

实验 3. JAVA-GUI 开发

实验目的

- (1) 掌握 Java 的 swing 界面开发步骤:
 1. 生成 JFrame 子类作为界面;
 2. 设计中间容器, 为中间容器设置布局;
 3. 添加组件;
 4. 为组件添加事件处理。
- (2) 熟悉使用 JDK_API 进行软件开发。

实验步骤

- 步骤 1:** 创建工程, 工程名以学号姓名方式命名, “2020010505-薛飞宇”。
- 步骤 2:** 为每道实习题目创建对应的包, 如“work1”、“work2”……;
- 步骤 3:** 按照要求创建源代码, 进行编写, 注意编码格式, 如缩进、命名规范等;
- 步骤 4:** 规范书写实习报告;
- 步骤 5:** 实现与测试。附上代码或以附件的形式提交, 同时贴上必要的代码运行截图;
- 步骤 6:** 及时总结心得体会与备忘。

实验过程

实验题 1 Focus 事件处理练习

本实验主要编写了一个用户注册界面, 根据用户输入的信息, 判断是否匹配合法的正则表达式, 然后根据相应的事件监听器对事件做出相应的处理。

通过本实验, 我也熟悉了 JAVA 界面开发的基本流程, 如下:

1. 编写一个类继承 JFrame
2. 编写构造方法, 构造方法中调用 5 个方法
 - (1) `setDefaultCloseOperation(EXIT_ON_CLOSE);`
 - (2) `setSize();`
 - (3) `setLocationRelativeTo(null);`
 - (4) `init();`
 - (5) `setVisible(true);`其中, `init()`方法中有自己实现的所有关于界面的代码, `setVisible(true)`要写在最后, 否则有些组件将会卡 bug, 不可见。
3. 在 `init()`方法中要实现有关界面的所有代码。
 - (1) 首先是 `setLayout()`, 确定界面总的布局框架
 - (2) 其次, 在设置的 Layout 中 add 相应的中间容器 JPanel
 - (3) 然后, 在 JPanel 中 add 程序需要的组件
 - (4) 最后, 对有关组件注册事件监听器

其中，实现事件监听器有多种方案：

- 内部类。这类监听器类将所有事件都放在一个程序块中，通过 `getSource()` 获取事件源。
- 外部友好类。和普通的类一样，对外部对象可见。
- 匿名类。十分先进的编程方法，用法类似于 `lambda` 表达式，实际开发中广泛使用。

4. MVC 结构。

MVC 分别代表模型、视图、操作。在实际开发环境中，将模型、视图和操作分开具有很大的意义。有利于实现高聚合、低耦合的代码结构，提高代码重用性。

废话不多说，我们先来看成果。

The image shows a standard registration form in a graphical user interface. The window has a title bar with the text '注册' and standard window controls. The form is centered and consists of four text input fields, each preceded by a label: '用户名:', '密码:', '确认密码:', and '邮箱:'. At the bottom of the form, there are two rectangular buttons with rounded corners, labeled '确认' and '取消'.

图 1.1 注册界面

注册

用户名:

用户名必须是不含空字符的6-20位字符串

密码:

密码必须是8-20位的可含标点符号或单词字符的字符串

确认密码:

确认密码必须与密码相同

邮箱:

请输入合法的邮箱地址

确认 取消

图 1.2 事件处理

注册

用户名:

用户名必须是不含空字符的6-20位字符串

密码:

密码必须是8-20位的可含标点符号或单词字符的字符串

确认密码:

邮箱:

ERROR

请重新填写!

确定

确认 取消

图 1.3 注册失败



注册

用户名: zhaojianbanghaoshuai

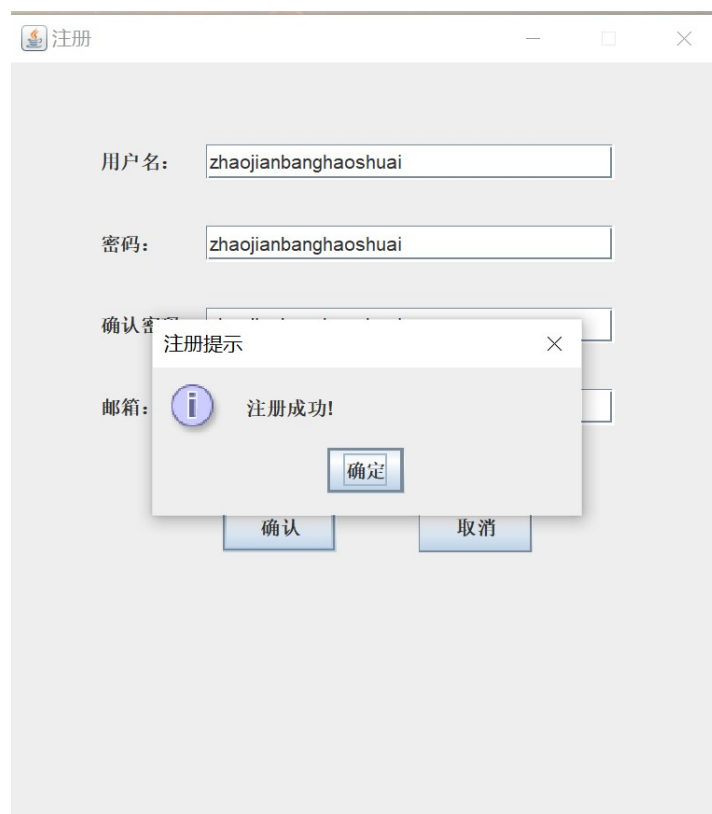
密码: zhaojianbanghaoshuai

确认密码: zhaojianbanghaoshuai

邮箱: shuai@qq.com

确认 取消

图 1.4 正确匹配



注册

用户名: zhaojianbanghaoshuai

密码: zhaojianbanghaoshuai

确认密码:

邮箱:

注册提示

注册成功!

确定

确认 取消

图 1.5 注册成功

经过一系列测试，程序的基本功能已经符合预期。但是还有以下可以改进的点：

- 密码框和确认密码框可以使用 JPasswordField
- 当正则表达式匹配，在文本框右侧显色绿色 ✓

下面，说一下设计思路。

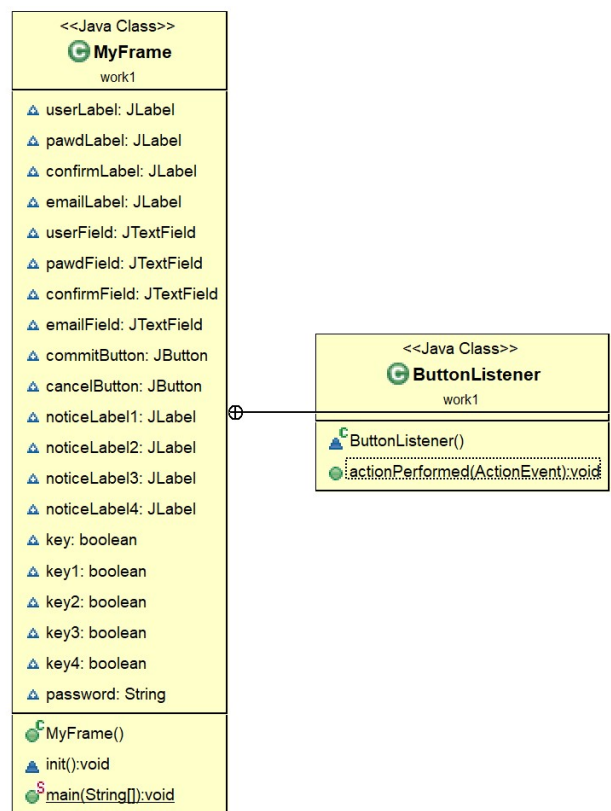


图 1.5 实验一 UML 类图

由于本程序不涉及对模型和数据的增删改查，故使用一个类 **MyFrame** 实现所有功能。首先，声明本程序需要的实例成员，包括 **JLabel**，**JTextField**，**JButton** 等。对这些基本组件通过空布局和 **setBounds()**方法设置他们在界面的位置和大小。在这里不再赘述。

然后，对相关的组件注册事件监听器。我使用了两种方法—对于 **JButton** 对象，采用内部类实现事件监听器。对于 **JTextField** 对象，采用匿名内部类实现事件监听器。经过比较，匿名内部类的方法更加方便和适合程序开发。

最后，对于 **JButton** 对象，事件监听器添加弹出窗口的操作，对于 **JTextField**，添加正则表达式的判断。根据题意，本程序需要的正则表达式分别如下：“**\\S{6,20}**”，**"[\\p{P}a-zA-Z]{8,20}"**，**"^[a-zA-Z0-9_-]+@[a-zA-Z0-9_-]+(\\.[a-zA-Z0-9_-]+)+\$"**，由于上次实验已详细说明了正则表达式的使用方法，这里不再赘述。

本程序还有一个细节值得注意。对于每一个 **focusLost** 事件，都设置一个 **key** 来判断正则表达式是否匹配成功，只有当 4 个 **key** 都为 **true** 时，点击“确认”按钮才可以弹出“注册成功”对话框，否则弹出“请重新填写”对话框。

总体上，这个程序比较简单。

实验题 2 日历

本程序比较简单，我们先来看实验结果。

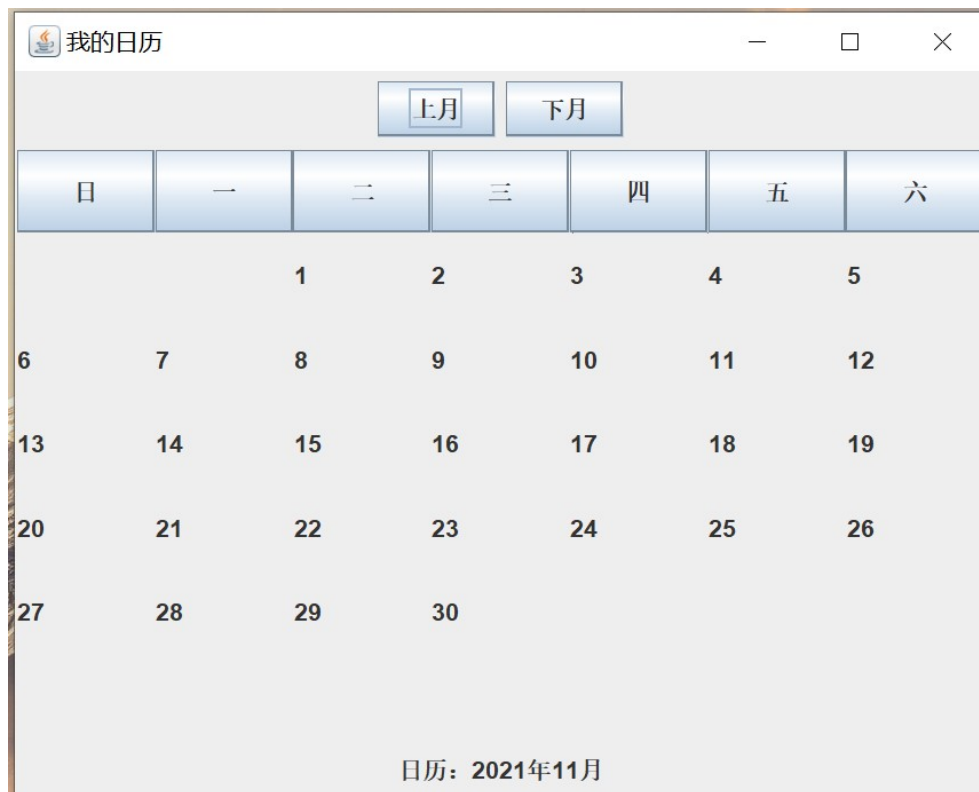


图 2.1 实验二结果

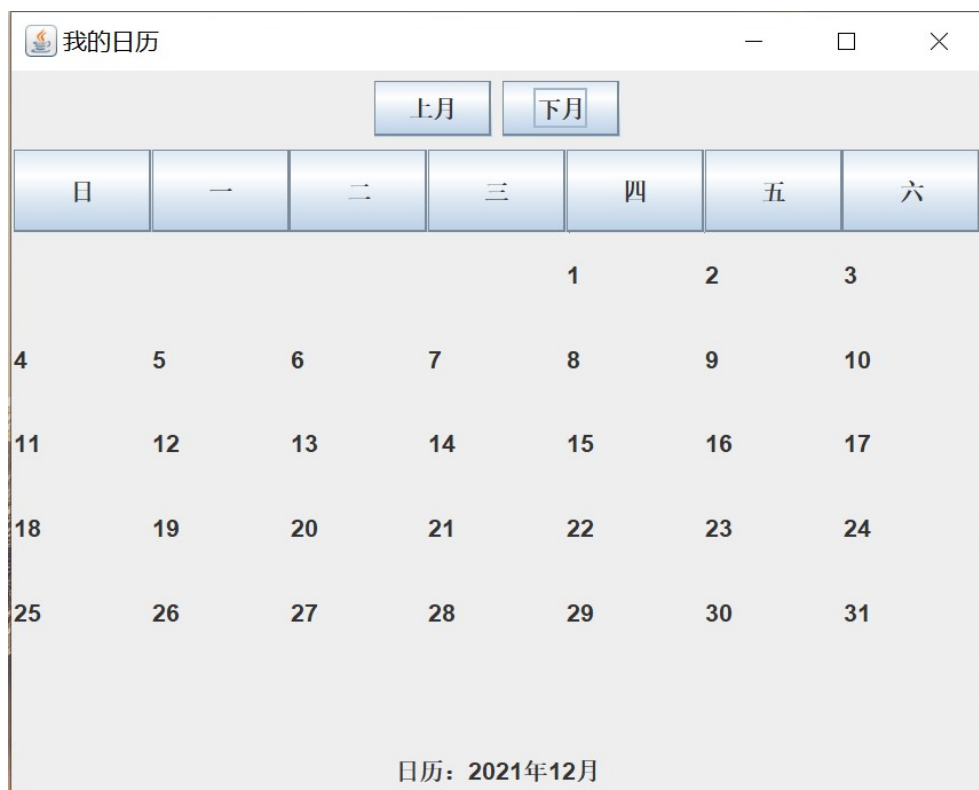


图 2.2 实验二结果

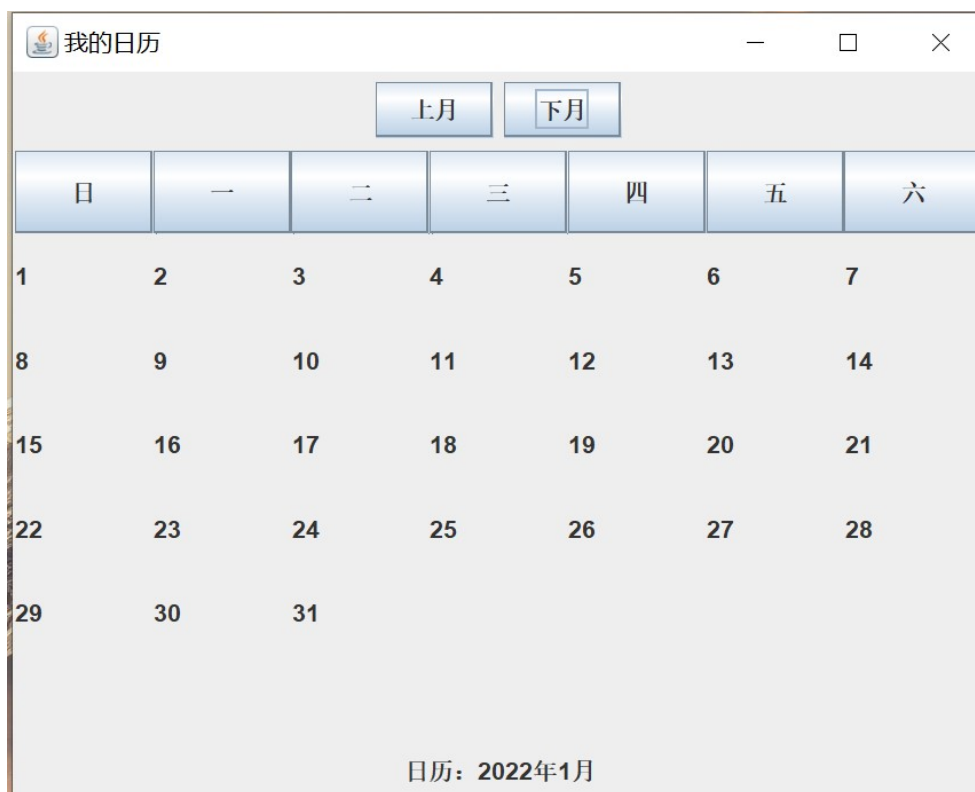


图 2.3 实验二结果

主界面采用边界布局。在北面，中间，南面分别添加 pCenter, pNorth, pSouth 容器。其中 pNorth, pSouth 采用流式布局，分别添加“上月”“下月”按钮和日历事件标签。pCenter 容器采用网格布局，设置 7*7 个网格，在最上面添加一行显示星期的按钮。在下面的格子中逐行添加日期的标签。这是界面的布局。下面看这个程序的类图。

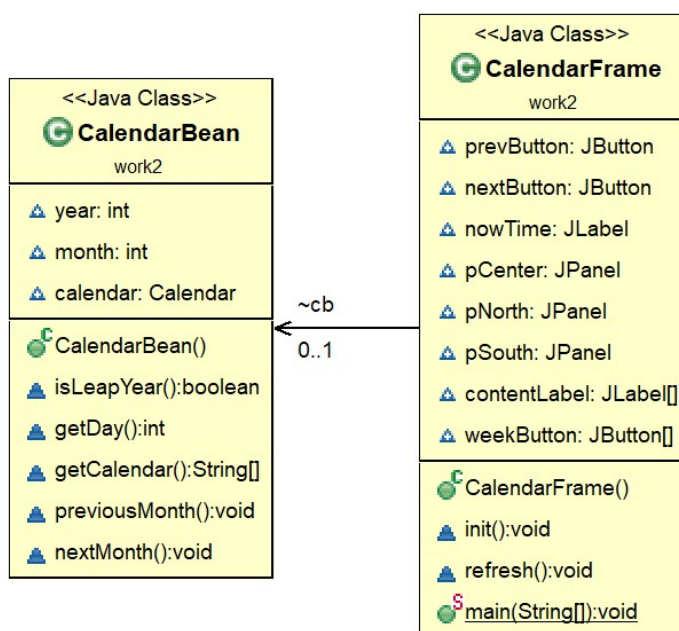


图 2.4 实验二 UML 类图

在类图中，我们看到这个程序尝试了 MVC 的结构。将所有与窗口有关的成员都放在 CalendarFrame 中，而所有与日历有关的成员都放在 CalendarBean 中。然而，事实上 CalendarBean 中的程序绝大多数在上一次控制台的日历程序中已经实现，这就为我们的开发大大节省了时间。在这个程序中，我们只需要修改部分的 previousMonth()和 nextMonth()方法即可。

此外，在 CalendarFrame 中添加相关的事件监听器，点击“上月”“下月”按钮，我们就调用 CalendarFrame 中的 refresh()方法，在 refresh()方法又调用 CalendarBean 中的 previousMonth()和 nextMonth()方法，最终将所有的表示日期的标签刷新一遍。

最后，经过测试，本程序可以正确执行。

实验总结

本次实验尚未解决的疑难：

本周时间过于紧张，只能保证完成基本的两个实验，还有剩下的三个实验没有完成。等时间充裕，再做完善。

心得体会：

本次实验收获颇丰。我掌握了 JAVA 图形界面开发的基本流程，如下图。



图 3.1 图形界面开发流程

了解了 JAVA 的各种布局：

FlowLayout（流布局管理器）

BorderLayout（边界布局管理器）

GridLayout（网格布局管理器）

CardLayout（卡片布局管理器）

BoxLayout

学会了各种组件的常用的使用方法。

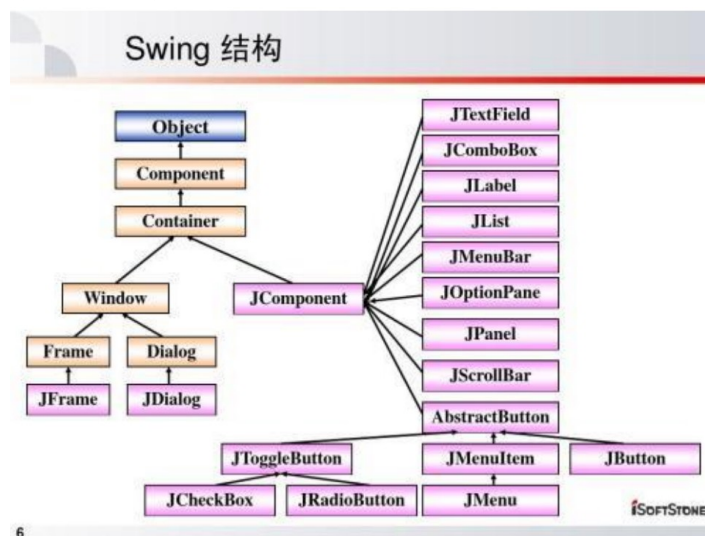


图 3.2 swing 组件

理解了 JAVA-GUI 的事件处理机制，学会了三种事件监听器的实现方法。
本次实验基本任务完成。