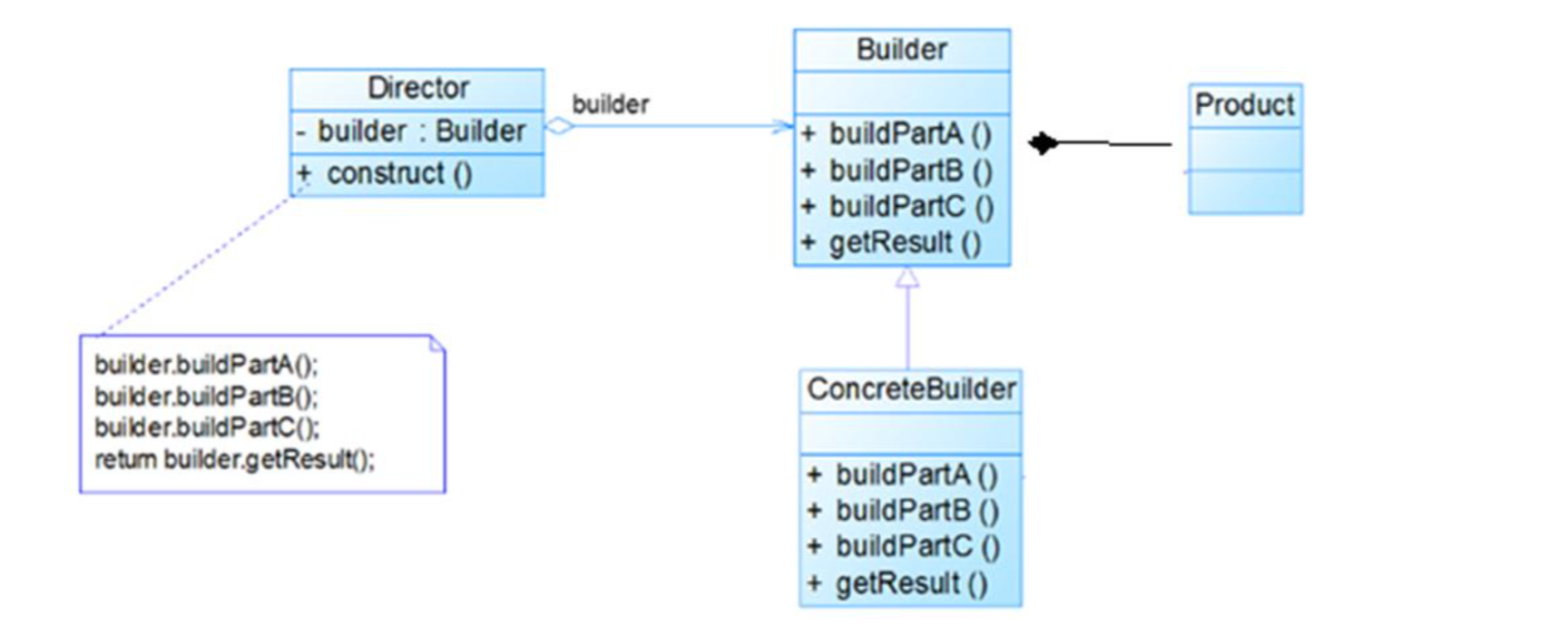
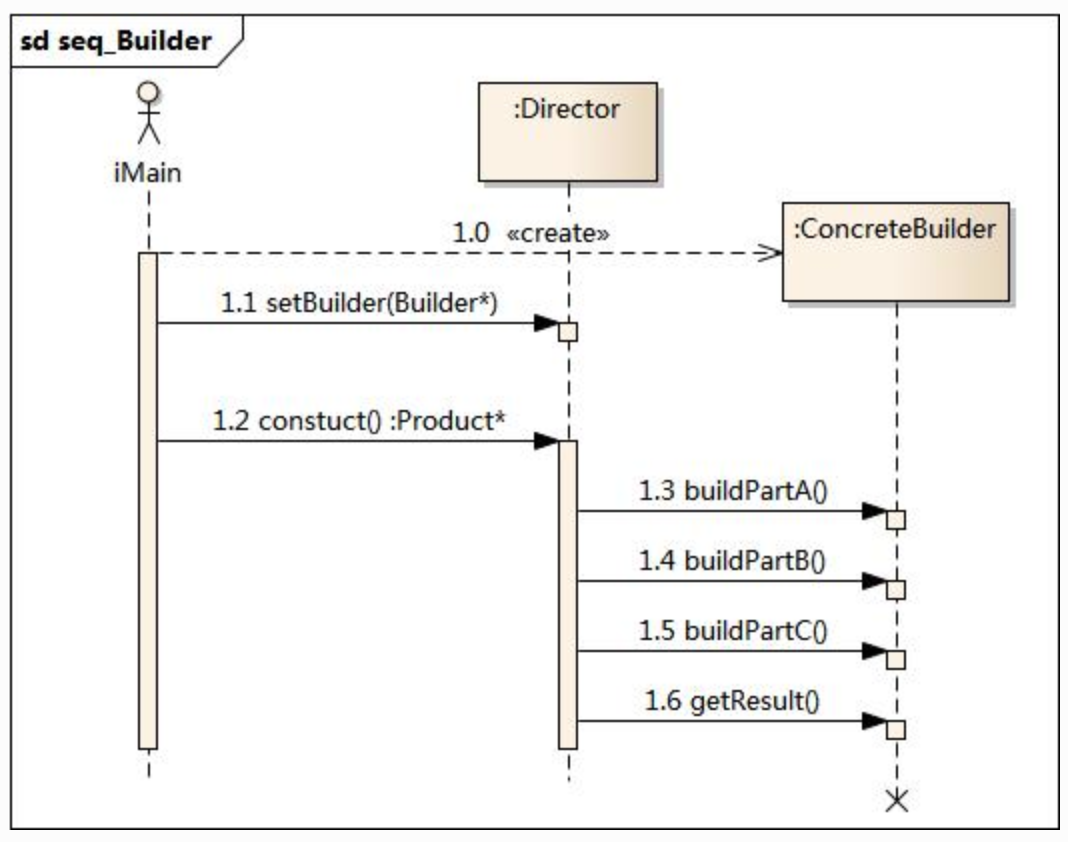
1. 建造者模式(Builder Pattern)：将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。
   1. 又叫生成器模式，是一种对象构建模式。它可以将复杂对象的建造过程抽象出来(抽象类别)，使这个抽象过程的不同实现方法可以构造出不同表现(属性)的对象
   2. 建造者模式是一步一步创建一个复杂的对象，它允许用户只通过指定复杂对象的类型和内容就可以构建它们， 用户不需要知道内部的具体构建细节
2. 类图

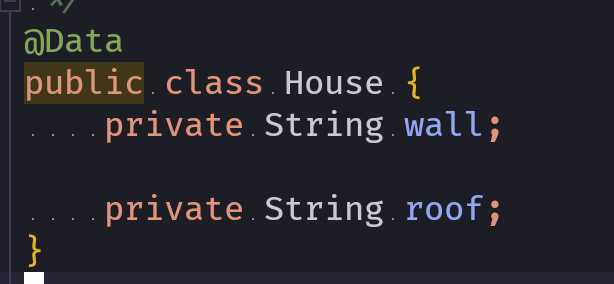


* 1. Product：一个具体的产品对象
  2. Builder：创建一个 Product 对象的各个部件指定的 接口**/**抽象类
  3. ConcreteBuilder: 实现接口，构建和装配各个部件
  4. Director: 构建一个使用 Builder 接口的对象。它主要是用于创建一个复杂的对象。它主要有两个作用，一是: 隔离了客户与对象的生产过程，二是:负责控制产品对象的生产过程

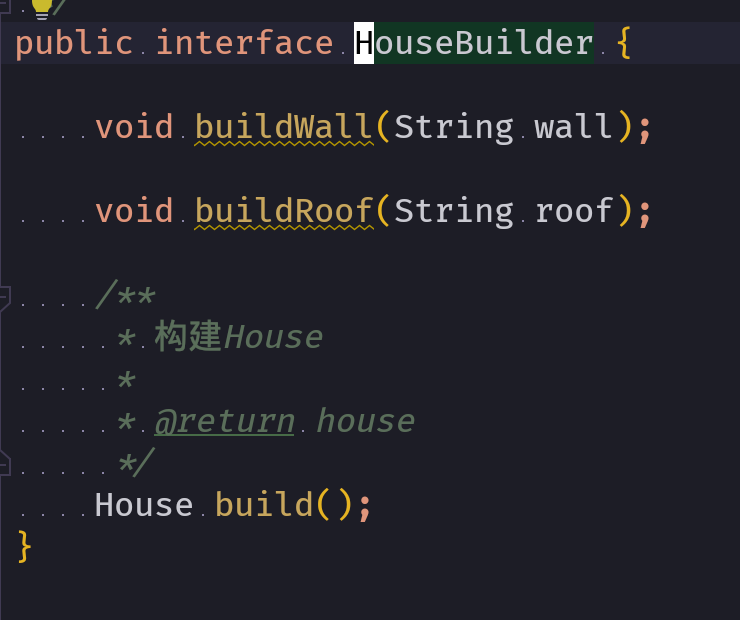
1. 时序图



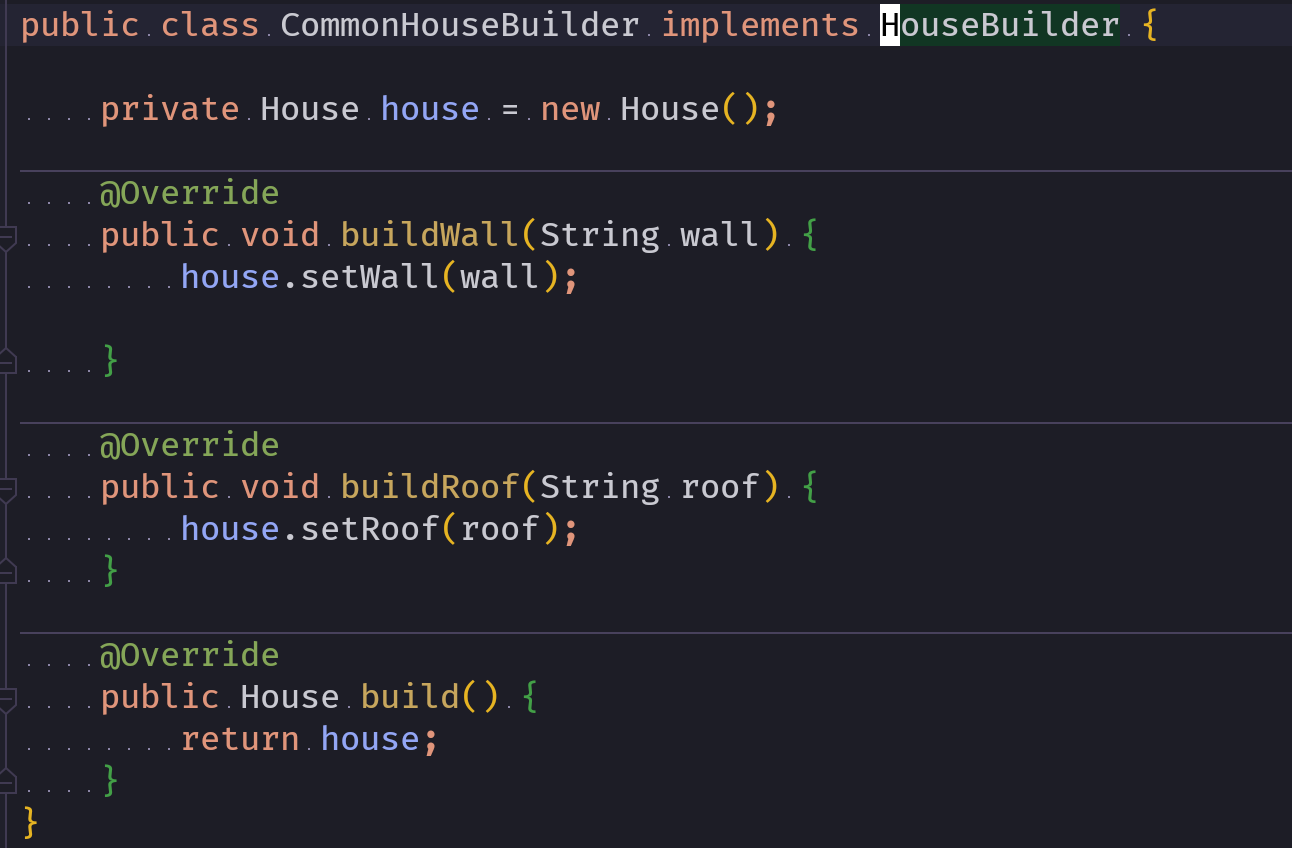
1. 模式分析
   1. 抽象建造者类中定义了产品的创建方法和返回方法
   2. 建造者模式的结构中还引入了一个指挥者类Director，该类的作用主要有两个：一方面它隔离了客户与生产过程；另一方面它负责控制产品的生成过程。指挥者针对抽象建造者编程，客户端只需要知道具体建造者的类型，即可通过指挥者类调用建造者的相关方法，返回一个完整的产品对象
   3. 在客户端代码中，无须关心产品对象的具体组装过程，只需确定具体建造者的类型即可，建造者模式将复杂对象的构建与对象的表现分离开来，这样使得同样的构建过程可以创建出不同的表现
2. 建造者模式优点
   1. 在建造者模式中， 客户端不必知道产品内部组成的细节**，**将产品本身与产品的创建过程解耦**，**使得相同的创建过程可以创建不同的产品对象**。**
   2. 每一个具体建造者都相对独立，而与其他的具体建造者无关，因此可以很方便地替换具体建造者或增加新的具体建造者， 用户使用不同的具体建造者即可得到不同的产品对象 。
   3. 可以更加精细地控制产品的创建过程 。将复杂产品的创建步骤分解在不同的方法中，使得创建过程更加清晰，也更方便使用程序来控制创建过程。
   4. 增加新的具体建造者无须修改原有类库的代码**，**指挥者类针对抽象建造者类编程**，**系统扩展方便**，**符合**“**开闭原则**”。**
3. 建造者模式缺点
   1. 建造者模式所创建的产品一般具有较多的共同点，其组成部分相似，如果产品之间的差异性很大，则不适合使用建造者模式，因此其使用范围受到一定的限制。
   2. 如果产品的内部变化复杂，可能会导致需要定义很多具体建造者类来实现这种变化，导致系统变得很庞大
4. 实例
   1. 产品Product



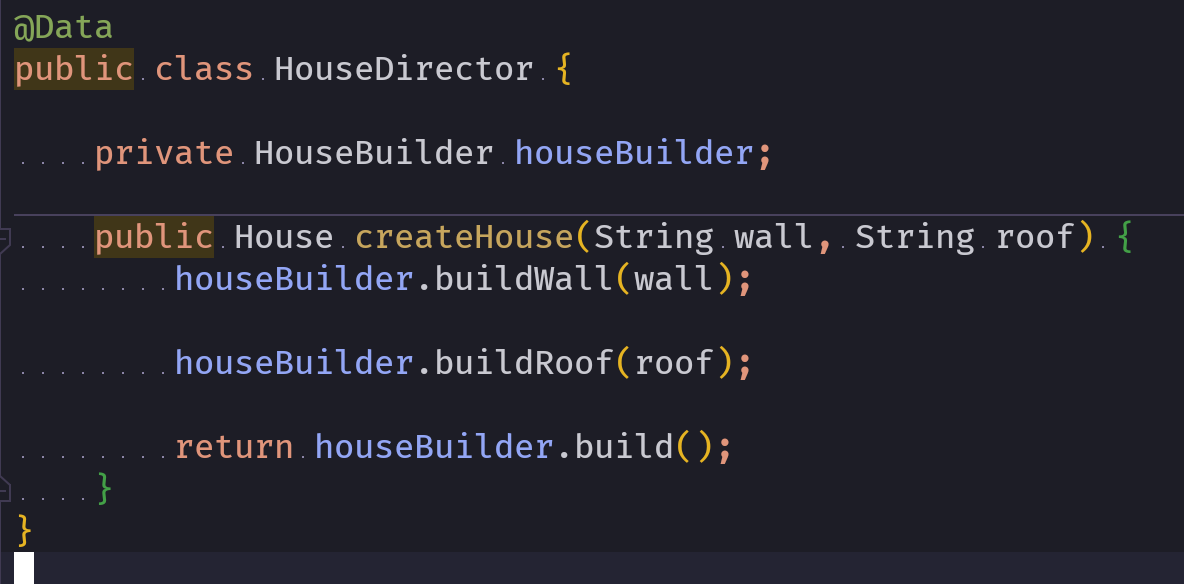
* 1. Builder



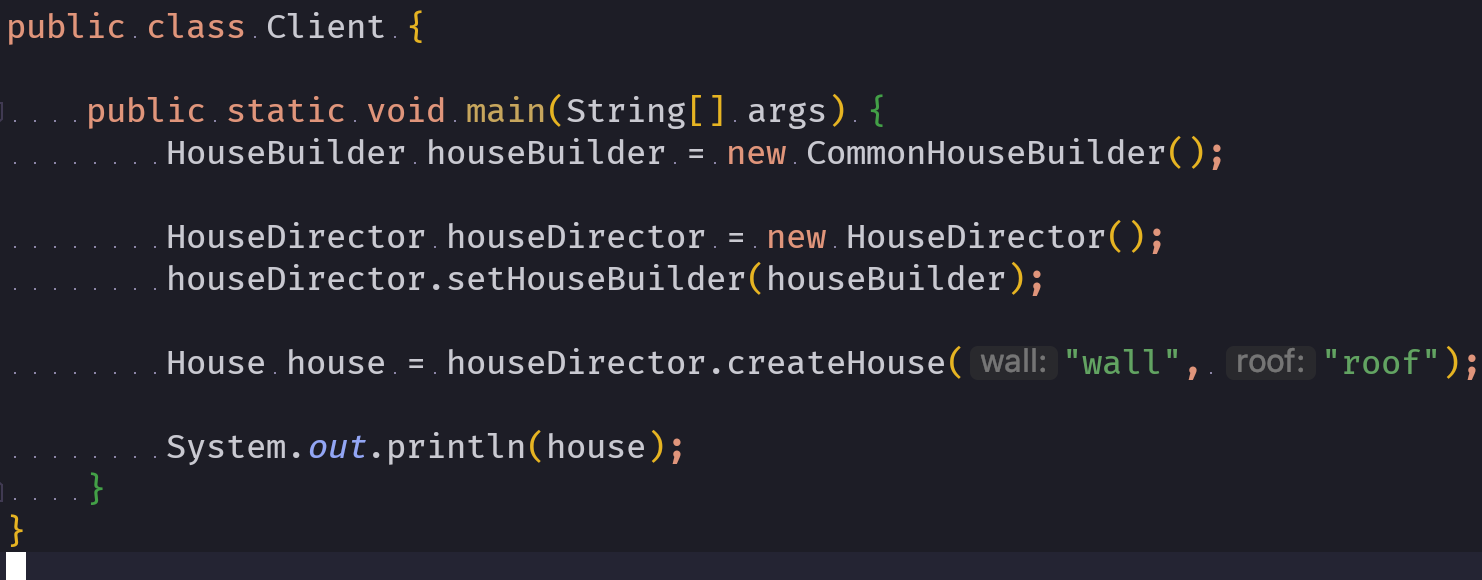
* 1. ConcreteBuilder



* 1. Director



* 1. Client



1. JDK中java.lang.StringBuilder使用了建造者模式