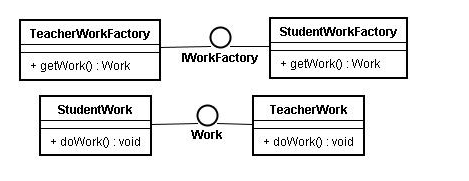
**一、工厂方法（FactoryMethod）**

* **概述**

定义一个用于创建对象的接口，让子类决定实例化哪一个类。FactoryMethod使一个类的实例化延迟到其子类。

* **适用性**
* 当一个类不知道它所必须创建的对象的类的时候；
* 当一个类希望由它的子类来指定它所创建的对象的时候；
* 当类将创建对象的职责委托给多个帮助子类中的某一个，并且你希望将哪一个帮助子类是代理者这一信息局部化的时候；
* **举例**

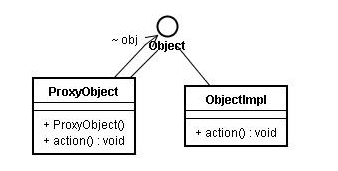


* **总结**

FactoryMethod模式是设计模式中应用最为广泛的模式，在面向对象的编程中，对象的创建工作非常简单，对象的创建时机却很重要。FactoryMethod解决的就是这个问题，它通过面向对象的收发，将所要创建的具体对象创建工作延迟到了子类，从而提供了一种扩展策略，较好的解决了这种紧耦合的关系。

**二、代理模式（Proxy）**

* 概述
* 为其他对象提供一种代理以控制对这个对象的访问。



* 举例

public class TestProxy {  
  
 public static void main(String[] args) {  
 Object obj = new ProxyObject();  
 obj.action();  
 }  
}  
  
interface Object {  
 void action();  
}  
  
class ProxyObject implements Object {  
 Object obj;  
  
 public ProxyObject() {  
 System.***out***.println("代理类创建成功");  
 obj = new ObjectImpl();  
 }  
  
 @Override  
 public void action() {  
 System.***out***.println("代理类开始执行");  
 obj.action();  
 System.***out***.println("代理类执行结束");  
 }  
}  
  
class ObjectImpl implements Object {  
  
 @Override  
 public void action() {  
 System.***out***.println("==被代理类开始执行==");  
 System.***out***.println("==具体操作==");  
 System.***out***.println("==被代理类执行完毕==");  
 }  
}