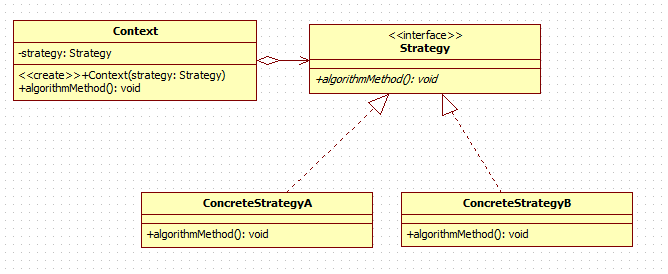
## 策略模式

1. 定义与使用场景

定义：策略模式定义了一系列的算法，并将每一个算法封装起来，而且使它们还可以相互替换。策略模式让算法独立于使用它的客户而独立变化。

使用场景：针对同一个类型问题的多种处理方式，仅仅是具体行为有差别时。

1. UML类图



Context：用来操作策略的上下文环境

Stragety：策略的抽象

ConcreteStrategyA、ConcreteStrategyB：具体的策略实现

1. 简单实现

通常如果一个问题有多个解决方案时，最简单的方式就是使用if-else或者switch-case方式根据不同的情景选择不同的解决方案，但这种简单的方案问题太多。如果解决方案中包含大量的逻辑需要封装，或者处理方式变动较大时，就显得混乱、复杂，当需要增加一种方案时就需要修改类中的代码。

应对这种情况使用策略模式就能很好地解决这类问题。它将各种方案分离开来，让程序客户端根据具体的需求来动态地选择不同的策略方案。

1. 源码实现

属性动画，TimeInterpolator

1. 总结

策略模式主要用来分离算法，在相同的行为为抽象下有不同的具体实现策略。这个模式很好地演示了开闭原则，也就是定义抽象，注入不同的实现，从而达到很好地可扩展性。