# 简单工厂模式

简单工厂模式，就是说到底要实例化谁，将来会不会增加实例化的对象（即有一些不可预知的变化的地方），应该考虑用一个单独的类来做这个创造实例的过程，这就是工厂。

具体代码：

public class OperationFactory

{

public static Operation createOperation(string operation)

{

Operation oper = null;

switch(operation){

case “+”：

oper = new OperationAdd();

break;

case "=":

oper = new OperationSub();

break;

case "\*":

oper = new OperationMul();

break;

case "/":

oper = new OperationDiv();

break;

}

return oper;

}

}

//客户端代码：

Operation oper;

oper = OperationFactory.createOperation("+");

oper.number1 = 1;

oper.number2 = 2;

double result = oper.GetResult()

简单工厂模式的最大优点在于工厂类中包含了必要的逻辑判断，根据客户端的选择条件动态实例化相关的类，对于客户端来说，去除了与具体产品的依赖。

但是，如果我们需要求解M的N次方运算，就应该在case语句中增加一个判断，修改原来的class类，这显然是不合理的，违背了开放-封闭原则，于是引入了下面的工厂方法模式。

# 工厂方法模式

工厂方法模式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，工厂方法使一个类的实例化延迟到其子类。