设计模式学习总结

目录

[工厂方法模式Factory Method 2](#_Toc532485105)

[抽象方法模式Abstract factory pattern 4](#_Toc532485106)

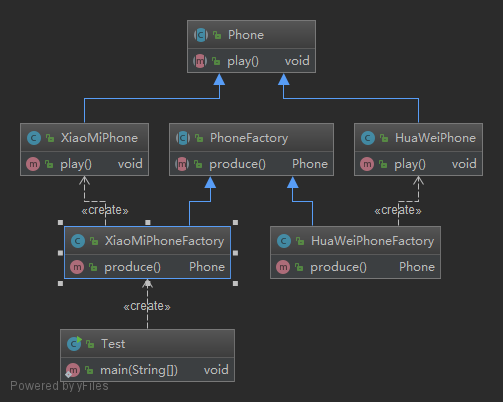
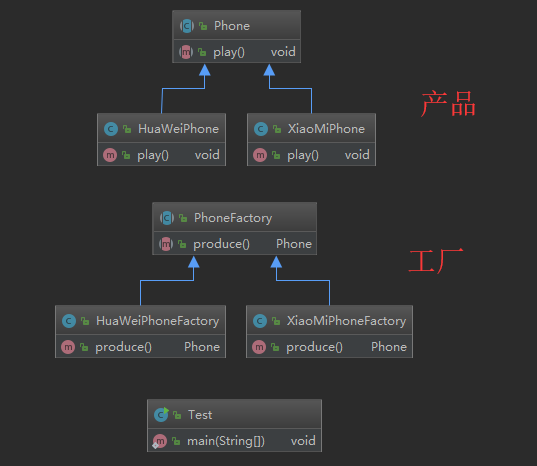
[建造者模式Builder Pattern 6](#_Toc532485107)

[单例模式Singleton 9](#_Toc532485108)

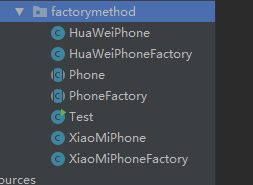
## 工厂方法模式Factory Method

主要应用场景是同一个产品等级，如小米手机，华为手机属于同一产品等级，但是属于不同的工厂，此时的设计方案应该是建立一个工厂和一个产品的抽象类或者接口，再建立相应的小米工厂子类和华为工厂子类，分别生产相应的产品。后续如果再新增同一产品等级的产品，如新增苹果手机，则只需要新建一个苹果工厂类，和一个苹果手机的产品类即可。

UML图：



文件结构：

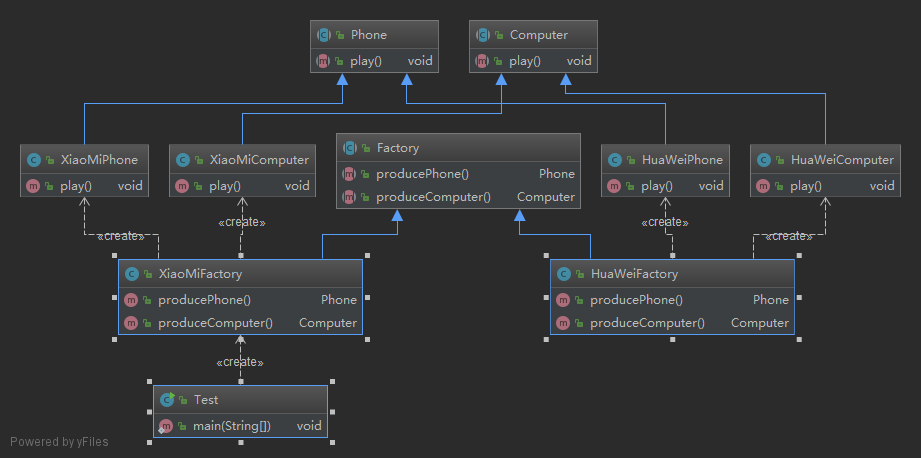
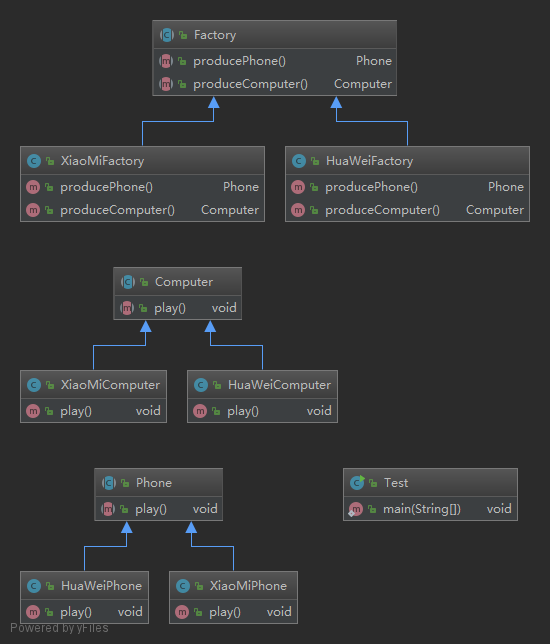


## 抽象方法模式Abstract factory pattern

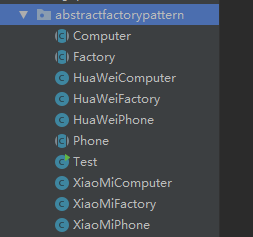
主要应用场景是同一个产品族，比如小米电视，小米手机，小米电脑，是同一产品族。

如果按照工厂方法模式来做，需要新建出来小米手机工厂，小米电脑工厂，如果业务很复杂，可能会出现类爆炸的情况。

抽象模式使用场景是产品等级不经常变动的业务场景，（比如小米工厂确定只生产手机和电脑两种产品，不会经常性的添加产品等级）而只需要其余产品族（如添加一个联想工厂，微软工厂等等），此类场景只需要添加相应工厂实现类和产品实现类即可。



文件结构：

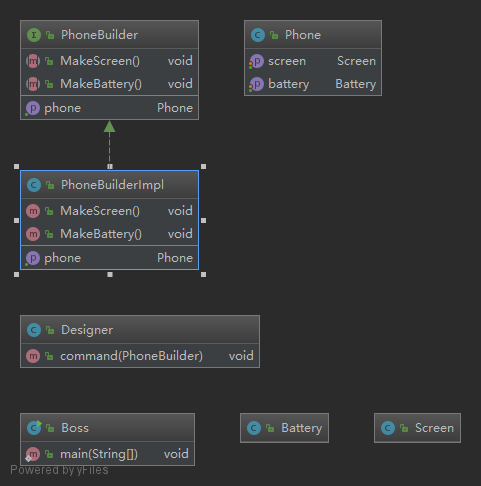


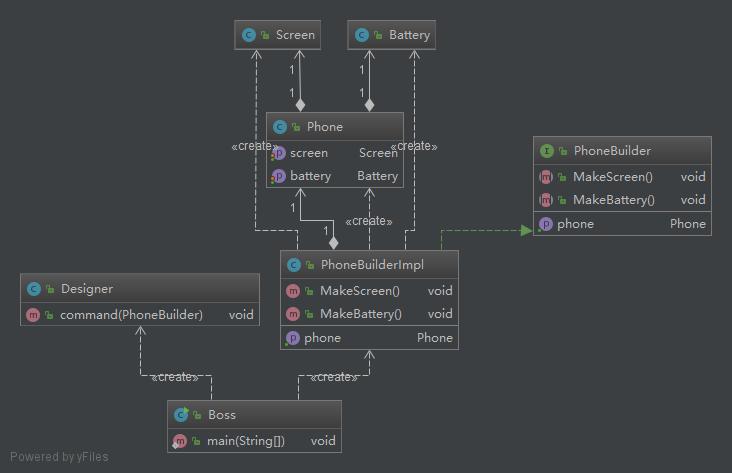
## 建造者模式Builder Pattern

建造者模式适合于一个具有较多属性的产品的创建过程，根据产品创建过程中属性的构造是否具有一致的先后顺序，分为两种形式。

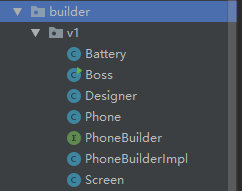
1. 通过Client、Director、Builder、Product形成的建造者模式。Builder负责构建Product类对象，Director负责指导Builder按照指定的顺序去构造Product，最后通过Builder返回建造后的结果。

场景：假设手机有屏幕Screen和电池Battery两个零件。Boss老板需要一个手机，就指派给设计师Designer一个任务说我需要一个手机，由设计师指挥命令PhoneBuilder工人去造手机，先造屏幕，再造电池，然后造出来一个手机给Designer（command方法实现），再由Designer把

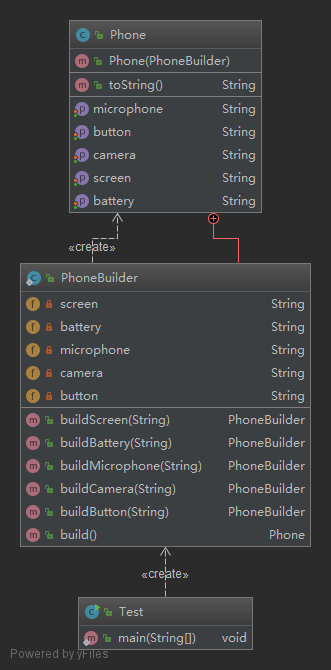


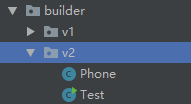


文件结构



1. 通过静态内部类方式实现的属性无序化构造，可以采用链式赋值，用到什么属性给什么属性赋值。





## 单例模式Singleton

1．懒汉式单例模式

