# 接口的基本定义

接口:一个类里只包含有抽象类和全局常量,连构造函数也没有

|  |
| --- |
| interface A{ //A接口  public abstract void print(); //抽象方法  public static final String MSG = "hello,world"; //全局常量  } |

特点:

1. 由于存在抽象方法,所以不能直接实例化,可以通过向上转型实例化
2. 接口必须要有子类,子类要使用implements可以实现多个接口
3. 接口的子类如果不是抽象类,必须要覆写接口中的全部抽象方法,如果是抽象类则随意
4. 如果子类既要实现接口也要继承类(抽象类),那么先extends 在implements
5. 接口里只能有一种访问权限public,对于抽象方法可以省略public abstract,对于全局常量可以省略public static final, 最好都不省略
6. 接口能够继承(extend)多个接口，并且只能继承接口,不能继承抽象类和普通类,也不能实现接口
7. 接口里能定义内部类(普通,抽象,接口),如果这些类加上了static就成了外部的了

**接口的三大核心作用:**

1. **定义不同层之间的操作标准**
2. **表示一种操作的能力**
3. **表示将服务器的远程方法视图暴露给客户端**

**范例:子类实现多个接口:**

|  |
| --- |
| public class test{  public static void main(String args[]){  X x = new X();  A a = x; //向上转型  B b = x;//向上转型  a.print();  b.get();  }  }  interface A{ //A接口  public abstract void print(); //抽象放啊  public static final String MSG = "hello,world"; //全局常量  }  interface B{  public abstract void get();  }  class X implements A,B{  @Override  public void print() {  System.out.println("A接口的方法");  }  @Override  public void get() {  System.out.println("B接口的方法");  }  } |

# 接口定义增强(JDK1.8才有的)

## 2.1default

定义:

default和static有冲突,在接口的方法里添加deafult,则接口的这个方法要写完整,子类将会继承这个方法

场景:如果一个接口有很多个子类,这些子类需要增加一个相同的方法,很明显到每一个子类里一个个去添加不现实,这时使用在接口里添加一个default方法,子类就相当于继承了这个方法,当然子类还可以覆写这个方法

|  |
| --- |
| public class test{  public static void main(String args[]) throws Exception {  X x = new X();  x.test();  }  }  interface A{  public void print();  public default void test(){ //default和static有冲突  System.out.print("hello,world");  }  }  class X implements A{  @Override  public void print() {  System.out.print("hahaahah");  }  } |

## 2.2static

接口里还可以用static定义静态方法,方法要写完整

|  |
| --- |
| public class test{  public static void main(String args[]) throws Exception {  A.print();  }  }  interface A{  public static void print(){ //静态方法  System.out.print("hello,world");  }  } |