## 简介

简单的对象一步步构建为复杂的对象。

## 问题

将一个复杂的构件语气表示分离，使得同样的构建过程可以构建不同的表示。

或者说，一个对象需要不同的构件组合而成，不同数量、类型、顺序的构件组合能组合成不同的对象实例。

## 角色

1. 建造者接口（builder）:建造产品的构件的接口，具体的实现方法由实现类完成。
2. 建造者接口实现类（ConcreteBuilder）：建造产品的构件，
3. 指挥者（Director）接口：根据建造者实现类构件不同的产品。
4. 产品（Product）：构建的最终产品。

## 优缺点

优点：

1. 封装性：客户端不必知道产品内部的组成细节。
2. 易扩展：builder和产品是独立的，对系统的扩展比较有利
3. 便于控制细节风险：由于具体的建造者是独立的，因此可以对建造过程逐步细化，而不对其他模块产生任何影响。

## 使用场景

1. 相同的方法，不同的执行顺序，产生不同的事件结果。
2. 多个部件或零件，都可以装配到一个对象中，但是产生的运行结果不相同
3. 产品类非常复杂，或者产品类中的调用顺序不同产生了不同的效能
4. 在对象创建过程中会使用系统中的一些其他对象，这些对象在产品对象的创建过程不易得到。该种场景，只能是一个补偿方法，因为一个对象不容易获得，而在设计阶段竟然没有发觉，而要通过创建者模式柔化创建过程，本身已经违反设计最初目标。

## 注意事项

建造者模式关注的是零件工艺和装配工艺，这是它与工厂模式最大的不同。