建造者模式主要分为以下几种角色：

1. 抽象建造者（Ibuilder）
2. 具体建造者（实现抽象建造者，builder）
3. 产品类（具体建造者里面创建具体产品类，product）
4. 指挥者（Director）

代码示例：

SetMealProduct对应产品类

ISetMealBuilder对应抽象建造者类

SetMealABuilder,SetMealBBuilder对应具体建造者类

KFCDirector对应指挥者类

ISetMealBuilder抽象类内部持有SetMealProduct对象，并提供创建产品组件的抽象方法与获取产品对象的方法

SetMealABuilder,SetMealBBuilder继承ISetMealBuilder抽象类并实现ISetMealBuilder里面创建产品组件的抽象方法

创建KFCDirector时需要传入一个具体的建造者对象（SetMealABuilder或SetMealBBuilder），并提供一个创建并获取产品的方法，这个方法内部调用SetMealABuilder或SetMealBBuilder

的创建组件的抽象方法，最后在调用获取产品的方法将创建好的产品返回给用户

个人感觉这个模式和工厂模式有些类似，个人理解工厂模式有些类似简化版的建造者模式，工厂模式是创建了某个单一的组件，而建造者模式是把一个产品里面的单一组件组成一个成品提供给客户来使用

**public class** SetMealProduct {  
  
 */\*\*  
 \* 食物  
 \*/* **private** String **foot**;  
  
 */\*\*  
 \* 饮料  
 \*/* **private** String **drink**;  
  
 **public** String getFoot() {  
 **return foot**;  
 }  
  
 **public void** setFoot(String foot) {  
 **this**.**foot** = foot;  
 }  
  
 **public** String getDrink() {  
 **return drink**;  
 }  
  
 **public void** setDrink(String drink) {  
 **this**.**drink** = drink;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "SetMealProduct{"** +  
 **"foot='"** + **foot** + **'\''** +  
 **", drink='"** + **drink** + **'\''** +  
 **'}'**;  
 }  
}

*/\*\*  
 \* 套餐创建类  
 \*/***public abstract class** ISetMealBuilder {  
  
 SetMealProduct **setMealProduct** = **new** SetMealProduct();  
  
 **abstract void** buildFoot();  
  
 **abstract void** buildDrink();  
  
 **public** SetMealProduct getSetMealProduct(){  
 **return setMealProduct**;  
 }  
}

*/\*\*  
 \* 套餐A创建类  
 \*/***public class** SetMealABuilder **extends** ISetMealBuilder{  
 @Override  
 **void** buildFoot() {  
 **super**.**setMealProduct**.setFoot(**"汉堡A"**);  
 }  
  
 @Override  
 **void** buildDrink() {  
 **super**.**setMealProduct**.setDrink(**"饮料A"**);  
 }  
}

*/\*\*  
 \* 套餐B创建类  
 \*/***public class** SetMealBBuilder **extends** ISetMealBuilder{  
 @Override  
 **void** buildFoot() {  
 **super**.**setMealProduct**.setFoot(**"汉堡B"**);  
 }  
  
 @Override  
 **void** buildDrink() {  
 **super**.**setMealProduct**.setDrink(**"饮料B"**);  
 }  
}

*/\*\*  
 \* 创建指挥者  
 \*/***public class** KFCDirector {  
  
 **private** ISetMealBuilder **setMealBuilder**;  
  
 **public** KFCDirector(ISetMealBuilder setMealBuilder){  
 **this**.**setMealBuilder** = setMealBuilder;  
 }  
  
 **public** SetMealProduct buildSetMealProduct(){  
 **setMealBuilder**.buildFoot();  
 **setMealBuilder**.buildDrink();  
 **return setMealBuilder**.getSetMealProduct();  
 }  
}

*/\*\*  
 \* 测试类  
 \*/***public class** Test {  
  
 **public static void** main(String[] arg0){  
 ISetMealBuilder setMealABuilder = **new** SetMealABuilder();  
 KFCDirector kfcADirector = **new** KFCDirector(setMealABuilder);  
 System.***out***.println(kfcADirector.buildSetMealProduct());  
  
 ISetMealBuilder setMealBBuilder = **new** SetMealBBuilder();  
 KFCDirector kfcBDirector = **new** KFCDirector(setMealBBuilder);  
 System.***out***.println(kfcBDirector.buildSetMealProduct());  
 }  
}