

EasyMock的应用的举例说明Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/644/2021_2022_EasyMock_E7_9A_c104_644866.htm 1 能够帮助解耦设计。即以接口设计为中心 2 检查你的代码使用其他对象的情况。通过为mock设置期望的行为，我们可以验证（verify）我们的代码是否正确的使用了我们被mock的接口 3 对我们的代码从里到外进行测试 4 使你的测试运行的更快。通过mock 数据库，通讯等接口，我们可以避免耗时的连接。 5 使得与硬件设备交换、远程系统的开发更容易。 6 可以推迟我们要实现的类。因为我们可以通过mock去模拟接口的实现，所以前期我们可以不用实现接口。 7 促进 基于接口的设计 8 鼓励多使用组合而不是继承 9 可以精炼接口 10 可以测试那些 不太容易发生，不太可能和一些异常情况 下载 easymock 最新版本是 2.2 看到他的 document也很简单 最重要的几句话 To get a Mock Object, we need to create a Mock Object for the interface we would like to simulate, record the expected behavior, and switch the Mock Object to replay state. 意思是说我们用easymock来做 必须做三点 1 创建接口的 mock类 2 记录期望的行为 3 把mock对象 设置为 replay状态 4 verify 是可选的 比如记录方法调用的次数等 好来写个东东试试 先写测试类（虽然书上这么建议，可是我在实际项目中 还是经常是先写 接口，而不是先写测试，感觉不太习惯。）

```
package ex2; import static org.junit.Assert.*; import static org.easymock.EasyMock.*; import org.junit.Before; import org.junit.Test; public class StrategyuseTest { private Strategy mock; private Strategyuse strategyuse; @Before public void setUp() throws
```

```
Exception { mock=createMock(Strategy.class). strategyuse=new
Strategyuse(). strategyuse.setStrategy(mock). } @Test public void
testAddtwoint() { expect(mock.add(1, 2)).andReturn(3).
replay(mock). assertEquals(3, strategyuse.addtwoint(1, 2)).
verify(mock). } @Test public void testMinustwoint(){
expect(mock.add(1, 2)).andReturn(-1). replay(mock).
assertEquals(-1, strategyuse.addtwoint(1, 2)). verify(mock). } }
package ex2. public interface Strategy { int add(int a,int b). int
minus(int a,int b). } package ex2. public class Strategyuse { private
Strategy strategy. public Strategy getStrategy() { return strategy. }
public void setStrategy(Strategy strategy) { this.strategy = strategy. }
public int addtwoint(int a,int b){ return this.strategy.add(a, b). }
public int minustowint(int a,int b) { return this.strategy.minus(a, b).
} } 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
```