使用

2.extends 是继承父类, 只要那个类不是声明为 final 或者那个类定义为 abstract 的就能继承, JAVA 中不支持多重继承, 但是可以用接口 来实现, 这样就要用到 implements, 继承只能继承一个类, 但 implements 可以实现多个接口, 用逗号分开就行了

比如

class A extends B implements C,D,E

==========

implements

学了好久,今天终于明白了 implements,其实很简单,看下面几个例子就 ok 啦~~

接口的一些概念

```
public inerface Runner
{
   int ID = 1;
   void run ();
}


interface Animal extends Runner
{
   void breathe ();
}

class Fish implements Animal
{
   public void run ()
{
     System.out.println("fish is swimming");
}

public void breather()
{
     System.out.println("fish is bubbing");
}
}

abstract LandAnimal implements Animal
{

   public void breather ()
```

```
{
     System.out.println("LandAnimal is breathing");
}
}

class Student extends Person implements Runner
{
     ......
     public void run ()
       {
                 System.out.println("the student is running");
       }
     ......
}



interface Flyer
{
   void fly ();
}

class Bird implements Runner , Flyer
{
   public void run ()
     {
             System.out.println("the bird is running");
     }
   public void fly ()
     {
             System.out.println("the bird is flying");
     }
}


class TestFish
{
   public static void main (String args[])
     {
           Fish f = new Fish();
           int j = 0;
           j = Runner.ID;
           j = f.ID;
     }
```

}
接口实现的注意点：

a.实现一个接口就是要实现该接口的所有的方法(抽象类除外)。
b.接口中的方法都是抽象的。
c.多个无关的类可以实现同一个接口，一个类可以实现多个无关的接口。
==========
extends 与 implements 的不同

extends 是继承父类，只要那个类不是声明为 final 或者那个类定义为 abstract 的就能继承，JAVA 中不支持多重 继承，但是可以用接口来实现，这样就要用到 implements，继承只能继承一个类，但 implements 可以实现多个接口，用逗号分开就行了
比如
class A extends B implements C,D,E

//
一个类通过关键字 implements 声明自己使用一个或者多个接口。在类的声明中，通过关键字 extends 来创建一个类的子类。
class 子类名 extends 父类名 implenments 接口名
{...

}

==========

A a = new B(); 结果 a 是一个 A 类的实例，只能访问 A 中的方法，那么又和 A a = new A(); 有什么区别呢？
==========
class B extends A
继承过后通常会定义一些父类没有的成员或者方法。
A a = new B();
这样是可以的，上传。
a 是一个父类对象的实例，因而不能访问子类定义的新成员或方法。
==========
假如这样定义：
```
class A{
int i;
void f(){}
}
class B extends A{
int j;
void f(){}//重写
void g(){}
}
```
然后：

B b = new B();
b 就是子类对象的实例，不仅能够访问自己的属性和方法，也能够访问父类的属性和方法。诸如 b.i,b.j,b.f(),b.g()都是合法的。**此时 b.f()是访问的 B 中的 f()**
A a = new B();
a 虽然是用的 B 的构造函数，但经过 upcast，成为父类对象的实例，不能访问子类的属性和方法。a.i,a.f()是合法的，而 a.j,a.g()非 法。此时访问 a.f()是访问 B 中的 f()
==========
A a = new B(); 这条语句，实际上有三个过程：
(1) A a;
将 a 声明为父类对象，只是一个引用，未分配空间
(2) B temp = new B();
通过 B 类的构造函数建立了一个 B 类对象的实例，也就是初始化
(3) a = (A)temp;
将子类对象 temp 转换未父类对象并赋给 a，这就是上传(upcast)，是安全的。
经过以上 3 个过程，a 就彻底成为了一个 A 类的实例。
子类往往比父类有更多的属性和方法，上传只是舍弃，是安全的；而下传(downcast)有时会增加，通常是不安全的。
==========
a.f()对应的应该是 B 类的方法 f()
调用构造函数建立实例过后，对应方法的入口已经确定了。
如此以来，a 虽被上传为 A 类，但其中重写的方法 f()仍然是 B 的方法 f()。也就是说，每个对象知道自己应该调用哪个方法。
A a1 = new B();
A a2 = new C();
a1,a2 两个虽然都是 A 类对象，但各自的 f()不同。这正是 1 楼说的多态性的体现。

这类问题在《Java 编程思想》上都讲的很清楚

implements 一般是实现接口。
extends 是继承类。

接口一般是只有方法声明没有定义的，
那么 java 特别指出实现接口是有道理的，因为继承就有感觉是父类已经实现了方法，而接口恰恰是没有实现自己的方法，仅仅有声明，也就是一个方法头没有方法体。因此你可以理解成接口是子类实现其方法声明而不是继承其方法。
但是一般类的方法可以有方法体，那么叫继承比较合理。
引入包可以使用里面非接口的一切实现的类。那么是不是实现接口，这个你自己决定，如果想用到那么你不是实现，是不能调用这个接口的，因为接口就是个规范，是个没方法体的方法声明集合。我来举个例子吧：接口可以比作协议，比如我说 一个协议是“杀人”那么这个接口你可以用 砍刀去实现，至于怎么杀砍刀可以去实现，当然你也可以用抢来实现杀人接口，但是你不能用杀人接口去杀人，因为杀人接口只不过是个功能说明，是个协议，具体怎么干，还要看他的实现类。

那么一个包里面如果有接口，你可以不实现。这个不影响你使用其他类。

implements

implements 是一个类实现一个接口用的关键字，他是用来实现接口中定义的抽象方法。比如:people 是一个接口,他里面有 say 这个方法。public interface people(){ public say();}但是接口没有方法体。只能通过一个具体的类去实现其中的方法体。比如 chinese 这个类，就实现了 people 这个接口。 public class chinese implements people{ public say(){System.out.println("你好！");}}

在 java 中 implements 表示子类继承父类,如类 A 继承类 B 写成 class A implements B{}

与 Extends 的不同

extends， 可以实现父类，也可以调用父类初始化 this.parent()。而且会覆盖父类定义的变量或者函数。这样的好处是：架构师定义好接口，让工程师实现就可以了。整个项目开发效率和开发成本大大降低。

implements，实现父类，子类不可以覆盖父类的方法或者变量。即使子类定义与父类相同的变量或者函数，也会被父类取代掉。

这两种实现的具体使用，是要看项目的实际情况，需要实现，不可以修改 implements，只定义接口需要具体实现，或者可以被修改扩展性好，用 extends。