# 实验二：工厂模式

## 实验内容

简单工厂方法模式：利用简单工厂方法模式创建pad, phone, watch的对象，并使用这些对象娱乐。

工厂模式：利用工厂模式创建pad, phone, watch的对象，并使用这些对象娱乐。

抽象工厂模式：利用抽象工厂模式创建华为、小米、苹果的pad, phone, watch的对象，并使用这些对象娱乐。

参考代码下载：<https://gitee.com/liangsan/sa-factory.git> ，抽象工厂模式的代码部分还没有完善，请注意更新。

类名、包名、项目名等请参考，可以适当调整，但不要随意乱命名。

格式等参照默认的规范，细节这里就不在仔细描述了。

## 简单工厂方法模式

### UML设计

### 核心代码

FactoryClient.java核心代码

public class FactoryClient {

private static void testSimpleFactory() {

// 简单工厂模式

ConsumerElectronics pad = cn.edu.scau.sec.tangxuexi.simpleFactoryMethod.factory.ConsumerElectronicsFactory.getConsumerElectronics("pad");

pad.recreation();

ConsumerElectronics phone = cn.edu.scau.sec.tangxuexi.simpleFactoryMethod.factory.ConsumerElectronicsFactory.getConsumerElectronics("phone");

phone.recreation();

ConsumerElectronics watch = cn.edu.scau.sec.tangxuexi.simpleFactoryMethod.factory.ConsumerElectronicsFactory.getConsumerElectronics("watch");

watch.recreation();

}}

ConsumerElectronicsFactory.java核心代码

public static ConsumerElectronics getConsumerElectronics(String consumerElectronicsType) {

ConsumerElectronics consumerElectronics = null;

switch(consumerElectronicsType) {

case "pad": consumerElectronics = new Pad();break;

case "phone": consumerElectronics = new Phone();break;

case "watch": consumerElectronics = new Watch();break;

}

return consumerElectronics;

}

ConsumerElectronics.java核心代码

void recreation();

Pad.java核心代码

public void recreation() {

System.out.println("使用平板娱乐");

}

Phone.java核心代码

public void recreation() {

System.out.println("使用手机娱乐");

}

Watch.java核心代码

public void recreation() {

System.out.println("使用手表娱乐");

}

### 实现效果

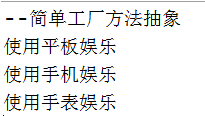


图1 简单工厂方法实现效果图

## 工厂模式

### UML设计

### 核心代码

FactoryClient.java核心代码

private static void testFactoryMethod() {

// 工厂模式

ConsumerElectronicsFactory padFactory = new PadFactory();

ConsumerElectronics pad = padFactory.getConsumerElectronics();

pad.recreation();

ConsumerElectronicsFactory phoneFactory = new PhoneFactory();

ConsumerElectronics phone = phoneFactory.getConsumerElectronics();

phone.recreation();

ConsumerElectronicsFactory watchFactory = new WatchFactory();

ConsumerElectronics watch = watchFactory.getConsumerElectronics();

watch.recreation();

}

ConsumerElectronicsFactory.java核心代码

public abstract class ConsumerElectronicsFactory {

public abstract ConsumerElectronics getConsumerElectronics();

}

PadFactory.java核心代码

public ConsumerElectronics getConsumerElectronics() {

return new Pad();

};

PhoneFactory.java核心代码

public ConsumerElectronics getConsumerElectronics() {

return new Phone();

};

WatchFactory.java核心代码

public ConsumerElectronics getConsumerElectronics() {

return new Watch();

};

### 实现效果

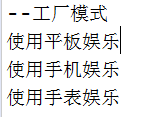


图2 工厂模式实现效果图

## 抽象工厂模式

### UML设计

### 核心代码

### 实现效果

## 实验体会