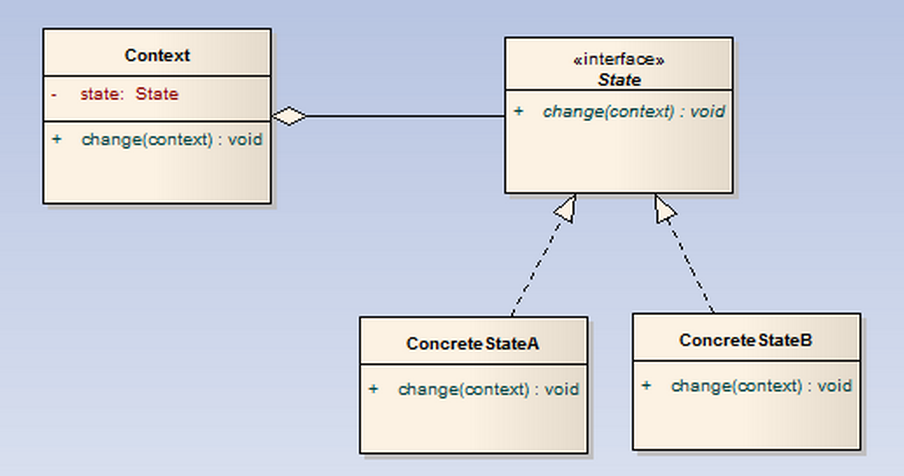
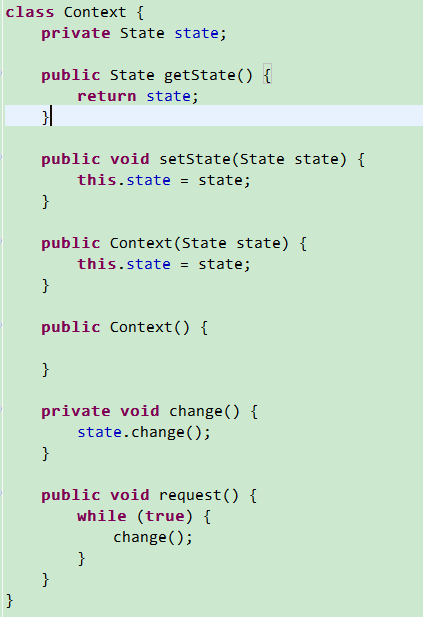
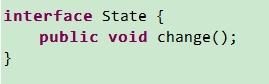
1. 状态模式：允许一个对象在其内部状态改变时改变它的行为。对象看起来似乎修改了它的类。状态模式主要解决的是当控制一个对象状态转换的条件表达式过于复杂时的情况。把状态的判断逻辑转移到表示不同状态的一系列类当中，可以把复杂的判断逻辑简化。当一个对象的行为取决于它的状态，并且它必须在运行时刻根据状态改变它的行为时，就可以考虑使用状态模式了。
2. 核心: 用于解决系统中复杂对象的状态转换以及不同状态下行为的封装问题
3. 类图(同策略模式类图)

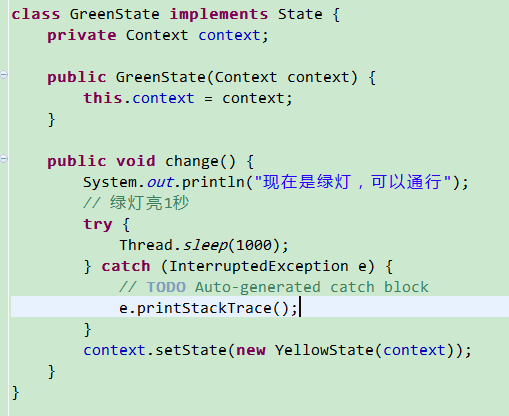


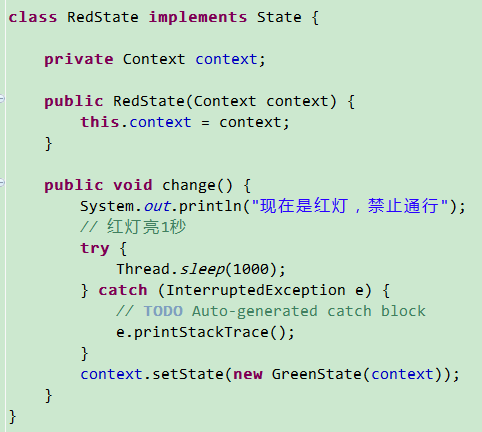
* 1. 抽象状态（State）角色：定义一个接口，用以封装环境对象的一个特定的状态所对应的行为。
  2. 具体状态（ConcreteState）角色：每一个具体状态类都实现了环境的一个状态所对的行为。
  3. 场景（Context）角色：定义客户端所感兴趣的接口，并且保留一个具体状态类的实例。这个具体状态类的实例给出此环境对象现有的状态。

1. 例

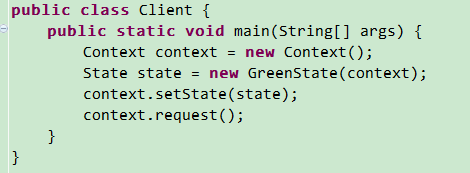












通过上面例子可以看出，状态模式将与特定状态相关的行为局部化，并且将不同状态的行为分割开来；所有状态相关的代码都存在于某个ConcereteState中，所以通过定义新的子类很容易地增加新的状态和转换；状态模式通过把各种状态转移逻辑分不到State的子类之间，来减少相互间的依赖。缺点是：会导致有很多State 的子类。

状态模式和策略模式结构完全一样，很容易混淆，这里列举下状态模式和策略模式的区别：

1、状态模式有明显的状态过渡，从一个状态到另一个状态转换过程，在整个生命周期里有多个状态转换；而策略模式一旦环境角色选择了一个具体的策略类，那么在整个生命周期它始终不会改变；

2、状态模式多数是外在原因在环境角色中放入一个具体的状态类，而策略模式是自主选择一个具体的策略类；

3、状态模式选择一个状态是会明显告诉客户端的，而策略模式则不会告诉客户端选择了什么策略。