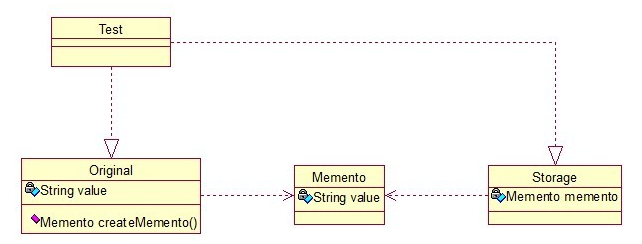
# 备忘录模式（Memento）设计模式

备忘录模式（Memento）主要目的是保存一个对象的某个状态，以便在适当的时候恢复对象，个人觉得叫备份模式更形象些，通俗的讲下：假设有原始类A，A中有各种属性，A可以决定需要备份的属性，备忘录类B是用来存储A的一些内部状态，类C呢，就是一个用来存储备忘录的，且只能存储，不能修改等操作。做个图来分析一下：



Original类是原始类，里面有需要保存的属性value及创建一个备忘录类，用来保存value值。Memento类是备忘录类，Storage类是存储备忘录的类，持有Memento类的实例，该模式很好理解。直接看源码：

public class Original {

private String value;

public Original(String *value*) {

this.value = value;

}

public String getValue() {

return value;

}

public void setValue(String *value*) {

this.value = value;

}

public Memento createMemento() {

return new Memento(value);

}

public void restoreMemento(Memento *memento*) {

this.value = memento.getValue();

}

}

public class Memento {

private String value;

public Memento(String *value*) {

this.value = value;

}

public String getValue() {

return value;

}

public void setValue(String *value*) {

this.value = value;

}

}

public class Storage {

private Memento memento;

public Storage(Memento *memento*) {

this.memento = memento;

}

public Memento getMemento() {

return memento;

}

public void setMemento(Memento *memento*) {

this.memento = memento;

}

}

测试类：

public class Test {

public static void main(String[] *args*) {

*// 创建原始类*

Original origi = new Original("egg");

*// 创建备忘录*

Storage storage = new Storage(origi.createMemento());

*// 修改原始类的状态*

System.out.println("初始化状态为：" + origi.getValue());

origi.setValue("niu");

System.out.println("修改后的状态为：" + origi.getValue());

*// 回复原始类的状态*

origi.restoreMemento(storage.getMemento());

System.out.println("恢复后的状态为：" + origi.getValue());

}

}

输出：

初始化状态为：egg  
修改后的状态为：niu  
恢复后的状态为：egg

简单描述下：新建原始类时，value被初始化为egg，后经过修改，将value的值置为niu，最后倒数第二行进行恢复状态，结果成功恢复了。其实我觉得这个模式叫“备份-恢复”模式最形象。