设计模式-软件体系结构-实验4-中南大学-软件学院

编辑整理：

尊敬的读者朋友们：

这里是精品文档编辑中心，本文档内容是由我和我的同事精心编辑整理后发布的，发布之前我们对文中内容进行仔细校对，但是难免会有疏漏的地方，但是任然希望（设计模式-软件体系结构-实验4-中南大学-软件学院）的内容能够给您的工作和学习带来便利。同时也真诚的希望收到您的建议和反馈，这将是我们进步的源泉，前进的动力。

本文可编辑可修改，如果觉得对您有帮助请收藏以便随时查阅，最后祝您生活愉快 业绩进步，以下为设计模式-软件体系结构-实验4-中南大学-软件学院的全部内容。

《软件体系结构》

实验报告

项目名称 结构型设计模式实验

专业班级

学 号

姓 名

实验成绩:

批阅教师:

年 月 日

实验4 结构型设计模式实验

实验学时：2

每组人数：1

实验类型：3（1:基础性 2：综合性 3：设计性 4：研究性）

实验要求:1（1：必修 2：选修 3：其它）

实验类别：3 (1:基础 2:专业基础 3：专业 4：其它）

一、实验目的

熟练使用PowerDesigner和任意一种面向对象编程语言实现几种常见的结构型设计模式,包括适配器模式、组合模式和外观模式，理解每一种设计模式的模式动机,掌握模式结构，学习如何使用代码实现这些模式。

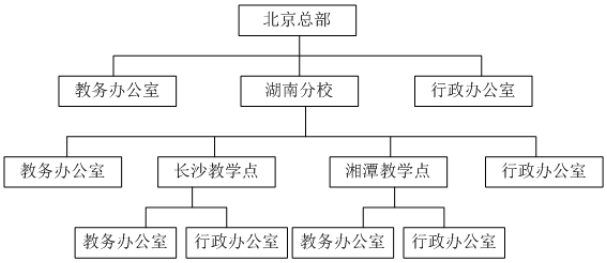
二、实验内容

1。 现有一个接口DataOperation定义了排序方法sort（int[］） 和查找方法search（int[], int），已知类QuickSort的quickSort(int[])方法实现了快速排序算法,类BinarySearch 的binarySearch(int[］, int）方法实现了二分查找算法。试使用适配器模式设计一个系统，在不修改源代码的情况下将类QuickSort和类BinarySearch的方法适配到DataOperation接口中。绘制类图并编程实现。（要求实现快速排序和二分查找，使用对象适配器实现）

2. Windows Media Player和RealPlayer是两种常用的媒体播放器,它们的API结构和调用方法存在区别。现在你的应用程序需要支持这两种播放器API，而且在将来可能还需要支持新的媒体播放器,请问如何设计该应用程序？绘制类图并编程模拟实现。

3. 使用组合模式设计一个杀毒软件（AntiVirus）的框架，该软件既可以对某个文件夹(Folder)杀毒,也可以对某个指定的文件(File)进行杀毒,文件种类包括文本文件TextFile、图片文件ImageFile、视频文件VideoFile。绘制类图并编程模拟实现。

4。 某教育机构组织结构如下图所示:



在该教育机构的OA系统中可以给各级办公室下发公文，试采用组合模式设计该机构的组织结构,绘制相应的类图并编程模拟实现，在客户端代码中模拟下发公文。

5。 某软件公司为新开发的智能手机控制与管理软件提供了一键备份功能，通过该功能可以将原本存储在手机中的通信录、短信、照片、歌曲等资料一次性全部拷贝到移动存储介质（例如MMC卡或SD卡）中。在实现过程中需要与多个已有的类进行交互，例如通讯录管理类、短信管理类等,为了降低系统的耦合度，试使用外观模式来设计并编程模拟实现该一键备份功能。

6。 某信息系统需要提供一个数据处理和报表显示模块，该模块可以读取不同类型的文件中的数据并将数据转换成XML格式，然后对数据进行统计分析，最后以报表方式来显示数据。由于该过程需要涉及到多个类,试使用外观模式设计该数据处理和报表显示模块.考虑到有些文件本身已经是XML格式,无须进行格式转换，为了让系统具有更好的扩展性，在系统设计中可以引入抽象外观类。

三、实验要求

1. 结合实例，正确无误地绘制适配器模式、组合模式和外观模式的模式结构图;

2。 使用任意一种面向对象编程语言实现适配器模式、组合模式和外观模式实例，代码运行正确无误。

四、实验步骤

1。 结合实例，使用PowerDesigner绘制适配器模式实例结构图并用面向对象编程语言实现该模式实例；

2。 结合实例，使用PowerDesigner绘制适配器模式实例结构图并用面向对象编程语言实现该模式实例；

3. 结合实例，使用PowerDesigner绘制组合模式实例结构图并用面向对象编程语言实现该模式实例；

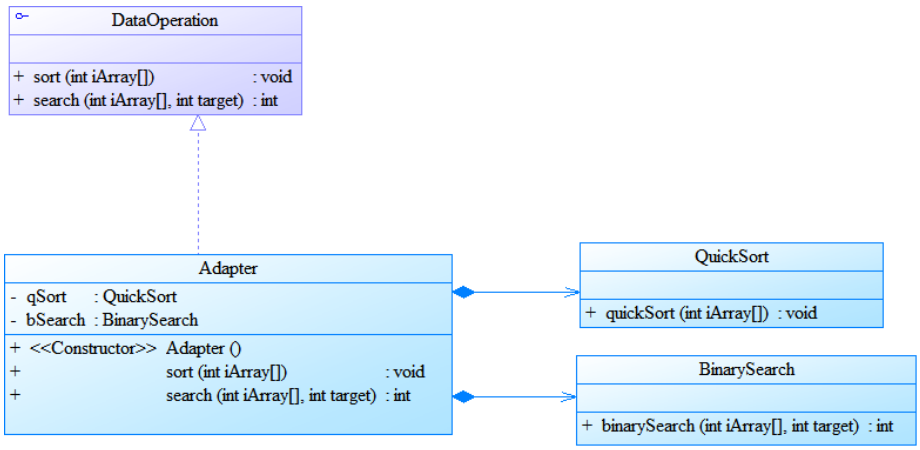
4。 结合实例，使用PowerDesigner绘制组合模式实例结构图并用面向对象编程语言实现该模式实例；

5。 结合实例，使用PowerDesigner绘制外观模式实例结构图并用面向对象编程语言实现该模式实例；

6。 结合实例，使用PowerDesigner绘制外观模式实例结构图并用面向对象编程语言实现该模式实例。

五、实验结果

1. 类图：



实现代码：

```
import util。XMLUtil;


public class Client {

public static void main（String[] args） {

DataOperation dataOperation = （DataOperation) XMLUtil.getBean();

int iArray［］ = {1,3， 2,5,4}；

boolean result = dataOperation。search(iArray,9);

for (int i = 0 ; i 〈 5 ; i ++） {

System.out。println(iArray［i］）；

}

System.out。println（result）；


}

}



public interface DataOperation {

public void sort（int［］ iArray）；

public boolean search（int［] iArray, int target）；


}
```

```
public class DataOperationAdapter implements DataOperation {
private QuickSort qSort;
private BinarySearch bSearch;

public DataOperationAdapter() {
qSort = new QuickSort();
bSearch = new BinarySearch（);
}

@Override
public void sort（int[] iArray) {
qSort。quickSort（iArray）；
}

@Override
public boolean search(int［] iArray, int target） {
return bSearch.binarySearch(iArray, target）；
}

}


public class BinarySearch {
public boolean binarySearch(int[］ iArray， int target） {
（new QuickSort()).quickSort(iArray）；
if (bSearch(iArray， target， 0, iArray.length — 1） == -1) {
return false;
}
return true;
}

public int bSearch（int［］ iArray， int target， int low, int high） {

if ( low > high） {
```

```
return -1；
}
int mid = (high + low ） / 2；
if (iArray[mid] == target) {
return mid；
}else if （iArray［mid] 〈 target） {
return bSearch（iArray， target， mid + 1, high）；
} else if (iArray[mid] 〉 target） {
return bSearch（iArray, target， low, mid — 1）；
} else {
return -1；
}
}
}
public class QuickSort {
public void quickSort(int［] iArray) {
sort(iArray, 0， iArray.length - 1）；
}
public void sort（int arr［]， int low， int high） {
int l = low；
int h = high；
int povit = arr[low］；
while (l 〈 h） {
while （l < h && arr[h］ >= povit)
h—-；
if （l 〈 h) {
int temp = arr［h]；
arr［h］ = arr[l］；
```

```
arr[l］ = temp；

l++;

}

while (l 〈 h & & arr［l] 〈= povit)

l++；

if （l 〈 h) {

int temp = arr[h];

arr[h］ = arr［l];

arr[l］ = temp；

h—-；

}

}

// print（arr）；

// System.out。print(”l=" + (l + 1） + "h=" + (h + 1) + ”povit=” + povit

// + "\n"）；

if （l > low）

sort(arr, low， h - 1);

if (h 〈 high)

sort(arr, l + 1, high);

}

}



package util;

import java.io。File;

import java。io。IOException;

import javax.xml.parsers.DocumentBuilder;

import javax。xml.parsers.DocumentBuilderFactory；

import javax.xml。parsers.ParserConfigurationException;
```

```
import org.w3c。dom.Document;

import org.w3c。dom。Node;

import org。w3c.dom。NodeList;

import org.xml。sax。SAXException;


public class XMLUtil {


public static Object getBean () {


try {

//创建DOM文档对象

DocumentBuilderFactory docFactory = DocumentBuilderFactory 。 newInstance
（）；

DocumentBuilder docBuilder = docFactory。newDocumentBuilder（）；

Document document = docBuilder。parse（new File（”config.xml"））；


//获取包含类名的 文本节点

NodeList nl = document.getElementsByTagName(”className");

Node classNode = nl.item（0）.getFirstChild（）；

String className = classNode.getNodeValue()；


//通过类名生成实例对象并将其返回

@SuppressWarnings("rawtypes")

Class clazz = Class.forName（className）；

Object obj = clazz.newInstance（）；


return obj；


} catch (ParserConfigurationException e） {

e.printStackTrace（）；

} catch （SAXException e) {

e.printStackTrace();

} catch (IOException e） {

e.printStackTrace（）；
```

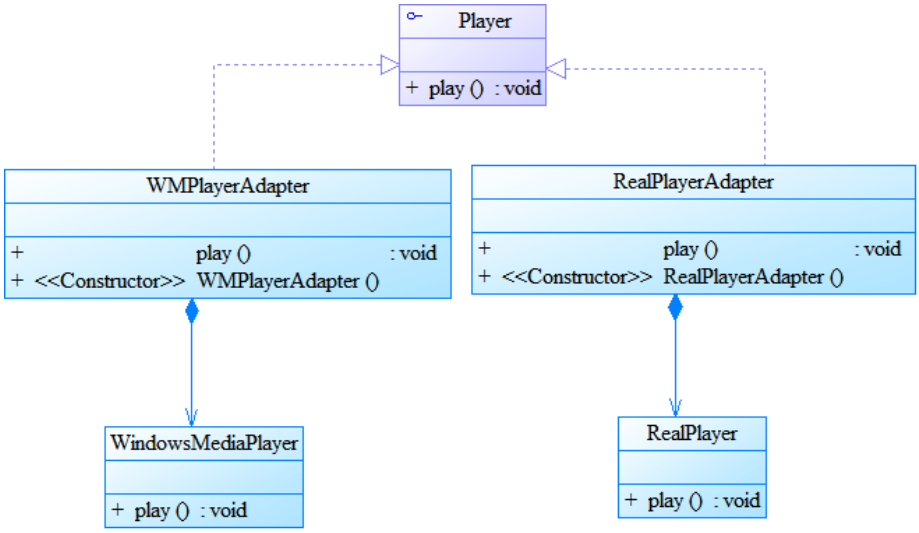
```
} catch (ClassNotFoundException e) {
e.printStackTrace（）;
} catch （InstantiationException e） {
e。printStackTrace();
} catch （IllegalAccessException e） {
e.printStackTrace()；
}

return null;
}

}

〈?xml version="1.0" encoding="UTF-8”？〉
<xml—body>
〈className>DataOperationAdapter</className〉
〈/xml—body>
```

2。 类图：



实现代码：

```
import util。XMLUtil;


public class Client {

public static void main（String［] args) {

Player player = (Player) XMLUtil。getBean（);

player。play（）;

}

}


public interface Player {

void play()；


}


public class RealPlayerAdapter implements Player {

public RealPlayer realPlayer;


public RealPlayerAdapter（） {

realPlayer = new RealPlayer();

}

public void play() {
```

```
realPlayer。play（）；
}

}

public class RealPlayer {
public void play（） {
System。out.println(”RealPlayer 。 。 .");
}

}

public class WMPlayerAdapter implements Player {
public WindowsMediaPlayer windowsMediaPlayer；
public WMPlayerAdapter （） {
windowsMediaPlayer = new WindowsMediaPlayer（）；
}
public void play() {
windowsMediaPlayer。play()；
}

}

public class WindowsMediaPlayer {
public void play（） {
System。out。println（”Windows Media Player...”);
}

}

package util；
```

```
import java。io.File;

import java.io。IOException;


import javax。xml.parsers。DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory；

import javax.xml。parsers。ParserConfigurationException；


import org.w3c。dom.Document;

import org.w3c.dom。Node；

import org。w3c.dom.NodeList;

import org。xml.sax.SAXException;


public class XMLUtil {


public static Object getBean (） {


try {

//创建DOM文档对象

DocumentBuilderFactory docFactory = DocumentBuilderFactory 。 newInstance
（）；

DocumentBuilder docBuilder = docFactory。newDocumentBuilder（);

Document document = docBuilder。parse（new File(”config。xml”)）；


//获取包含类名的 文本节点

NodeList nl = document。getElementsByTagName("className”)；

Node classNode = nl.item(0)。getFirstChild（)；

String className = classNode.getNodeValue（);


//通过类名生成实例对象并将其返回

@SuppressWarnings（"rawtypes”）

Class clazz = Class.forName(className）；

Object obj = clazz.newInstance()；


return obj；
```

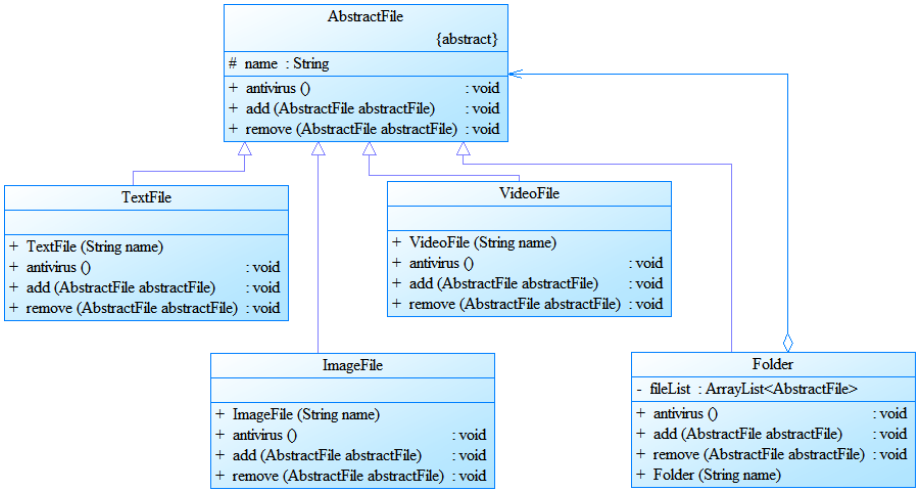
```
} catch （ParserConfigurationException e） {
e。printStackTrace（）；
} catch （SAXException e） {
e。printStackTrace（）；
} catch （IOException e） {
e.printStackTrace（）；
} catch (ClassNotFoundException e) {
e。printStackTrace（）；
} catch (InstantiationException e) {
e。printStackTrace（）；
} catch (IllegalAccessException e) {
e.printStackTrace（）;
}


return null;
}


}


〈？ xml version=”1。0” encoding="UTF—8”？〉
〈xml-body〉
<className〉 WMPlayerAdapter〈/className〉
〈/xml—body〉
```

3. 类图：



实现代码:

```
public class Client {

public static void main(String [] args) {

AbstractFile kejian = new Folder(”课件");

AbstractFile dasan = new Folder("大三课件");

AbstractFile shejimoshi = new Folder（"设计模式”);

AbstractFile shiyanyaoqiu = new TextFile("实验要求"）；

AbstractFile liucheng = new ImageFile（"实验流程图"）；

AbstractFile shinian = new VideoFile(”十年video”）；

AbstractFile sjkejian = new TextFile（”设计模式课件”）；

AbstractFile shiyanbaogao = new Folder("实验报告"）；

AbstractFile shiyanbaogao1 = new TextFile（”设计模式实验报告1”)；

AbstractFile shiyanbaogao2 = new TextFile("设计模式实验报告2"）；

AbstractFile shiyanbaogao3 = new TextFile（”设计模式实验报告3”）；

AbstractFile shiyanbaogao4 = new TextFile(”设计模式实验报告4"）；


kejian。add（dasan)；

dasan.add（shejimoshi）；

shejimoshi。add（shiyanyaoqiu）；

shejimoshi.add（liucheng)；

shejimoshi.add（shinian)；

shejimoshi。add（sjkejian）；

shejimoshi。add（shiyanbaogao)；
```

```
shiyanbaogao。add(shiyanbaogao1);

shiyanbaogao.add(shiyanbaogao2）;

shiyanbaogao。add（shiyanbaogao3）;

shiyanbaogao。add（shiyanbaogao4)；

kejian.antivirus()；

}

}



public abstract class AbstractFile ｛

protected String name；

public abstract void antivirus()；

public abstract void add（AbstractFile abstractFile）;

public abstract void remove（AbstractFile abstractFile);

}



import java.util。ArrayList;

public class Folder extends AbstractFile ｛

private ArrayList〈AbstractFile> fileList；

public Folder （String name) ｛

this.name = name；

fileList = new ArrayList〈AbstractFile〉 （）；

}

@Override

public void antivirus（） ｛
```