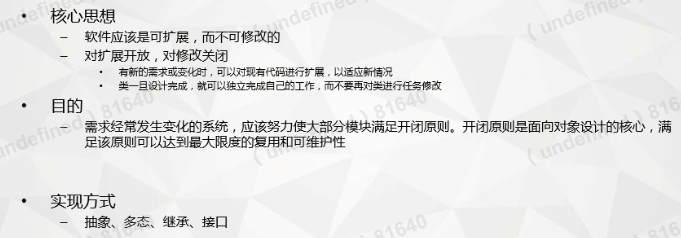
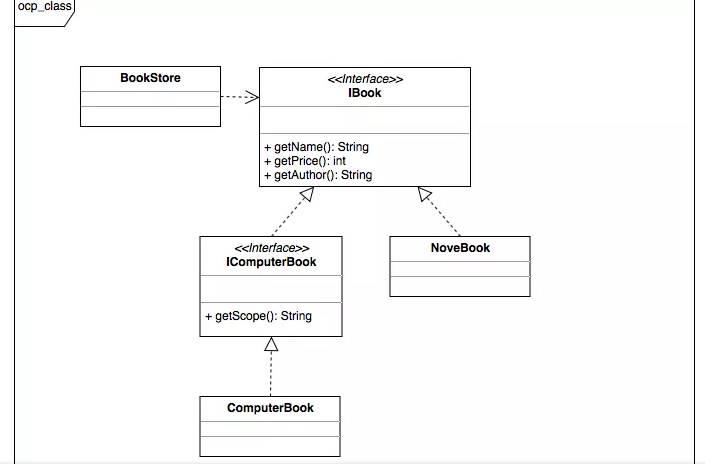
**开闭原则**



0.1622018.05.07 22:07:44字数 305阅读 5934

**1、定义**

**开闭原则，对于扩展是开放的，对于修改是关闭。**



**2、实现**

**（1）创建书籍接口**

interface IBook {

String getName();

int getPrice();

String getAuthor();

}

**（2）创建计算机书籍接口**

interface IComputerBook extends IBook {

public String getScope();

}

**（3）实现小说类**

class NovelBook implements IBook {

private String name;

private int price;

private String author;

public NovelBook(String name, int price, String author) {

this.name = name;

this.price = price;

this.author = author;

}

@Override

public String getName() {

return name;

}

@Override

public int getPrice() {

return price;

}

@Override

public String getAuthor() {

return author;

}

}

**（4）实现计算机书籍类**

class ComputerBook implements IComputerBook {

private String name;

private int price;

private String author;

private String scope;

public ComputerBook(String name, int price, String author, String scope) {

this.name = name;

this.price = price;

this.author = author;

this.scope = scope;

}

@Override

public String getName() {

return name;

}

@Override

public int getPrice() {

return price;

}

@Override

public String getScope() {

return scope;

}

@Override

public String getAuthor() {

return author;

}

}

**（4）实现书店类**

class BookStore {

private final static ArrayList<IBook> sBookList = new ArrayList<IBook>();

static {

sBookList.add(new NovelBook("天龙八部", 3200, "金庸"));

sBookList.add(new NovelBook("巴黎圣母院", 5600, "雨果"));

sBookList.add(new NovelBook("悲催世界", 3500, "雨果"));

sBookList.add(new NovelBook("金瓶梅", 4300, "兰陵笑笑生"));

sBookList.add(new ComputerBook("Think in Java", 5800, "Bruce Eckel", "编程语言"));

}

public static void main(String[] args) throws IOException {

NumberFormat format = NumberFormat.getCurrencyInstance();

format.setMaximumFractionDigits(2);

Log.d("JimmyZhang", "----书店卖出去的书籍记录如下---");

for (IBook book : sBookList) {

Log.d("JimmyZhang", "书籍名称:" + book.getName()

+ "\t书籍作者:" + book.getAuthor()

+ "\t书籍价格:" + format.format(book.getPrice() / 100.00) + "元");

}

}

}

**3、原则**

**（1）通过接口或抽象类约束扩展，对扩展进行边界限定；**

**（2）参数类型、引用对象尽量使用接口或者抽象类，而不是实现类；**

**（3）抽象层尽量保持稳定，一旦确定就不允许修改；**

**（4）将相同的变化封装在一个接口或抽象类中；**

**（5） 将不同的变化封装到不同的接口或抽象类中。**

**4、总结**

**（1）单一职责原则要求实现类要职责单一；**

**（2）里氏替换原则要求不要去破坏继承系统；**

**（3）依赖倒置原则要求面向接口编程；**

**（4） 接口隔离原则要求在设计接口的时候要精简单一；**

**（5） 迪米特法则要求要降低耦合；**

**（6）开闭原则是总纲，要求对扩展开发，对修改关闭。**

4人点赞

[设计模式](https://www.jianshu.com/nb/3970364)

