Ch6 接口

主　　讲：张春元（计算机学院309室）
联系电话：13876004640
课程邮箱：haidajava@126.com
密　　码：zhangchunyuan88

Ch6 接口

在上一章中我们知道Java对类的继承只支持单重继承，而不支持多重继承，而且子类和父类之间是“强是”关系，如Student is a Person，但现实世界中却存在多重继承现象，又该如何描述呢？

此外，在现实世界中一只鸡和一个苹果是可以作比较的，但鸡显然不是苹果，苹果显然也不是鸡，如果在Java中需要比较鸡和苹果，又该如何描述呢？

我们可以通过引入接口的定义来解决这一问题。

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.1 接口

■ 接口

接口在结构上与抽象类有些相似，接口之间也可以继承，每个接口经编译后为独立的字节码文件。但接口内部只能包含公有静态常量和公有抽象实例方法（？），而抽象类还可以包含变量和其它类型的方法。因而接口在定义形式和处理方法上与抽象类还是有一些不同。

- ✓ 6.1.1 接口的定义
- ✓ 6.1.2 接口的继承

6.1 接口

6.1.1 接口的定义

接口的定义格式为：

```
[public] interface  接口名称{
   //静态常量及抽象方法
}
```

其中：方括号表示可省略部分。访问属性控制符public与用于修饰类的public意义一致，如果省略public则就是第4章所介绍的默认访问控制属性。interface是用于定义接口的关键字，常量和抽象方法的声明放在一对大括号中。

6.1 接口

注意：在Java接口中，编译器将常量的定义默认为public static final类型的静态常量，不论是否使用了这些修饰符，它都是这样处理的。所以在定义常量时，可以只给出其数据类型说明和常量名，同时，定义时要为每个常量都赋值。因为成员方法都是抽象的，在定义成员方法时也可以省略关键字abstract，它默认也是抽象的。

```
public interface T{
      public static final int  K=1;
      public abstract void p();
}
```

等价于

```
public interface T{
      int  K=1;
      void p();
}
```

6.1 接口

✓ 接口定义注意事项

- ☞ 接口内方法的定义必须是公有和抽象的，如果没有包括这些限定符，它们将被自动转换为公有和抽象的。
- ☞ 不能在接口内将方法声明为私有（private）或保护的(protected)。
- ☞ 接口内定义的常量必须声明为公有、静态和final，或者不使用限定符。

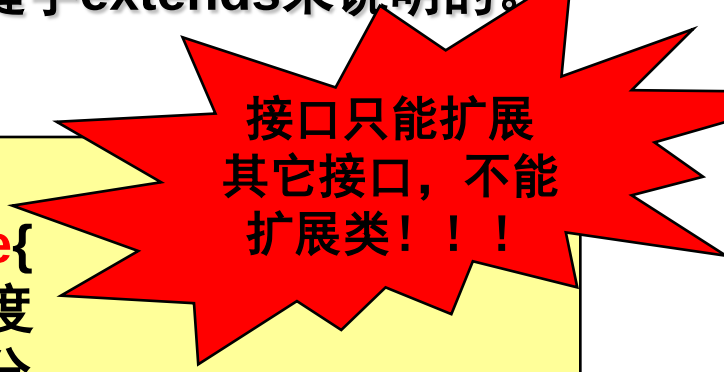
6.1 接口

■ 6.1.2 接口的继承

接口和类一样，也可以继承。不过，**类仅支持单继承，而接口既支持单继承，也支持多重继承**。通过继承，一个接口可以继承父接口中的所有成员。接口之间的继承也是通过关键字extends来说明的。

接口的多继承示例：

```
public interface SubMathInterface
 extends MathInterface, PhysicalInterface{
    double minuteToRadian();//分转换为弧度
    double RadianToMinute();//弧度转换为分
    double secondToRadian();//秒转换为弧度
    double radianToSecond();//弧度转换为秒
}
```



通过继承，在子接口SubMathInterface中不仅有此处定义的四个方法，而且也继承了父接口MathInterface、PhysicalInterface中的所有常量和方法。

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.2 接口的实现

类不能扩展接口，
只能实现接口！！

■ 接口的实现

定义抽象类也好，定义接口也好，都是为了使用。要使抽象类发挥功能，必须通过抽象类扩展出一个非抽象子类，并在子类中覆盖掉父类中的所有抽象方法来实现。但是，如何使接口发挥其功能呢？显然通过扩展子接口是无法完成的，因为扩展出的子接口还是接口，同样不能实例化对象。

在Java中，要让接口发挥其功能，需定义一个普通的类，在这个类中**要覆盖掉接口中的所有方法**，以便将其实现，称为**该类对接口的实现**，实现接口是通过关键字**implements**来说明的。

注意：定义一个类来实现接口时，需要在类中覆盖掉接口中的所有方法，不能有选择地实现其中的某些方法，否则只能将这个类定义成一个抽象类。（教材P146-148例子1及相应的UML图、TestEdible.java）

6.2 接口的实现

✌ 一个类在实现多个接口时，如果两个接口中定义了名称相同的方法，可以分三种方式来处理：

1. 如果两个方法的方法头相同，则可在类中用一个方法进行实现，其定义可满足两个接口；
2. 如果两个方法只是名称相同，参数不同，则在类中分别进行重载即可；
3. 如果两个方法方法名称、参数列表相同，仅是返回值类型不同，则在类中无法创建一个能满足两个接口的方法，需要对所实现的接口进行修改。

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.3 接口的UML图

■ 接口的UML图

接口的UML图和表示类的UML图类似，使用一个长方形描述一个接口的主要构成，将长方形垂直地分为三层。

- 顶部第1层是名字层，接口的名字必须是斜体字形，而且需要用<<interface>>修饰名字，并且该修饰和名字分列在两行。
- 第2层是常量层，列出接口中的常量及类型，格式是“+常量名字：类型”。
- 第3层是方法层，也称操作层，列出接口中的方法及返回类型，格式是“+方法名字（参数列表）：类型”。

6.3 接口的UML图

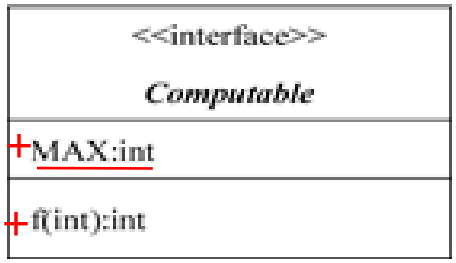


图 6.2　接口 UML 图

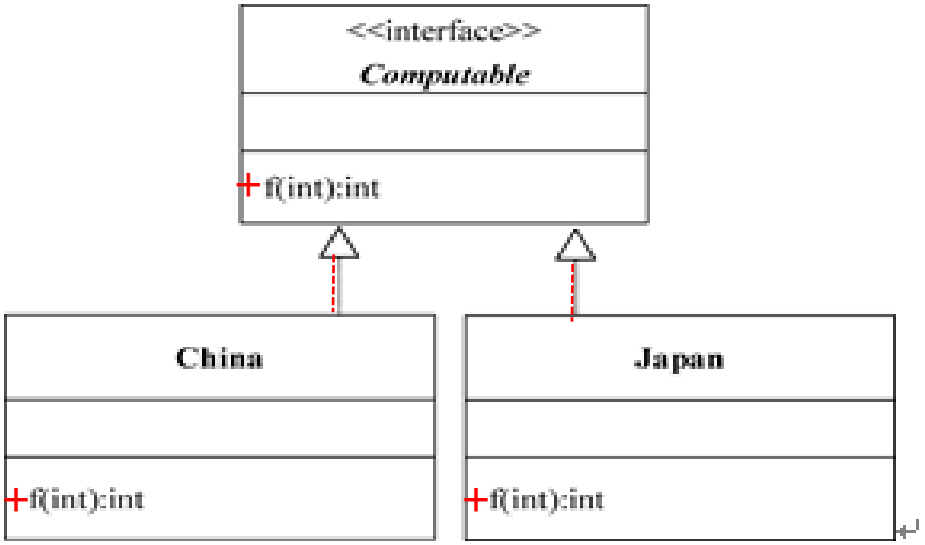


图 6.3　实现关系的 UML 图

实现接口的类用虚线空心箭头指向接口，实现抽象类的子类用实线空心箭头指向抽象类。注：继承关系一律用实线箭头（包括父子接口、父子类都是如此）

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.4 接口回调

■ 接口回调（有些类似于上章讲的上转型对象）

接口回调是指：可以把实现某一接口的类创建的对象的引用赋给该接口声明的接口变量中，那么该接口变量就可以调用被类重写的接口方法。需要注意的是：接口变量不能调用实现该接口的类中的其它非此接口的方法。

例： Com com;//声明接口对象

ImpleCom obj= new ImpleCom();//实现接口子类对象

com = obj; //接口回调

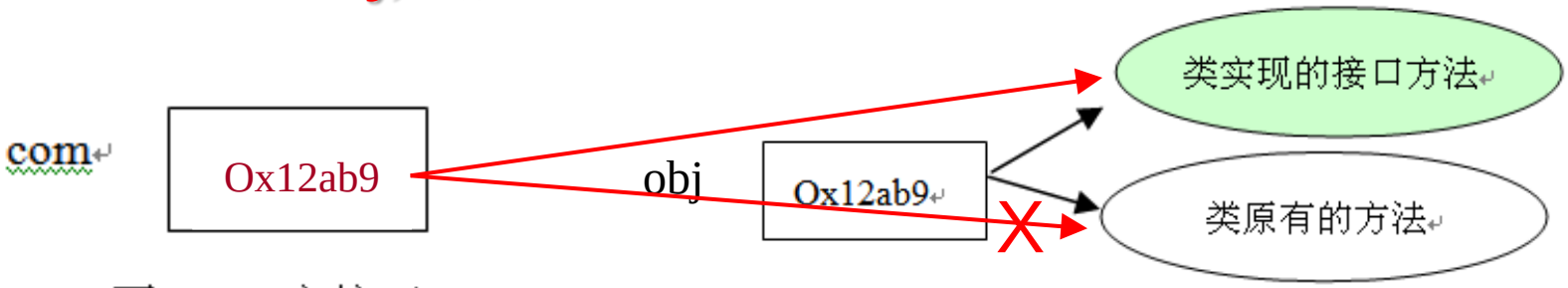


图 6.5 空接口

图 6.6 对象调用方法的内存模型

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.5 理解接口

■ 什么时候使用接口？

接口和抽象类都可以抽象出重要的行为标准，该行为标准用抽象方法来表示。那什么情况下使用接口？

子类和父类之间是“强是(is-a)”关系，明显的父子关系用类来模拟；类与接口之间是“弱是(like-a)”关系，一般用接口来定义不相关的多个类的共有属性(并不要求这些类间关系是父子类关系)。此外，通过接口还可以来实现类间所不具备的多重继承关系。

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.6-6.7 接口与多态、接口参数

接口的多态性

包含多态性（父接口有多个子接口、接口在不同类中可能具有不同的实现形式）

接口回调

例：

教材P153 Example6_4.java
教材P154 Example6_5.java

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结
- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

6.8 abstract类与接口的比较

- 接口与抽象类的区别
  - 数据域区别：在接口中只能定义常量；抽象类的数据域则既可定义常量、也可定义变量。
  - 方法的区别：接口中的方法必须是公有抽象的实例方法，没有其它类型的方法（包括构造方法也不存在）；抽象类中的方法没有限制（存在构造方法，但不能用其实例化对象），但其抽象方法必须是可访问的抽象实例方法。
  - 继承的区别：接口之间可以存在多重继承，接口不能继承类；类之间只能单重继承，并且可以实现多个接口。

第6章 本章导读

- 主要内容
  - 6.1 接口
  - 6.2 实现接口
  - 6.3 接口的UML图
  - 6.4 接口回调
  - 6.5 理解接口
  - 6.6 接口与多态
  - 6.7 接口参数
  - 6.8 abstract类与接口的比较
  - 6.9 面向接口的编程
  - 6.10 应用举例
  - 6.11 小结

  } 自己看

- 重点与难点
  - 重点：接口的理解；抽象类和接口的区别。
  - 难点：抽象类和接口的异同点。

其它：UML图总结

- 各种关系
- 继承关系：实线空心箭头（子类指向父类、子接口指向父接口）。
- 实现关系：虚线空心箭头（类指向接口）
- 关联关系：实线非空心箭头（A类中的某成员变量是用B类声明的，称作A关联于B或者A是用B部分组合成的，用A→B表示，详参教材P76）
- 依赖关系：虚线非空心箭头（A类中的某些形参是用B类声明的或者某个方法的返回值类型是B类，称作A依赖于B，用A ⇢ B表示，详参教材P76）

其它：开闭原则

- **开闭原则（教材5.11节）**
  - ✓ 所谓“开-闭原则”(Open-Closed Principle)就是让设计的系统应当对扩展开放，对修改关闭。
  - ✓ 在设计系统时，应当首先考虑到用户需求的变化，将应对用户变化的部分设计为对扩展开放，而设计的核心部分是经过精心考虑之后确定下来的基本结构，这部分应当是对修改关闭的，即不能因为用户的需求变化而再发生变化，因为这部分不是用来应对需求变化的。
  - ✓ 如果系统的设计遵守了“开-闭原则”，那么这个系统一定是易维护的，因为在系统中增加新的模块时，不必去修改系统中的核心模块。

第6章 作业

- **本章作业** (做在五毛钱的小本子，不得做在信笺或草稿纸上，也不得做在厚且大的本子上，写上班级学号)
  - **P158-160　1、2、3**


把结果看
成快乐
的个
一
时
快
乐；
把过程看
成快乐的个
一生快乐