抽象工厂模式 Abstract Factory Pattern

# 实现功能

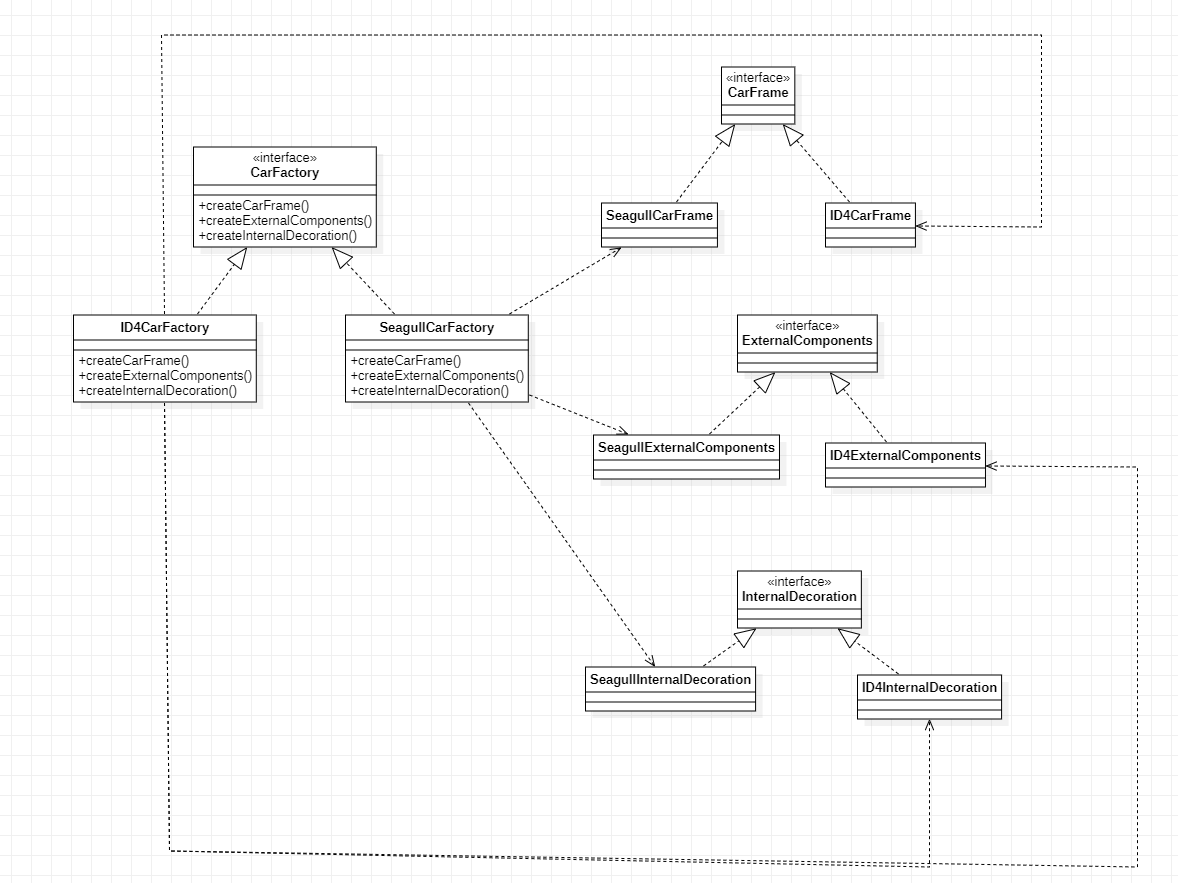
抽象工厂模式提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。当一个产品族中的多个对象被设计成一起工作时，它能保证客户端始终只使用同一个产品族中的对象。

在这个抽象工厂模式的示例中，实现了一个生产不同的配套的汽车构件的工厂。

通过一个抽象工厂CarFactory，它提供生产若干种零件的接口。其中，createCarFrame()可以生产车身框架，createExternalComponents()可以生产车的外部构件；createInternalDecoration()可以生产车的内部构件。而两个具体工厂ID4CarFactory和SeagullCarFactory分别能够生产不同的对应车型所需的不同构件。这些具体的构件类(SeagullCarFrame、ID4CarFrame等)均继承自抽象类，即CarFrame、ExternalComponents和InternalDecoration类。

一个工厂中创建的每个对象都是同一个系列的（“Seagull”或者“ID4”）。客户端程序只需要知道得到的对象是车身、外部构件或是内部构件，而不需要知道具体的系列。

# 类图



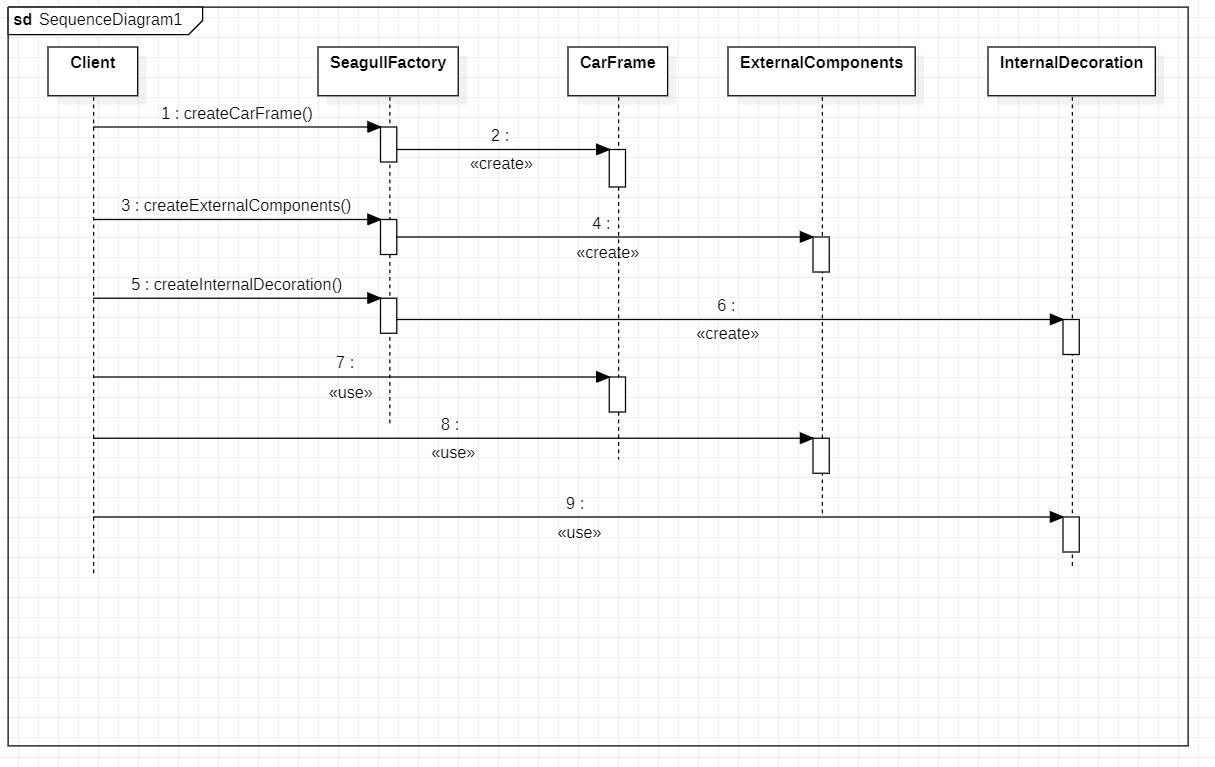
- 抽象工厂类CarFactory

- 具体工厂类ID4CarFactory、SeagullCarFactory

- 抽象产品类 CarFrame、ExternalComponents、InternalComponents

- 具体产品类 SeagullCarFrame、SeagullExternalComponents、SeagullInternalDecoration、ID4CarFrame、ID4ExternalComponents、ID4InternalDecoration

# 时序图



# 代价分析

在添加新的产品对象（如电池）时，难以扩展抽象工厂来生产新种类的产品，这是因为在抽象工厂角色中规定了所有可能被创建的产品集合（车身、外部构件、内部构件），要支持新种类的产品就意味着要对该接口进行扩展，而这将涉及到对抽象工厂角色及其所有子类的修改，修改会很麻烦。