J2EE设计模式之State模式计算机等级考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/645/2021_2022_J2EE_E8_AE_BE_E8_AE_A1_c97_645195.htm 设计模式，这个概念现在是满天飞，大家手里面估计都有，Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, in the book "Design Patterns - Element of Re-Useable Object-Oriented Software",这本书。根据其中的说法，有三种，创建型模式、结构型模式和行为型模式。一共包括23个模式，在这里不一一列举。实际应用中，我们不可能在短时间内消化吸收掉，因为模式本身就是约定俗成的东西，依靠大家以往的项目经验总结出来的习惯用法。习惯要时间去培养，去形成，所以大家不要急着去用所有的模式，但一定要有这种意识，在项目中、学习中开始用上几个看看，研究研究模式给你会带来什么价值。但如果，你对Java JDK1.4 源码用到的设计模式感兴趣，可以到官方网站下载！ 在进入正题之前，我申明该文章没有任何商业目的，其中涉及到开源软件的一些源代码，所以涉及到知识产权问题时，如果有人将其用于商业目的，个人不负责该方面的责任。在这里只用于学习、交流的目的。谢谢合作。 但问题出来了，如何将这些模式具体用到Java Project中呢？刚开始确实很头疼。一方面，我们要知道模式的适用场合、引入模式给系统带来的好坏，等等都需要我们去考虑的。另一方面，模式本身的理解消化吸收。再者，还有你的老板逼着你完成任务-：）。等等。其实，冷静分析一下，也不是没有可能。我们在项目中，完成自己的工作是一方面，但另一方面要考虑好自己的职业生涯，要想把Java Coder、Java Developer、Java

Architect、或者什么的……做好,这些都是基本功。 学者，就是喜欢把1 1，这么简单的问题用什么。。。。理论推导出来，作为技术工作者，比如我们，看的是结果，技术的实作性。而大家在学设计模式中的过程中，往往脱离实践，看看设计模式的UML图（说句实在话，大家的UML功底都不会太好吧，把它用于我们的UP估计也少。）、还有模式的什么意图，别名，动机，适用性，结构，参与者，协作，效果，实现，代码实例，。。。。。。。。脑袋都晕掉了。大家是否给忘记了，这种GP,本来 就很抽象，加上这么多条条框框，两个字，“ 郁闷 ”。我们不妨从分析GP代码入手效果很很不错，我学习过程中就是这样的。要不我们现在就看看一个例子。以行为型模式State为例。（模式的实现例子网络上有很多。Java的实现也很多，比如，Together ControlCenter里面就内置了这种基于GP的编程模板，详细内容大家用用就知道了。） 大家知道，State的用意在于，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他的类。我们先看http://www.javacoder.net/patterns.jsp 上提供的GOF SOFTWARE DESIGN PATTERNS CATALOGUE的State模式源码实现。 首先看接口类，State.java public interface State { public void handle(). } 以定义接口以封装与Conext(代码稍候陈述！)的一个特定状态相关的行为。 然后看，接口的实现类。第一个，ConcreteState1.java public class ConcreteState1 implements State { public void handle() { System.out.println("ConcreteState1.handle() executing"). } } 第二个，ConcreteState2.java public class ConcreteState2 implements State { public void handle() {

System.out.println("ConcreteState2.handle() executing"). } } 这两个类实现了State接口。 然后再看，Context.java是如何将上述三个.java文件联系起来。 public class Context { public static final int STATE_ONE = 0. public static final int STATE_TWO = 1. //大家注意，这句话很关键，该模式做手脚的地方！全国计算机等级考试网，加入收藏 private State currentState = new ConcreteState1(). public void request() { currentState.handle(). } public void changeState(int state) { switch (state) { case STATE_ONE: currentState = new ConcreteState1().//关键点 break. case STATE_TWO: currentState = new ConcreteState2().//关键点 break. } } } 这样，写好4个.java文件后，其实您已经实现了State设计模式，很有趣，对吧？就是这样简单。 再看看如何使用该设计模式了。写一个Client.java看看。 public class Client { public static void main(String[] args) { // 构造Context Context ctx = new Context(). // 唤起Context.request() ctx.request(). // 改变ctx的状态？为什么改变了呢？大家想想看。 ctx.changeState(Context.STATE_TWO). // 再次唤起Context.request()，结果大不一样。 ctx.request(). } } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com