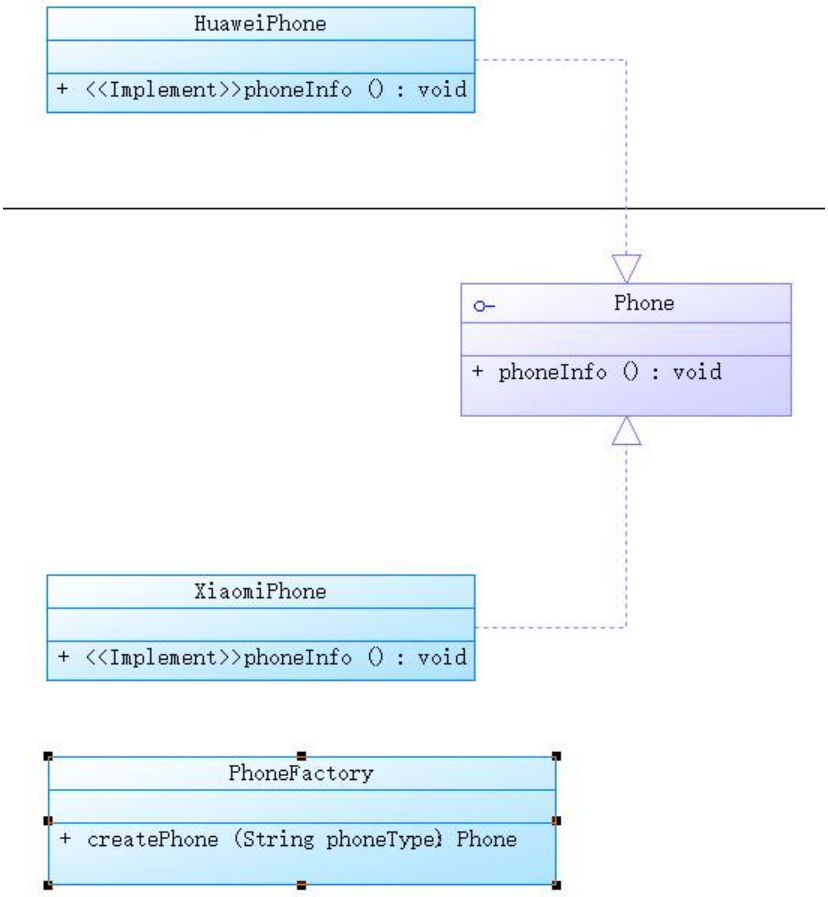
工厂模式

作者：wenyl
原文链接：https://ld246.com/article/1636534071825
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

1.工厂模式

工厂模式根据传递的参数来决定生产的产品，增加产品需要改动工厂的的代码，在明确不同条件生产同实例的情况下可以使用这个模式（一般这个模式可以应付大多数场景了）



原文链接：工厂模式

代码

手机接口：

```
public interface Phone {
    void phoneInfo();
}
```

华为手机

```
public class Huawei implements Phone {

    @Override
    public void phoneInfo() {
        System.out.println("我是华为手机");
    }
}
```

小米手机

```
public class Xiaomi implements Phone {

    @Override
    public void phoneInfo() {
        System.out.println("我是小米手机");
    }
}
```

生产工厂，在工厂中，根据不同的类型，生成不同的手机

```
public class PhoneFactory {
    public Phone createPhone(String type){
        Phone product =null;
        switch (type){
            case "huawei":
                product = new Huawei();
                break;
            case "xiaomi":
                product = new Xiaomi();
                break;
        }
        return product;
    }
}
```

测试工厂方法

```
public class CreatePhone {
    public static void main(String[] args){
        PhoneFactory factory = new PhoneFactory();
        Phone huawei = factory.createPhone("huawei");
        huawei.phoneInfo();
        Phone xiaomi = factory.createPhone("xiaomi");
        xiaomi.phoneInfo();
    }
```

```
}
```

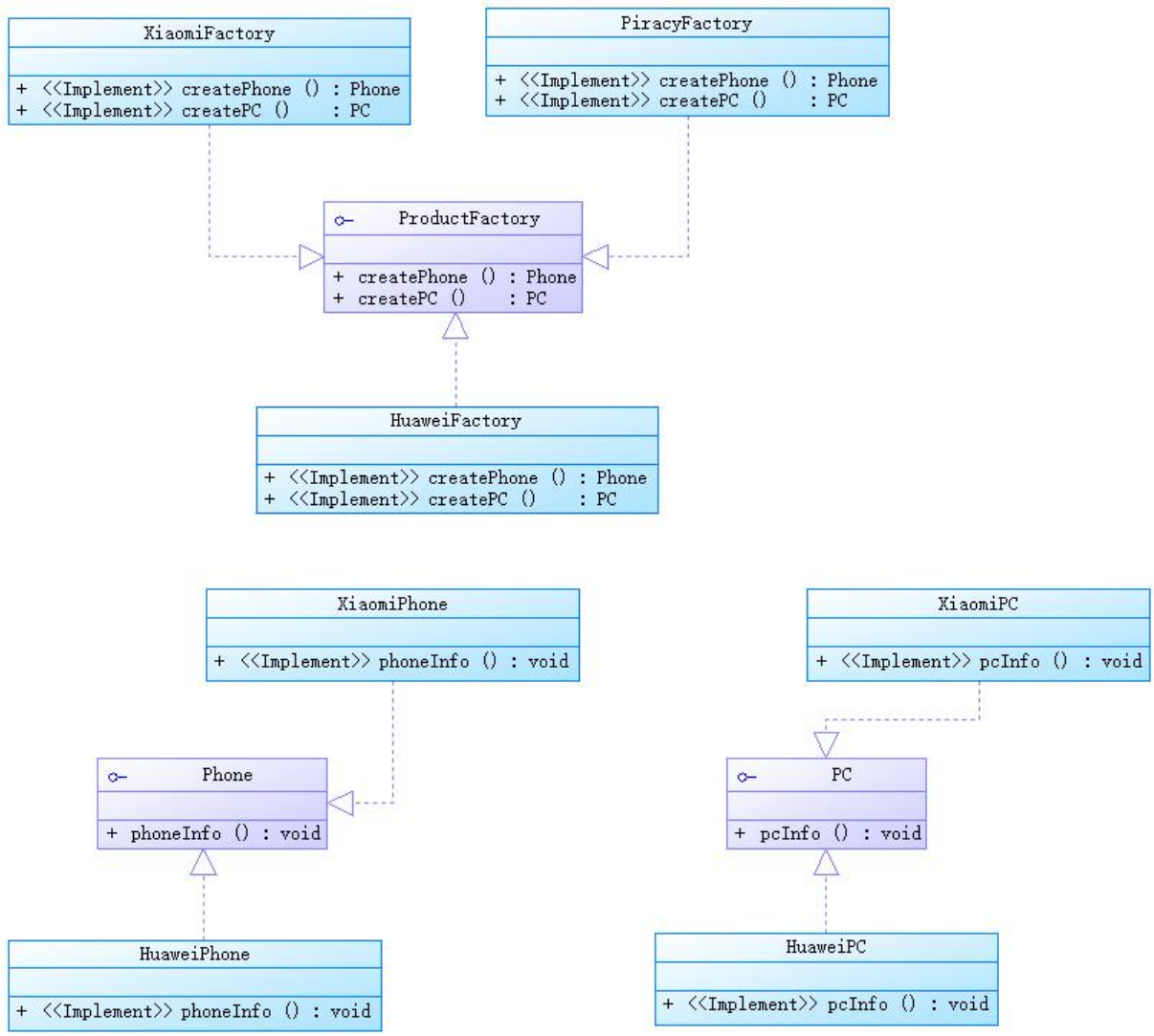
测试结果

```
"D:\Program Files\Java\jdk1.8.0_221\bin\java.exe" ...
Connected to the target VM, address: '127.0.0.1:61171', transport: 'socket'
我是华为手机
我是小米手机
Disconnected from the target VM, address: '127.0.0.1:61171', transport: 'socket'

Process finished with exit code 0
```

2.抽象工厂模式

抽象工厂模式解决了工厂模式增加产品需要修改工厂代码的问题，其实就是对工厂也进行抽象，得到个生产工厂的工厂，在具体的工厂中，可以增加产品，每个工厂也是一个产品簇，这样符合开闭原则功能扩展性强，但是产品簇的扩展比原来难



原文链接：工厂模式

代码信息

手机接口

```
public interface Phone {
    void phoneInfo();
}
```

华为手机

```
public class HuaweiPC implements PC {
    @Override
    public void pcInfo() {
        System.out.println("我是华为电脑");
    }
}
```

小米手机

```
public class XiaomiPhone implements Phone {
    @Override
    public void phoneInfo() {
        System.out.println("我是小米手机！");
    }
}
```

电脑信息

电脑接口

```
public interface PC {
    void pcInfo();
}
```

华为电脑

```
public class HuaweiPC implements PC {
    @Override
    public void pcInfo() {
        System.out.println("我是华为电脑");
    }
}
```

小米电脑

```
public class XiaomiPC implements PC {
    @Override
    public void pcInfo() {
        System.out.println("我是小米电脑");
    }
}
```

工厂信息

抽象工厂

```
public interface ProductFactory {
    Phone createPhone();
    PC createPC();
}
```

华为工厂生产华为手机和华为电脑

```
public class HuaweiFactory implements ProductFactory {
    @Override
    public Phone createPhone() {
        return new HuaweiPhone();
    }

    @Override
    public PC createPC() {
        return new HuaweiPC();
    }
}
```

小米工厂生产小米手机和小米电脑

```
public class XiaomiFactory implements ProductFactory {
    @Override
    public Phone createPhone() {
        return new XiaomiPhone();
    }

    @Override
    public PC createPC() {
        return new XiaomiPC();
    }
}
```

盗版工厂生产华为手机和小米电脑

```
public class PiracyFactory implements ProductFactory {
    @Override
    public Phone createPhone() {
        System.out.println("盗版商家--");
        return new HuaweiPhone();
    }

    @Override
    public PC createPC() {
        System.out.println("盗版商家--");
        return new XiaomiPC();
    }
}
```

测试生产

```
public class CreateProduct {
    public static void main(String[] args){
        ProductFactory xiaomiFactory = new XiaomiFactory();
```

```
        ProductFactory huaweiFactory = new HuaweiFactory();
        ProductFactory piracyFactory = new PiracyFactory();
        xiaomiFactory.createPhone().phoneInfo();
        xiaomiFactory.createPC().pcInfo();
        huaweiFactory.createPhone().phoneInfo();
        huaweiFactory.createPC().pcInfo();
        piracyFactory.createPhone().phoneInfo();
        piracyFactory.createPC().pcInfo();
    }
}
```

结果

```
"D:\Program Files\Java\jdk1.8.0_221\bin\java.exe" ...
Connected to the target VM, address: '127.0.0.1:61522', transport: 'socket'
我是小米手机!
我是小米电脑
我是华为手机!
我是华为电脑
盗版商家--
我是华为手机!
盗版商家--
我是小米电脑
Disconnected from the target VM, address: '127.0.0.1:61522', transport: 'socket'

Process finished with exit code 0
```