课程设计情况概述

课程设计题目：打字训练测试软件

课题任务分工：

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 姓名 | 主要完成任务清单 | 备注 |
| 06 | 郭淳 | 1. 报告中的程序测试、截图 2. 找所需函数的资料 3. SetJPanel.java | 组员 |
| 08 | 陈昕怡 | 1. 撰写课程设计报告 2. 程序流程图的绘制 3. UserJPanel.java | 组员 |
| 10 | 张勇 | 1. 程序的编写和组合 2. TypingTest.java 3. TypingJPanel.java 4. PauseThread.java 5. Speed\_TimeThread.java 6. InputActionListener.java | 组长 |

目 录

[课程设计情况概述 I](#_Toc360395556)

[目 录 II](#_Toc360395557)

[第1章 课题介绍 1](#_Toc360395558)

[1.1 课程设计的目的 1](#_Toc360395559)

[1.2 课程设计的要求 1](#_Toc360395560)

[第2章 总体设计 2](#_Toc360395561)

[2.1 程序流程图 2](#_Toc360395562)

[2.2 软件模块图 3](#_Toc360395563)

[2.3 子模块介绍 3](#_Toc360395564)

[第3章 详细设计及程序实现 5](#_Toc360395565)

[3.1 系统界面设计 5](#_Toc360395566)

[3.1.1 用户显示界面的设计 5](#_Toc360395567)

[3.1.2类型选择界面的设计 5](#_Toc360395568)

[3.1.3 打字输入界面的设计 7](#_Toc360395569)

[3.2 主要组件的事件监听 8](#_Toc360395570)

[3.2.1 用户显示界面的事件监听 8](#_Toc360395571)

[3.2.2 类型选择界面的事件监听 9](#_Toc360395572)

[3.2.3 打字输入界面的事件监听 10](#_Toc360395573)

[第4章 程序测试 13](#_Toc360395574)

[第5章 总 结 16](#_Toc360395575)

[参考文献 18](#_Toc360395576)

[答辩记录及评分 19](#_Toc360395577)

第1章 课题介绍

1.1 课程设计的目的

掌握java面向对象程序设计的概念，将所学的java知识用到实际的应用中，设计一个小型应用程序。此课题主要用到图形界面，多线程处理，输入输出流，容器及常用类的使用，熟悉基本组件及布局管理方式的应用，巩固java的基础知识。

1.2 课程设计的要求

1）可实现用户名的自定义设置或默认身份的进入 。

2）可实现测试类型的选择、通过测试文章的下拉列表选择已导入目录的文章和用户本地导入、测试时间的自定义。

3) 监听你的键盘录入，同时启动线程去判断用户输入的字符与模板上的字符是否一致，不正确的用红色标记，正确的显示为绿色。

4) 启动测试时记录起始时间，结束时获得结束时间，通过时间差统计出错误率和平均录入的速度。

5）可实现重新开始打字练习、暂停打字练习并随时可恢复练习功能的实现。

6）可实现用户所有打字记录被保存并能查询。

7）可实现能在界面之间切换。

第2章 总体设计

2.1 程序流程图

程序启动后进入用户显示界面，输入用户名后可进入类型选择界面，设置完成后就进入打字模块，按任意键就进入打字练习并开始启动及计时线程进行计时及打字状态的显示。详细的程序流程图如图2-1所示：



**图2-1 程序流程图**

2.2 软件模块图

要实现对用户结果的记录，需要设置用户名及一些类型选择（如用户自定义文章和自定义时间），将此两点与最终的打字界面单独分割开，确定要设计三个界面模块，用户显示模块、类型选择模块、打字输入模块。软件模块如图2-2所示：



**图2-2 软件模块图**

2.3 子模块介绍

1、用户显示模块：此模块设计为一个卡片布局，将加入此模块的面板jp约束在中间，并在面板上添加控件。此模块可进行用户名的设置，默认以随机的游客身份进入。点击“提交”按钮后进入打字测速练习软件，点击“重置”按钮可重新输入用户名。

2、类型选择模块：此面板也是一个卡片布局，将jp面板约束在此面板的中间位置，在jp面板里面进行添加控件，jp面板的布局是网格布局，采用四行三列的形式。此模块可以选择测试类型、测试文章、测试时间，及是否选择从本地导入文章。测试类型有“英文练习”和“中文练习”两种选择，“中文练习”功能尚未实现。可通过测试文章的下拉列表选择已导入目录的文章，也可以点击导入文章进行本地导入文本文件作为测试文章。用户可自定义测试时间。之后点击“开始”按钮进入打字界面，开始进行打字测速练习。点击“重置”按钮所有选择返回默认状态用户可重新自定义。点击“返回”按钮返回之前的用户显示模块。

3、打字输入模块：此面板是边框形式布局，中间设计字符的显示与接受用户输入的文本框。此模块可以与用户进行交互，响应用户的输入。进入打字模块后直接进行打字训练，按Enter回车键可进入下一行。练习的同时，界面上方的“速度”和“正确率”以秒为单位进行更新。并且有“剩余时间”来提醒用户剩余练习时间。根据用户输入的正确与否，字体显示不同的颜色。打字正确时显示绿色，错误则显示红色。按“暂停”按钮所有的文本框处于不可接受用户输入状态，此时“暂停”按钮将变成恢复按钮，等待用户返回继续练习。“文件”按钮包含“返回”和“退出”两个按钮，用户可分别进行返回上一届面和整体退出程序两个功能的选择。“设置”按钮可进行“重新开始”项的选择，用户之前测试结果被记录到文件的同时，所有的状态会恢复成新的界面。通过“查询”按钮则可以进行之前所有打字记录的查询，包括“用户名”、“速度”、“正确率”、“使用时间”、“测试时间”这五个数据。当点击“帮助”菜单栏里面的“帮助文档”项时，可以查看帮助文档，从而获得帮助提示。点击帮助文档中的“确定”按钮可返回练习。

这三个模块全部放在一个卡片布局的面板里，当用户点击各个按钮时，分别显示不同的模块界面。

第3章 详细设计及程序实现

3.1 系统界面设计

3.1.1 用户显示界面的设计

用户界面显示界面，约束此界面在主界面的中间显示。在此面板首先添加主题打字测试练习，然后添加一些提示和接受用户的组件，以及响应用户动作的组件。用户名有默认的产生方式，提交和重置按钮是以居中的方式放进面板。详细代码如下：

public UserJPanel(TypingTest tt) {

this.tt = tt;

this.setLayout(userCardLayout);

JPanel jp = new JPanel(new GridLayout(3,1));

welcomJLabel = new JLabel("打字测速练习", SwingConstants.CENTER);

welcomJLabel.setFont(new Font("宋体", Font.BOLD, 20));

jp.add(welcomJLabel);

JPanel jp1 = new JPanel(new FlowLayout(FlowLayout.CENTER));

userNameJLabel = new JLabel("用户名:");

userNameJTextField = new JTextField(12);

userNameJTextField.setText("游客" + random);

userNameJTextField.setToolTipText("默认以随机产生的游客登陆，可以 设定用户名");

jp1.add(userNameJLabel);

jp1.add(userNameJTextField);

jp.add(jp1);

JPanel jp2 = new JPanel(new FlowLayout(FlowLayout.CENTER, 30, 0));

submitJButton = new JButton("提交");

resetJButton = new JButton("重置");

jp2.add(submitJButton);

jp2.add(resetJButton);

jp.add(jp2);

this.add(jp);

setListenerForSubmitJButton();

setListenerForResetJButton();

}

3.1.2类型选择界面的设计

类型选择界面面板的布局时 4行3列的网格布局，向面板里面依次添加测试文章类型（英文练习），这是默认的选择方式。测试文章选择，支持用户从本地导入自己的文章（仅支持纯文本的英文文本）。用户可以自己设定要进行测试的时间，默认的时间是5分钟。详细代码如下：

public SetJPanel(UserJPanel ujp) {

this.ujp = ujp; //持有对方的引用，为了获得用户名和主面板

this.setLayout(setCardLayout);//约束作用

JPanel jp = new JPanel(new GridLayout(4,3));

FlowLayout leftAlign = new FlowLayout(FlowLayout.LEFT, 3 , 22);

testTypeJLabel = new JLabel("测试类型:", SwingConstants.RIGHT);

englishJRadioButton = new JRadioButton("英文练习");

chineseJRadioButton = new JRadioButton("中文练习");

englishJRadioButton.setSelected(true);

ButtonGroup bg = new ButtonGroup();

bg.add(englishJRadioButton);

bg.add(chineseJRadioButton);

jp.add(testTypeJLabel);

jp.add(englishJRadioButton);

jp.add(chineseJRadioButton);

testArticleJLabel = new JLabel("测试文章:", SwingConstants.RIGHT);

JPanel jp1 = new JPanel(leftAlign);

testArticleJComboBox = new JComboBox<String>();

jp1.add(testArticleJComboBox);

testArticleJComboBox.setEditable(false);

testArticleJComboBox.setMaximumRowCount(7);

addEnglishArticleToJComboBox(); //向下拉列表中添加文章标题

JPanel jp2 = new JPanel(leftAlign);

importJButton = new JButton("导入文章");

jp2.add(importJButton);

jp.add(testArticleJLabel);

jp.add(jp1);

jp.add(jp2);

testTimeJLabel = new JLabel("测试时间:", SwingConstants.RIGHT);

JPanel jp3 = new JPanel(leftAlign);

testTimeJTextField = new JTextField(10);

testTimeJTextField.setText("5");

testTimeJTextField.setToolTipText("测试时间默认为五分钟");

jp3.add(testTimeJTextField);

jp.add(testTimeJLabel);

jp.add(jp3);

jp.add(new JLabel(""));

FlowLayout centerAlign = new FlowLayout(FlowLayout.CENTER, 3, 22);

JPanel jp4 = new JPanel(centerAlign);

startJButton = new JButton("开始");

jp4.add(startJButton);

JPanel jp5 = new JPanel(centerAlign);

resetJButton = new JButton("重置");

jp5.add(resetJButton);

JPanel jp6 = new JPanel(centerAlign);

returnJButton = new JButton("返回");

jp6.add(returnJButton);

jp.add(jp4);

jp.add(jp5);

jp.add(jp6);

this.add(jp);

setListenerForTestArticleJComboBox();//为下拉列表注册事件监听

setListenerForResetJButton(); //为重置按钮注册事件监听

setListenerForStartJButton(); //为开始按钮注册事件监听

setListenerForReturnJButton();//为返回按钮注册事件监听

setListenerForImportJButton();//为导入按钮注册事件监听

}

3.1.3 打字输入界面的设计

此面板的布局是边框布局，上面设计的显示的状态，中间的面板是用createCenterJPanel()方法实现的，createCenterJPanel()方法里面先读取文件的内容，记录文件中的字符数，然后你每行70个字符的布局形式将文件中的所有字符显示上去，每个字符是放置一个JLabel里面，每行字符对应一个文本输入框，根据用户的输入正确性，JLabel的字符显示不同的颜色。详细代码如下：

public TypingJPanel(SetJPanel sjp) {

this.sjp = sjp; //持有前一面板对象的引用

this.fileName = sjp.fileName;

this.userName = sjp.ujp.userName;

createJMenuBar(); //创建菜单栏

JPanel jp = new JPanel(new BorderLayout());

JPanel topJPanel = new JPanel(new GridLayout(1, 5));

userNameJLabel = new JLabel("用户名:

" + userName,SwingConstants.LEFT);

speedJLabel = new JLabel("速度:", SwingConstants.LEFT);

accuracyJLabel = new JLabel("正确率:", SwingConstants.LEFT);

timeJLabel = new JLabel("剩余时间:", SwingConstants.LEFT);

pause\_resumeJButton = new JButton("暂停");

// 在开始测试之前设置暂停按钮为不可接受状态

pause\_resumeJButton.setEnabled(false);

JPanel jp\_pause\_resume = new JPanel(new CardLayout(40, 2));

jp\_pause\_resume.add(pause\_resumeJButton);

topJPanel.add(userNameJLabel);

topJPanel.add(speedJLabel);

topJPanel.add(accuracyJLabel);

topJPanel.add(timeJLabel);

topJPanel.add(jp\_pause\_resume);

jp.add(topJPanel, BorderLayout.NORTH);

createCenterJPanel();//创建中间面板

jsp = new JScrollPane(centerJPanel);

jsp.setVerticalScrollBarPolicy(JScrollPane.

VERTICAL\_SCROLLBAR\_ALWAYS);

jsp.setWheelScrollingEnabled(true);

jsp.setAutoscrolls(true);

jsp.setPreferredSize(new Dimension(830, 500));

jp.add(jsp, BorderLayout.CENTER);

centerJPanel.revalidate();

this.add(jp);

inputActionListener = new InputActionListener(this);

setListenerForPause(); // 为暂停按钮注册事件监听

setListenerForSearchJMenuItem(); // 为查询记录项注册事件监听

setListenerForRestartJMenuItem(); // 为重新开始项注册事件监听

setListenerForExitJMenuItem(); // 为退出项注册事件监听

setListenerForReturnJMenuItem(); // 为返回项注册事件监听

setListenerForHelpJMenuItem(); // 为帮助项注册事件监听

}

3.2 主要组件的事件监听

3.2.1 用户显示界面的事件监听

下面的方法是为用户界面中提交按钮注册事件监听，当此组件被触发时，将首先检测用户名是否合法，在用户名合法的基础上建立一个设置界面放在卡片布局的主面板里面，并显示设置界面。详细代码如下：

public void setListenerForSubmitJButton() {

submitJButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(userNameJTextField.getText() != null

&& !userNameJTextField.getText().trim().equals("")){

userName = userNameJTextField.getText();

} else {

JOptionPane.showMessageDialog(null,

"你没有输入用户名，或用户名为空！",

"提示", JOptionPane.OK\_OPTION);

return;

}

SetJPanel setJPanel = new SetJPanel(UserJPanel.this);

tt.mainJPanel.add("setJPanel", setJPanel);

tt.cardLayout.show(tt.mainJPanel, "setJPanel");

}

});

}

3.2.2 类型选择界面的事件监听

下面的方法是为设置界面中的导入文章按钮注册事件监听，当此组件被触发时，将从本地文件中导入一个文件进行练习，如果没有选择文件则返回，如果选择了文件，则将以选择文件和已知的文件后缀名进行匹配，如果符合纯文本文件的后缀名，则将所选择文件名记录。详细代码如下：

public void setListenerForImportJButton() {

importJButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

JFileChooser fileChooser = new JFileChooser();

boolean isPureText = false;

String getPath;

fileChooser.setMultiSelectionEnabled(false);

if(fileChooser.showOpenDialog(null)

== JFileChooser.CANCEL\_OPTION) {

return;}

getPath = fileChooser.getSelectedFile().toString();

for(int i = 0; i < pureTextSuffix.length; i++) {

if(getPath.endsWith(pureTextSuffix[i])) {

isPureText = true;

break;}

}

if(isPureText) {

fileName = getPath;

JOptionPane.showMessageDialog(null, "导入"+getPath +

"文件成功","提示",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(null,

"你没有选择纯文本文件，无法导入！",

"警告", JOptionPane.ERROR\_MESSAGE);

}

});

}

下面的方法是为开始按钮注册事件监听，当此按钮被触发时，将会检测你设定的格式是否符合规定的格式，如果不符合则会给予一定的提示，在设定正确前提下，新建一个打字面板放在主面板中，并将此打字面板显示，进入打字练习状态。在新建打字面板之前会检测你是否从打字面板返回，如果是返回，则将上一次打字面板从主面板中移出。详细代码如下：

public void setListenerForStartJButton() {

startJButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

try {

setTime = Double.parseDouble(

testTimeJTextField.getText());

} catch (NumberFormatException e1) {

JOptionPane.showMessageDialog(null,

"您输入时间格式不对请重新输入，时间以分钟为单位",

"提示", JOptionPane.INFORMATION\_MESSAGE);

return;

}

if(typingJPanel != null) {

ujp.tt.mainJPanel.remove(typingJPanel);}

typingJPanel = new TypingJPanel(SetJPanel.this);

ujp.tt.mainJPanel.add("typingJPanel", typingJPanel);

ujp.tt.cardLayout.show(ujp.tt.mainJPanel

, "typingJPanel");

}

});

}

3.2.3 打字输入界面的事件监听

下面的方法是为暂停恢复按钮注册事件监听，当此暂停组件被触发式，置暂停标志为true，同时所有的输入文本框将不会接受用户输入，置按钮的显示文字为恢复。当恢复按钮被触发式，置暂停标志为false，所有文本输入框恢复可接受用户输入状态，同时找到上一次暂停时文本框的引用，将其设置为请求输入状态，按钮的显示文字也会跟着改变为暂停。详细代码如下：

public void setListenerForPause() {

pause\_resumeJButton.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

JButton jb = (JButton) e.getSource();

if (jb.getText().equals("暂停")) {

inputActionListener.isPause = true;

pause\_resumeJButton.setText("恢复");

for (int i = 0; i < rows; i++) {

// 暂停后使各输入框处于不可编辑状态

inputJTextField[i].setEditable(false);}

} else {

inputActionListener.isPause = false;

pause\_resumeJButton.setText("暂停");

for (int i = 0; i < rows; i++) {

inputJTextField[i].setEditable(true);

if (inputActionListener.jtf == inputJTextField[i]){

// 恢复之后找到当前要输入的文本框

inputJTextField[i].requestFocusInWindow();

}

}

}

}

});

}

下面的方法是为设置菜单里面的重新开始项注册事件监听，当此项被触发时，会让用户确定是否要重新开始，当用户点击取消按钮时，则继续上一次的练习状态，否则将重新开始一次打字训练。重新开始后，将是否是第一个键的标志设为false，将重新开始的标志设为true，所有的输入文本框要恢复处于可编辑状态和可编辑状态，所有字体设置成正常的状态。同时要跟新按钮的状态，重新设置暂停和恢复按钮为暂停状态，还要去跟新滚动条的位置，将滚动条置顶，使下次开始从头开始，测试文字在用户的可见状态。详细代码如下：

public void setListenerForRestartJMenuItem() {

restartJMenuItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int ok = JOptionPane.showConfirmDialog(null, "你确定要重

新开始吗?", "重新开始确认"

, JOptionPane.YES\_NO\_OPTION);

if (ok == JOptionPane.OK\_OPTION) {

inputActionListener.isFirstKey = false;

inputActionListener.isAgainStart = true;

for (int i = 0; i < rows; i++) {

inputJTextField[i].setText("");

inputJTextField[i].setEditable(true);

inputJTextField[i].setEnabled(true);

}

for (int i = 0; i < wordlength; i++) {

wordJLabel[i].setForeground(PLAINCOLOR);

if (wordJLabel[i].getText().equals(" ")) {

wordJLabel[i].setBackground(wordJLabel[0].

getBackground());

}

}

inputJTextField[0].requestFocusInWindow();

pause\_resumeJButton.setText("暂停");

inputActionListener = new InputActionListener(

TypingJPanel.this);

jsp.getVerticalScrollBar().setValue(0);

jsp.getVerticalScrollBar().validate();

jsp.getVerticalScrollBar().repaint();

}

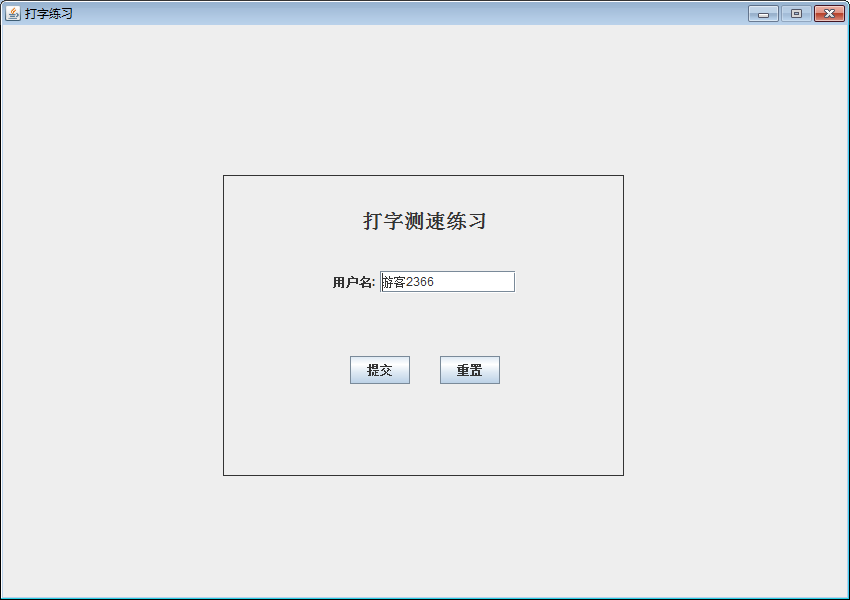
}

});

}

第4章 程序测试

打开打字训练测试软件，运行jar包，主界面显示用户登录的界面。通过提交用户名，从而登录软件进入打字测速练习。用户名显示界面如图4-1所示：



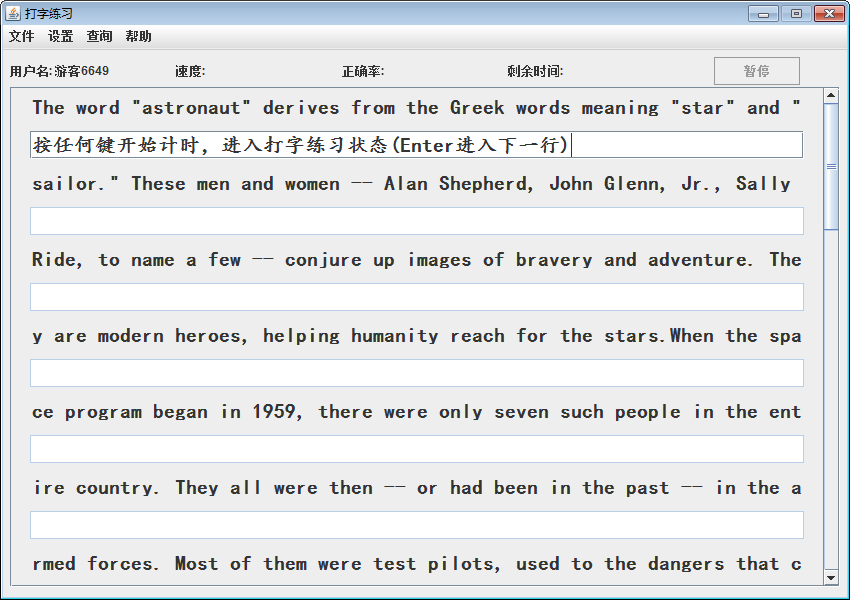
**图4-1 用户名显示界面**

在用户登录后，进入类型选择界面。用户可以选择测试类型、测试文章、测试时间，从而开始进行打字测速练习。类型选择界面如图4-2所示：



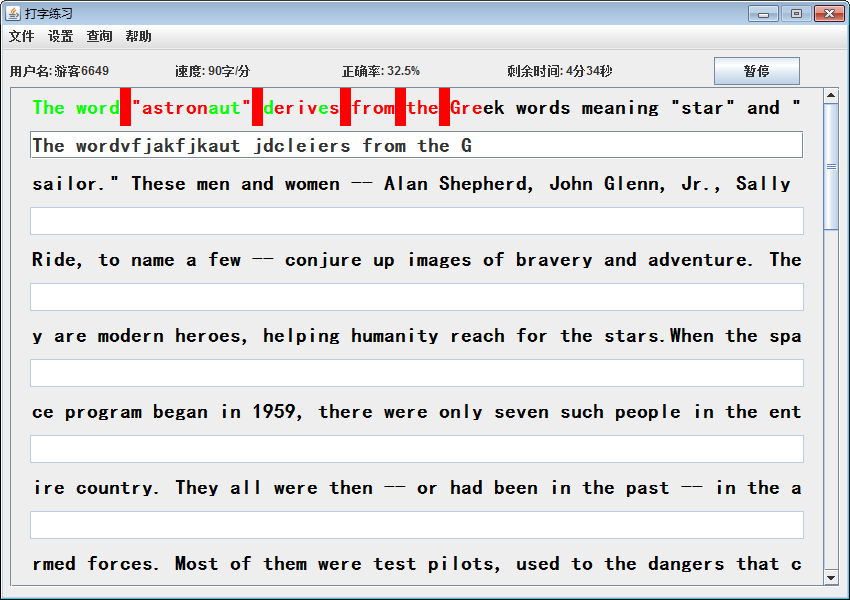
**图4-2 类型选择界面**

在类型选择界面，用户可以通过点击“导入文章”按钮，进行本地导入文本文件作为测试文章。在类型选择界面完成后，点击“开始”按钮，从而进入打字界面。打字界面如图4-3所示：

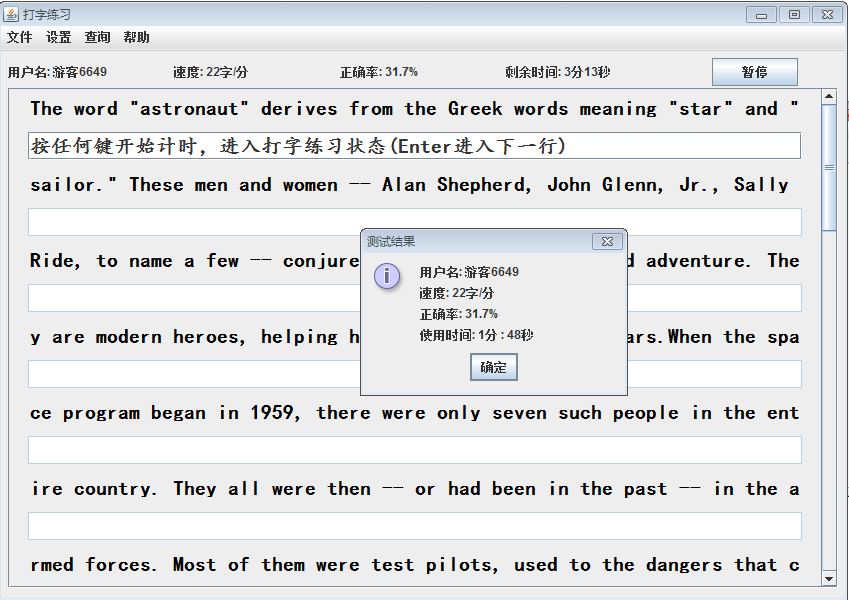


**图4-3 打字界面**

进行打字训练时，根据用户输入的正确与否，字体显示不同的颜色。最上面的一栏是显示当前的状态信息，用户名，速度，正确率和剩余时间。打字训练界面如图4-4所示：

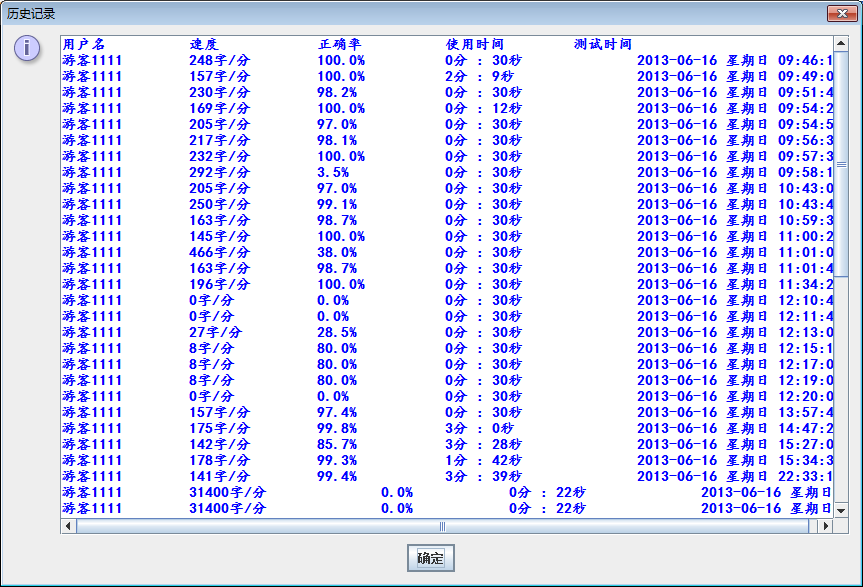


**图4-4 打字训练界面**

当点击“设置”按钮，选择菜单栏里面的“重新开始”项时，会将上次的测试结果记录到文件中，并且所有的状态会恢复成新的界面，结果如图4-5所示：

**图4-5 重新开始图**

当点击“查询”菜单栏里面的“历史记录”项时，可以查看所有的历史记录，所有的记录保存在一个文件里面，结果如图4-6所示：



**图4-6 查看记录图**

第5章 总 结

在这两周的Java课程设计中，小组配合默契，共同完成了一个知识性、应用性及趣味性较强的程序。不仅使Java语言知识得到了巩固，解决能力也有了一定的提高。

设计的打字训练测试软件，主要的思想是从文件中调入英文录入材料，并将其显示到图形界面的窗口中，在用户打字结束后，根据用户的按键依次可以给出错误率，以及平均录入速度。在使用此程序的过程中，用户可以了解自己打字速度的水平，从而提高打字速度。

Java语言是这学期学习的新语言，一开始对要设计的打字训练测试软件没有什么头绪。但在课程设计开始前，就做了充足的准备，在课程设计过程中，通过图书馆查阅资料和上网搜索信息，同时还请教王老师和一些同学，逐渐理清了思路，编写好了大致的程序，完成了界面等框架，编写了设定用户名、暂停、速度测试等功能的程序。

课程设计正式开始后，又在原来的程序基础上，增加了文件导入的功能。任务书中要求编写英文的打字训练测试软件，在完善程序时想编写中文的打字训练测试软件，但由于中文的字节和英文的有所不同，而的键响应功能不能改变，因此中文打字功能不能实现。在后期对文件导入处理上，可以同时导入中英文文件，而界面上只显示英文，从而实现英文的打字训练测试软件。

编写程序的过程中遇到了很多问题，但是通过查资料和与同学讨论都一一解决了，主要问题如下：

Q1：开始写程序之前，要设计总体窗口，开始不知是在各个窗口之间切换还是用其它的方式？

S1：翻看课本，看到一个卡片布局的管理器，一个面板里面可以放很多面板，并且可以在不同的面板里面切换，所以选择这个方式来设计主界面，当点击不同的按钮时，卡片布局管理器响应不用的面板，这个可以实现在不同面板之前的切换。

Q2:将面板添加到面板里面时，如何让被添加的面板只显示在添加面板的中间？

S2:通过查看API文档，发现卡片布局的管理器可以约束一个组件与前一个组件的水平距离和垂直距离，通过约束水平距离和垂直距离的方式，可以将面板放在另一个面板的中间位置。通过此种方式也带来了一些问题，JDK1.7上面可以调试通过，低版本的JDK不支持此类语法。

Q3：在设置打字面板，考虑要用什么方式将字符添加上去，要实现字符可以变成其它的颜色？

S3：开始是将字符放在JTextArea组件中的，下面的打字输入也是JTextArea的，但是发现，JTextArea里面的字体是不能用不同颜色的形式显示的，查看资料发现JTextPane组件可以实现放置不同颜色的文字，但这种方式也是有问题的，无法实现字体的匹配问题（设置字体大小的方式没有去尝试），最终小组讨论决定用JLabel的方式，将所有的字符全部放在一个JLabel里面，根据用户输入的正确性，JLable显示不同的前景色和背景色。当JLabel字符非空时，颜色显示没什么问题，但当JLabel中字符为空时，不能显示任何颜色，查看资料发现，必须要JLabel设置透明状态，这样才能使空的JLabel显示成其它的颜色。采用每行70个JLabel的形式放在一个JPanel里面，然后把JPanel放在一行，然后以一个JTextField与之对应。

Q4：文本可以输入很多字符，与上面的JLabel一行70个会产生不对应的情况，还有字体的情况。

S4：解决方式是当文本输入框输入有70个字符时，就让其变成不可编辑状态，按退格键让其恢复可编辑状态，不对应情况是用设置指定字体大小的方式解决的。

Q5：从本地导入文章这里，因为程序不能支持中文打字，所以要对导入的文章进行处理？

S5：解决方式是将导入的文章一行一行的判断，如果一行有中文的话，这一行将会被舍弃。还有个判断是只能导入纯文本文件，会判断文件的后缀名。

Q6：如何产生滚动条？滚动条如何与你打字同步？

S6：因为打字面板的大小要大于界面可显示的大小，所以要用到滚动条，如果只是简单的添加滚动条，无法达到预设的效果。在网上查看一些资料获知，必须要设置JScrollPane的首选大小，当添加到其它的面板大于其首选大小时，会自动的添加滚动条。打字同步的问题是通过设置JScrollBar的值来控制的，每当用按回车键时，滚动条的值就会增加一个指定的大小，这样就可以实现滚动条自动滚动了。

本次课程设计在编写程序时，常常会遇到一些bug，通过不断的调试和运行，来完善程序。非常感谢这两周来王老师的指导和帮助，以及同组同学的付出。虽然遇到过一些挫折，但互相配合、互相帮助，本次课程设计完成得十分顺利。

参考文献

[1] 张桂珠，张平，陈爱国.Java面向对象程序设计(JDK 1.6)（第3版）.北京：北京邮电大学出版社，2010.8

[2] 贾振华，庄连英，李杰，史永生.Java语言程序设计（第二版）.北京：中国水利水电出版社，2010.1

[3] 林信良. Java学习笔记. 北京:清华大学出版社，2012.5

答辩记录及评分

1. **问：**设置面板大致思想？

**郭淳回答：**设置一个卡片面板将一个4行3列的网格布局约束在中间，网格布局中加入了各个组件，其中五个组件设置了项目或事件监听。

**问：**设置面板中的矩形框如何生成？

**郭淳回答：**加入Graphics方法构造一个矩形，不断重画。

**问：**为什么需要不断重画？

**郭淳回答：**第一次构造后，组件之间会互相影响使矩形框部分消失，所以需要不断重画保持矩形完整。

**问：**EnglishArticleToJComboBox方法是在哪里激发？

**郭淳回答：**在下拉列表导入文章时。

**问：**下拉列表的项目监听中为什么要判断是否为最终选择状态？

**郭淳回答：**JComboBox的事件激发有两次，一次表示之前的选项不再是选择状态了，第二次表示当前的选项被选中了，为了不使事件多次进行没必要的激发，设置了一个if语句判断是否为最终选择状态，若是最终状态事件才被激发。

1. **问：**在用户登录时，用户名中的随机数是如何产生的？

**陈昕怡回答：**在public int random = (int)(Math.random() \* 9000) + 1000这条语句中，Math.random产生0至1间的随机数，0取得到，1取不到，从而产生1000到9999的随机数。

**问：**userNameJTextField.setToolTipText("默认以随机产生的游客登陆，可以设定用户名");这条语句是什么作用？

**陈昕怡回答：**当鼠标在用户名后的框上停留一会时，会有提示：“默认以随机产生的游客登陆，可以设定用户名”,帮助用户进行用户名正确输入。

**问：**如何判断用户名是否输入错误？

**陈昕怡回答：**利用if语句来判断用户名是否输入错误，用户名在两种情况下输入错误。一种是在没有输入用户名，即用户名为空时，另一种情况是当用户名在去除字前字后为空，即用户名全为“空格”时，系统会提示“你没有输入用户名，或用户名为空！”

1. **问：**程序总体分几个类，总体是如何设计的？

**张勇回答：**程序总体分7个类，主类TypingTest是一个JFrame框架，这个框架是一个卡片布局，由网格布局将框架约束在中间。UserJPanel和SetJPanel分别为用户显示面板和类型选择面板。通过UserJPanel输入用户名登陆，进入SetJPanel面板，进行测试类型、文章、时间的选择。之后点击“开始”按钮，开始进入打字测试。在打字输入界面TypingJPanel，其中分别用类PauseThread暂停、Speed\_TimeThread测速，InputActionListener监听。

**问：**输入监听和计时线程类是如何实现的？

**张勇回答：**输入监听是对键进行监听的，每输入一个键就开始响应，对输入的键的字符和指定字符比较，如果正确，则设置标签的前景色为绿色，否则为红色，但是这不能实现标签背景色的显示，必须要设置标签为透明状态才能显示背景颜色。如果用户输入回车键，则换到下一行，上一行就处于不可编辑状态，没有输入的字符为错误。退格键可以将用的输入清除掉，返回开始的状态，当一行文本的输入达到70个字符时，不再接受用户输入，按退格键可以实现让用户再次输入。计时线程作用是显示剩余时间、正确率和速度，当用户点击第一个键时，计时线程就开始运行，当打字测速的时间小于用户设置的制定时间时，一直处于运行状态，每隔一秒就跟新一下状态栏的信息，正确率的实现是每秒检测一下所有标签的颜色状态，绿色是正确的。用accyury记录正确单词总数，用正确的字符总数除以当前输入字符的总数就得到正确率。剩余时间是用设置时间减去打字测试时间，打字测试时间时由当前时间减去开始时间和暂停的时间。速度是由输入的单词总数除以测试的时间。计时线程在执行过程中会判断用户是否点击暂停按钮，如果用户点击了暂停按钮，就是另起一个暂停线程来阻塞当前线程，直至用户点击恢复按钮，暂停线程就会终止运行，计时线程重新得到执行。计时线程里面也会判断是否文章结束，是否点击了重新开始和返回按钮，点击了就终止计时线程，重新回到新的一轮打字状态。

课程设计评分：

|  |  |  |
| --- | --- | --- |
| 学 号 | 姓 名 | 分 数 |
| 06 | 郭淳 |  |
| 08 | 陈昕怡 |  |
| 10 | 张勇 |  |