福建師範大學
FUJIAN NORMAL UNIVERSITY

Java语言

学　　期：2025-2026-1

教　　师：赵珊珊

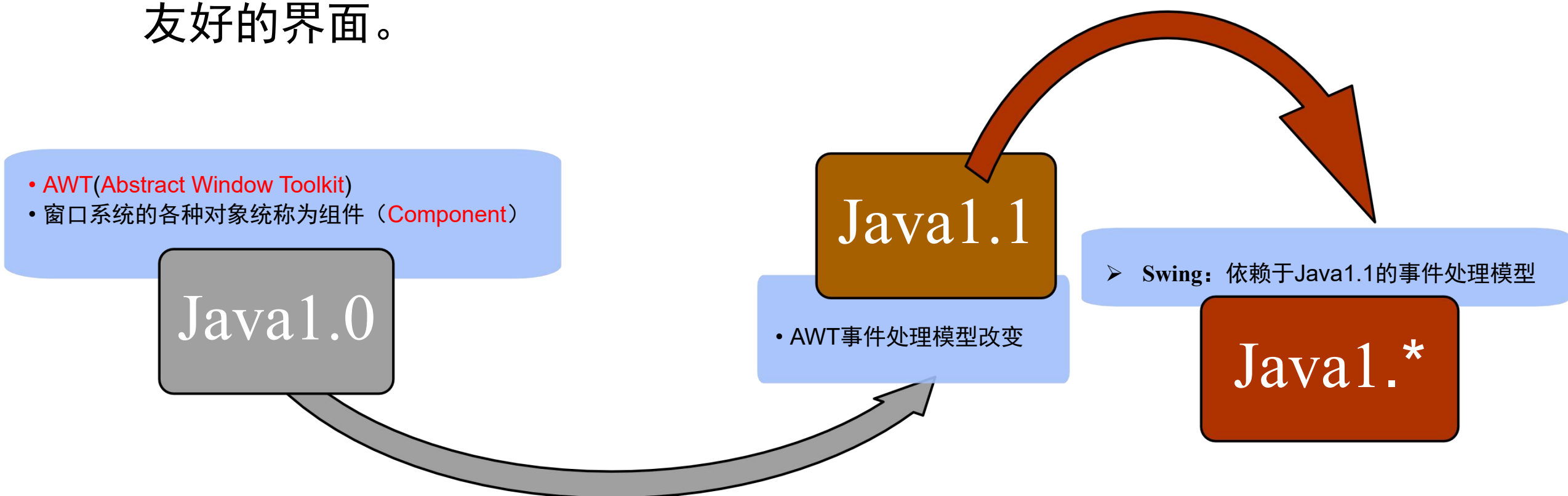
第10章GUI（图形用户界面）

第10章GUI（图形用户界面）

- 10.1Swing概述
- 10.2Swing容器
- 10.3Swing常用组件
- 10.4布局管理器
- 10.5事件处理

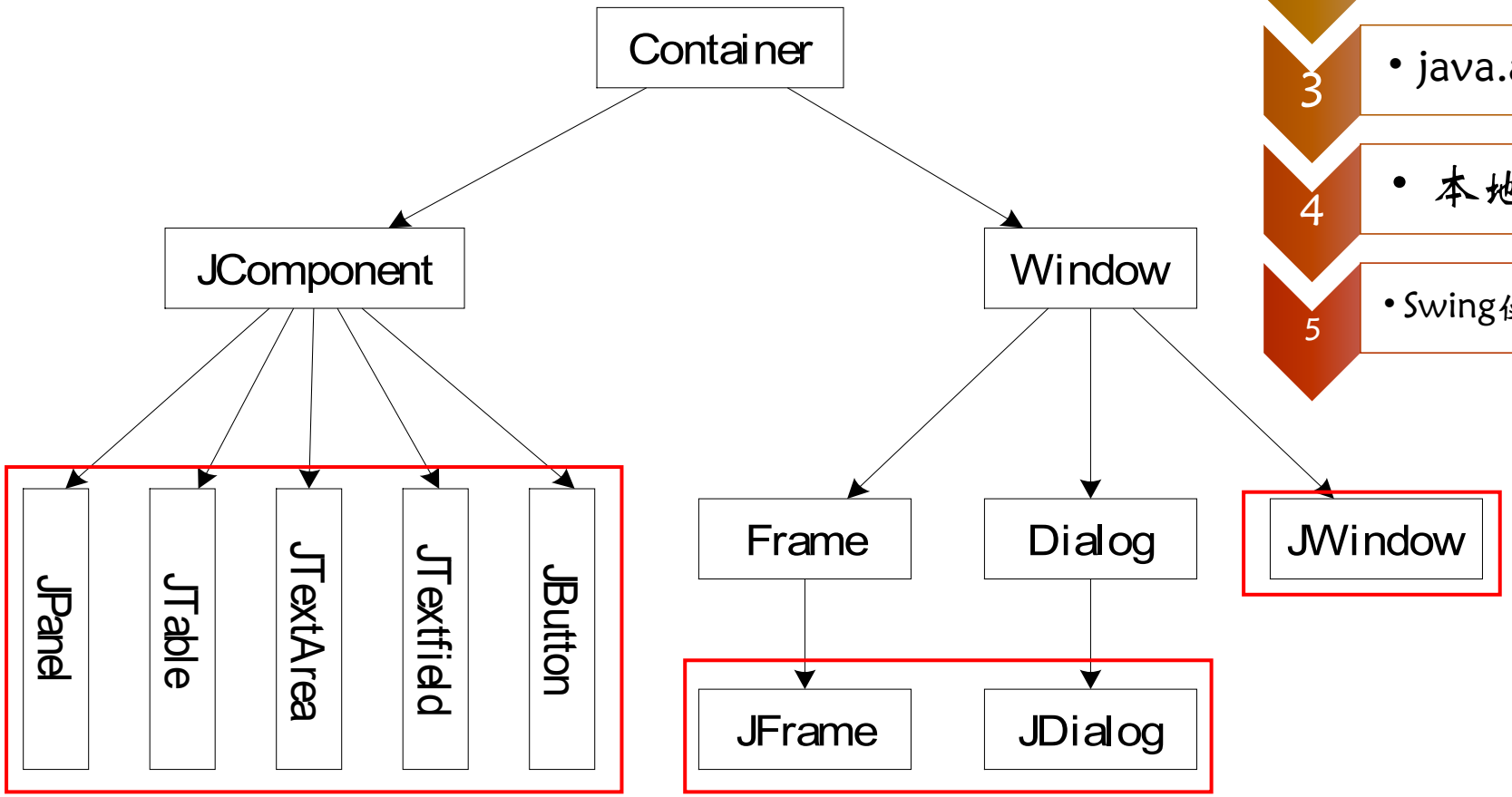
10.1Swing概述

- Swing是一种轻量级组件，它由Java语言开发，同时底层以AWT为基础，使跨平台应用程序可以使用任何可插拔的外观风格，并且Swing可以通过简洁的代码、灵活的功能和模块化组件来创建友好的界面。



10.1Swing概述

- Swing组件的主要类



- AWT → Swing

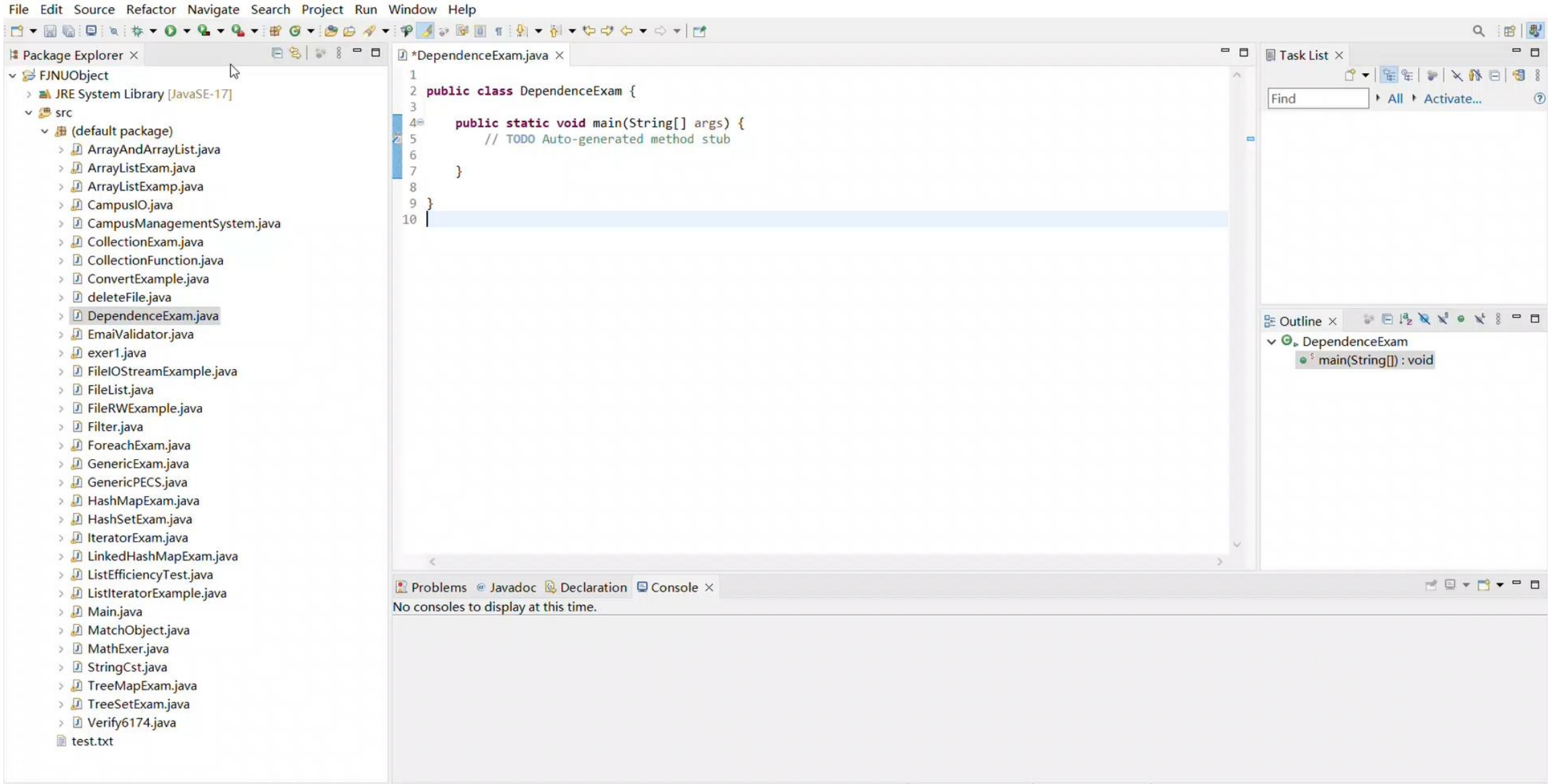
1. 重量组件→轻量组件
2. 原有组件→新增组件
3. java.awt组件名→javax.swing J+组件名
4. 本地GUI工具→纯Java组件跨平台
5. Swing使用AWT定义的事件、监视器接口、布局管理器

10.1Swing概述

- Swing组件的容器类比较

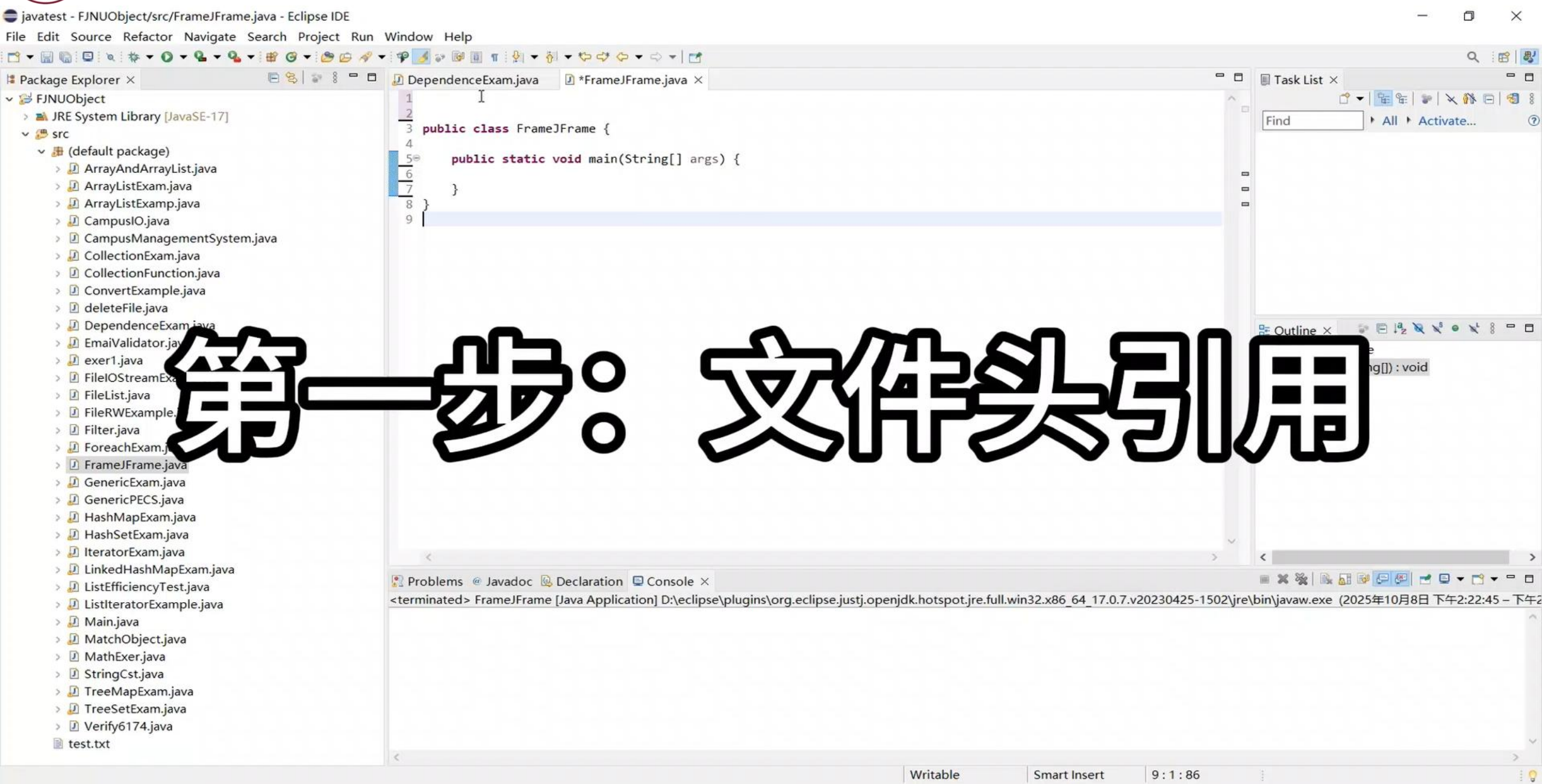
特性	JWindow 很少使用	JFrame +	JDialog
继承关系	顶层容器，是JFrame和JDialog的基类	继承自JWindow	继承自JWindow
边框与标题栏	无边框、无标题栏	有边框、有标题栏	有边框、有标题栏
菜单支持	不支持菜单栏	支持菜单栏（JMenuBar）	不支持菜单栏
调整大小	大小不可调整	大小可以调整	大小可以调整
模态支持	非模态	非模态	可设为模态或非模态
依赖关系	必须指定所有者（JFrame或 JDialog）	可独立存在	必须指定所有者（JFrame或 JDialog）
典型用途	启动画面（Splash Screen）、不规则形状窗口、浮动提示	应用程序的主窗口	临时交互窗口，如消息提示、用户输入

10.1Swing概述


File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FJNUObject
JRE System Library [JavaSE-17]
src
(default package)
ArrayAndArrayList.java
ArrayListExam.java
ArrayListExamp.java
CampusIO.java
CampusManagementSystem.java
CollectionExam.java
CollectionFunction.java
ConvertExample.java
deleteFile.java
DependenceExam.java
EmaiValidator.java
exer1.java
FileIOStreamExample.java
FileList.java
FileRWExample.java
Filter.java
ForeachExam.java
GenericExam.java
GenericPECS.java
HashMapExam.java
HashSetExam.java
IteratorExam.java
LinkedHashMapExam.java
ListEfficiencyTest.java
ListIteratorExample.java
Main.java
MatchObject.java
MathExer.java
StringCst.java
TreeMapExam.java
TreeSetExam.java
Verify6174.java
test.txt
*DependenceExam.java
1 public class DependenceExam {
2
3 public static void main(String[] args) {
4 // TODO Auto-generated method stub
5
6 }
7
8 }
9
10
Task List
Find
All
Activate...
Outline
DependenceExam
main(String[]) : void
Problems
Javadoc
Declaration
Console
No consoles to display at this time.

10.2Swing容器

- 10.2.1JFrame框架
- JFrame窗体是一个容器，它是Swing程序中各个组件的载体，可以将JFrame看作是承载这些Swing组件的容器。
- Frame窗体需要注册监听事件实现窗体关闭功能，Jframe调用setDefaultCloseOperation(int operation)方法即可。


javatest - FJNUObject/src/FrameJFrame.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FJNUObject
JRE System Library [JavaSE-17]
src
(default package)
ArrayAndArrayList.java
ArrayListExam.java
ArrayListExamp.java
CampusIO.java
CampusManagementSystem.java
CollectionExam.java
CollectionFunction.java
ConvertExample.java
deleteFile.java
DependenceExam.java
EmaiValidator.java
exer1.java
FileIOStreamExa
FileList.java
FileRWExample.
Filter.java
ForeachExam.j
FrameJFrame.java
GenericExam.java
GenericPECS.java
HashMapExam.java
HashSetExam.java
IteratorExam.java
LinkedHashMapExam.java
ListEfficiencyTest.java
ListIteratorExample.java
Main.java
MatchObject.java
MathExer.java
StringCst.java
TreeMapExam.java
TreeSetExam.java
Verify6174.java
test.txt
DependenceExam.java
*FrameJFrame.java
1
2
3 public class FrameJFrame {
4
5 public static void main(String[] args) {
6
7 }
8 }
9
Task List
Find
All
Activate...
Outline
ng[]) : void
Problems
Javadoc
Declaration
Console
<terminated> FrameJFrame [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (2025年10月8日 下午2:22:45 – 下午2
Writable
Smart Insert
9 : 1 : 86

第一步：文件头引用

10.2.2JDialog

- JDialog是Swing组件中的对话框，它继承了AWT组件中的Dialog类，其功能是从一个窗体中弹出另一个窗体，JDialog窗体与JFrame窗体类似，实质上是另一种类型的窗体。

构造方法	所有者窗口	标题	模态性	适用场景
JDialog()	无	无	非模态	快速创建简单的独立对话框
JDialog(Frame owner)	有 (Frame)	无	非模态	创建与主窗口关联但无标题的辅助窗口
JDialog(Frame owner, String title)	有 (Frame)	有	非模态	创建有关联、有明确提示标题的工具框
JDialog(Frame owner, String title, boolean modal)	有 (Frame)	有	可指定 (true/false)	创建需要用户立即响应的关键交互对

javatest - FJNUObject/src/JDialogExam.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- FJNUObject
  - JRE System Library [JavaSE-17]
  - src
    - (default package)
      - ArrayAndArrayList.java
      - ArrayListExam.java
      - ArrayListExamp.java
      - CampusIO.java
      - CampusManagementSystem.java
      - CollectionExam.java
      - CollectionFunction.java
      - ConvertExample.java
      - deleteFile.java
      - DependenceExam.java
      - EmaiValidator.java
      - exer1.java
      - FileIOStreamExample.java
      - FileList.java
      - FileRWExample.java
      - Filter.java
      - ForeachExam.java
      - FrameJFrame.java
      - GenericExam.java
      - GenericPECS.java
      - HashMapExam.java
      - HashSetExam.java
      - IteratorExam.java
      - JDialogExam.java
      - LinkedHashMapExam.java
      - ListEfficiencyTest.java
      - ListIteratorExample.java
      - Main.java
      - MatchObject.java
      - MathExer.java
      - StringCst.java
      - TreeMapExam.java
      - TreeSetExam.java
      - Verify6174.java
  - test.txt

DependenceExam.java | FrameJFrame.java | *JDialogExam.java

```
1 import java.awt.*;
2 import java.awt.event.*;
3 import javax.swing.*;
4 public class JDialogExam {
5     public static void main(String[] args) {
6
7     }
8 }
9
```

Task List

Find | All | Activate...

Outline

- JDialogExam
  - main(String[]) : void

Problems | Javadoc | Declaration | Console

JDialogExam [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (2025年10月8日 下午4:13:45) [pid: 15132]

Writable | Smart Insert | 3 : 22 : 67

10.3 Swing常用组件

- 10.3.1面板组件
- Swing中不仅有JFrame和JDialog这样的顶级窗口，还提供了一些中间容器，这些容器不能单独存在，只能放置在顶级窗口中，其中常用的两种分别为JPanel和JScrollPane，接下来分别介绍这两种容器。

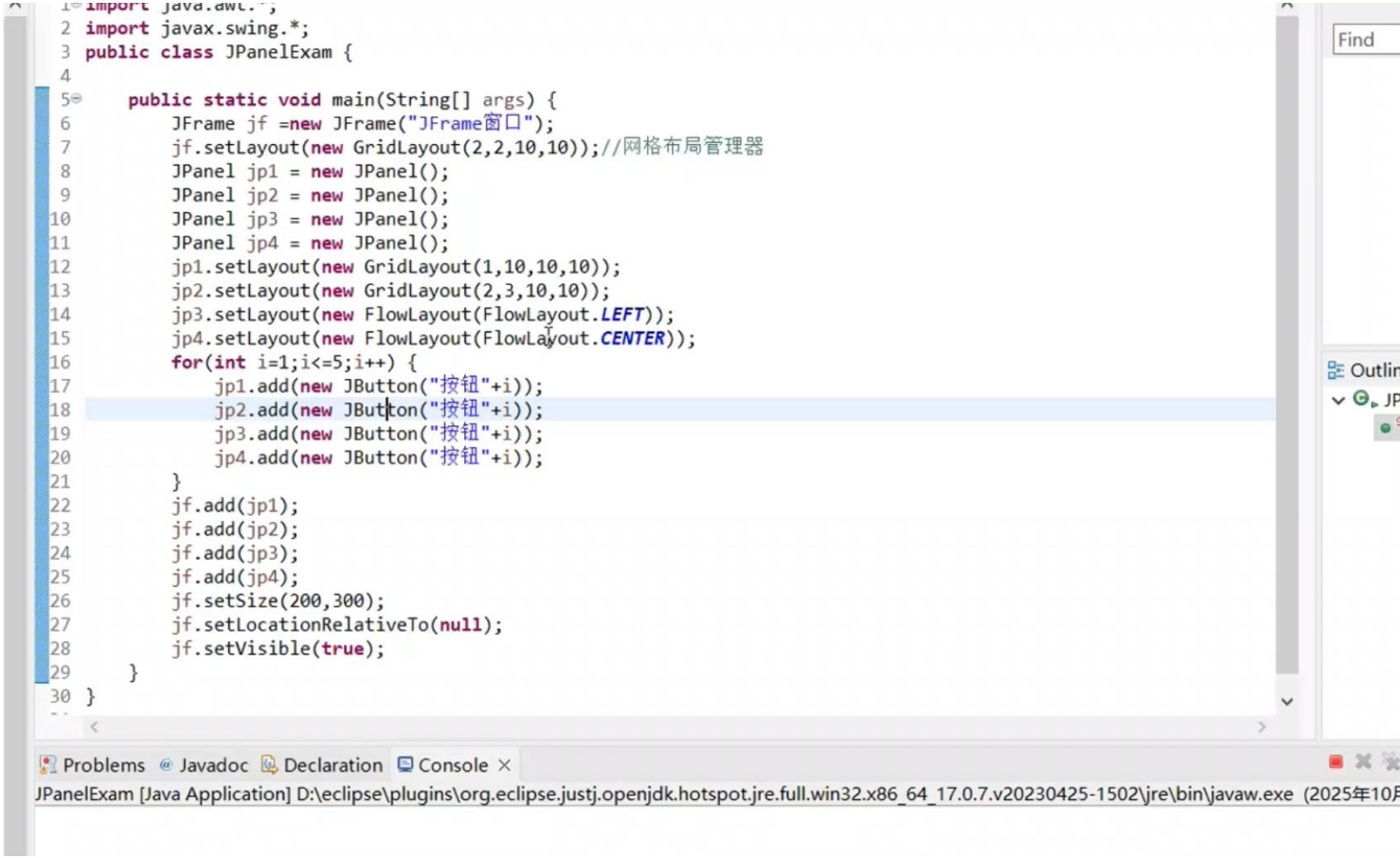
10.3 Swing常用组件

- 1.Jpanel
- JPanel面板组件是一个无边框，不能被移动、放大、缩小或者关闭的面板，它的默认布局管理器是FlowLayout
- 2.JScrollPane
- 在设置界面时，可能会遇到一个较小的容器窗体中显示较多内容的情况，这时可以使用JScrollPane面板，JScrollPane是一个带滚动条的面板容器，但是JScrollPane只能放置一个组件，并且不能使用布局管理器，如果需要在其中放置多个组件，需要将多个组件放置在JPanel面板容器上，然后将JPanel面板作为一个整体组件添加到JScrollPane面板中。

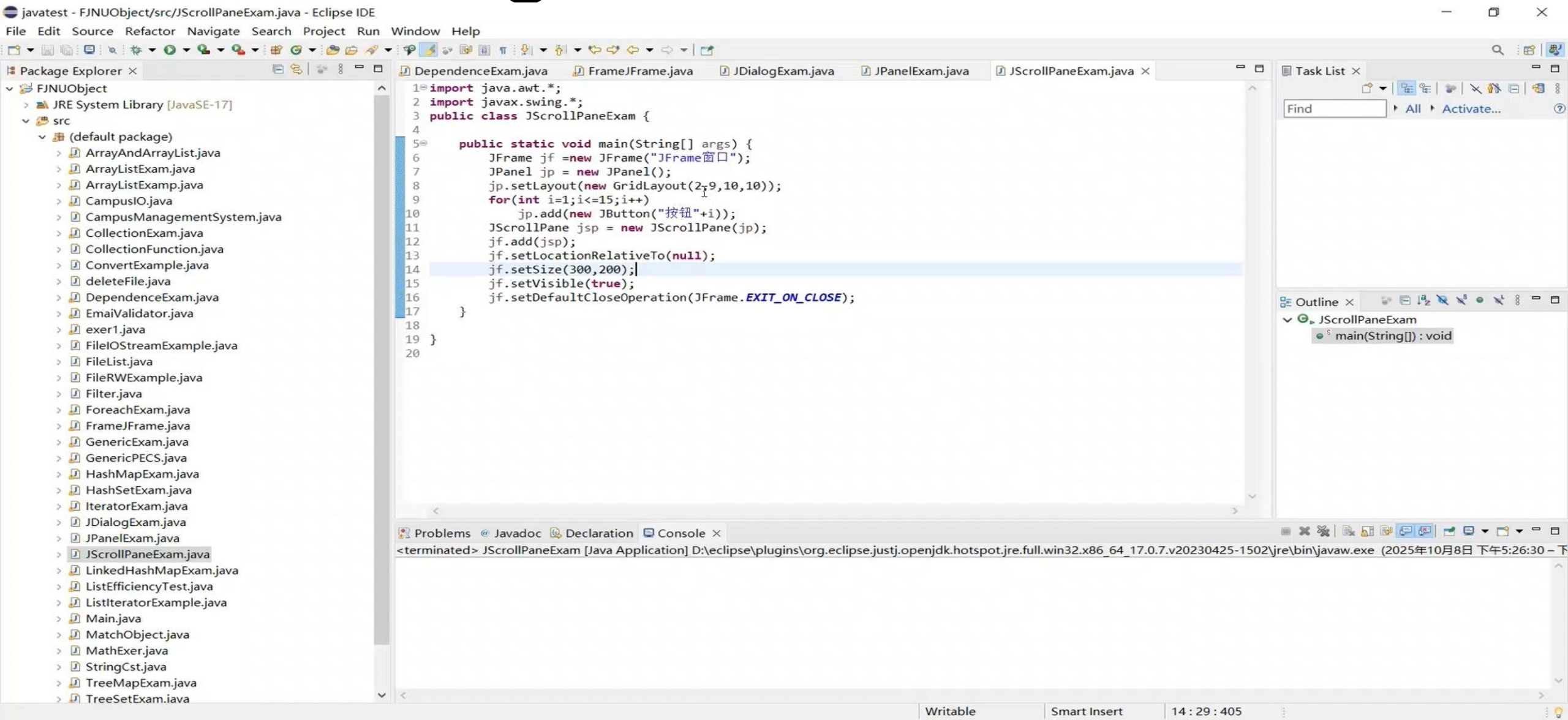
10.3 Swing常用组件

```java
import java.awt.*;
import javax.swing.*;
public class JPanelExam {

    public static void main(String[] args) {
        JFrame jf =new JFrame("JFrame窗口");
        jf.setLayout(new GridLayout(2,2,10,10));//网格布局管理器
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();
        JPanel jp3 = new JPanel();
        JPanel jp4 = new JPanel();
        jp1.setLayout(new GridLayout(1,10,10,10));
        jp2.setLayout(new GridLayout(2,3,10,10));
        jp3.setLayout(new FlowLayout(FlowLayout.LEFT));
        jp4.setLayout(new FlowLayout(FlowLayout.CENTER));
        for(int i=1;i<=5;i++) {
            jp1.add(new JButton("按钮"+i));
            jp2.add(new JButton("按钮"+i));
            jp3.add(new JButton("按钮"+i));
            jp4.add(new JButton("按钮"+i));
        }
        jf.add(jp1);
        jf.add(jp2);
        jf.add(jp3);
        jf.add(jp4);
        jf.setSize(200,300);
        jf.setLocationRelativeTo(null);
        jf.setVisible(true);
    }
}
```



10.3 Swing常用组件



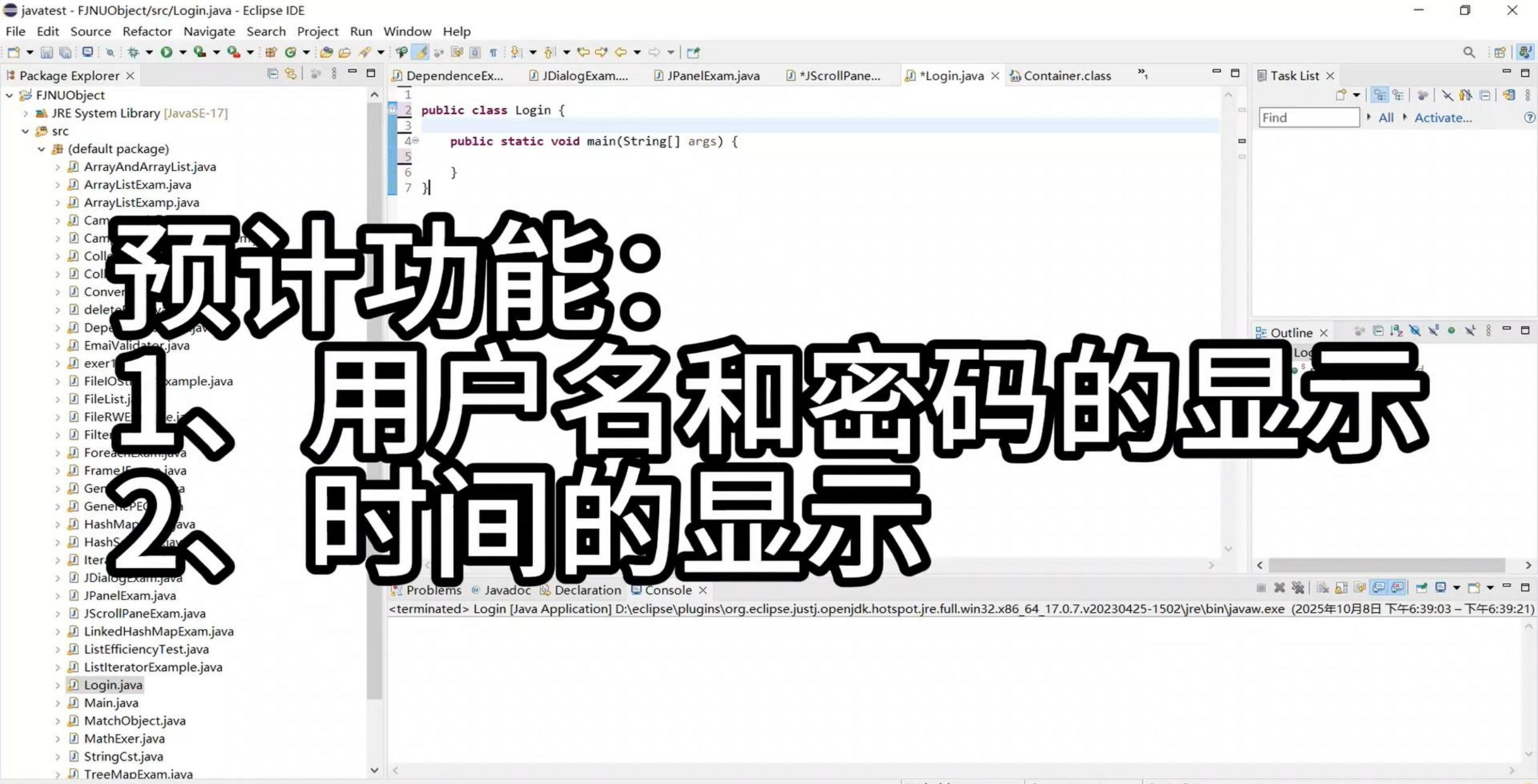
10.3 Swing常用组件

- 10.3.2文本组件
- 文本组件用于接收用户输入的信息或向用户展示信息，其中包括文本框（JTextField）、密码框（JPasswordField）和文本域（JTextArea），它们都有一个共同的父类JTextComponent。

JTextComponent方法名	功能说明	适用场景
String getText()	返回文本组件中包含的所有文本内容。	获取用户输入或显示的文本。
void setText(String text)	将文本组件的内容设置为指定的字符串。	初始化显示文本或清空/重置内容。
String getSelectedText()void selectAll()void select(int selStart, int selEnd)**	返回当前选中的文本。选中组件中的所有内容。选中指定开始和结束位置之间的文本。	获取或操作用户选中的文本部分。
void setEditable(boolean b)	设置文本组件是否可被用户编辑。	创建只读的显示区域。
int getCaretPosition()void setCaretPosition(int position)	获取/设置文本插入符（光标）的当前位置。	精确控制光标位置或进行文本插入。
void replaceSelection(String content)	用给定的字符串替换当前选中的内容。	快速替换已选中的文本块。

10.3 Swing常用组件

- 10.3.2文本组件
- 1.文本框（JTextField）
- JTextField称为文本框，它只能接受单行文本的输入
- 2.密码框（JPasswordField）
- 密码框（JPasswordField）与文本框的定义和用法类似，唯一不同的就是密码框将用户输入的字符串以某种符号进行加密。
- 3.文本域（JTextArea）
- JTextArea称为文本域，它能接受多行文本的输入，使用JTextArea构造方法创建对象时可以设定区域的行数、列数



预计功能：
1、用户名和密码的显示
2、时间的显示

10.3 Swing常用组件

- 10.3.3标签组件
- 在Swing组件中，除了有用与输入功能的文本组件外，还提供了仅供展示的标签组件，标签组件也是Swing中很常用的组件。Swing中的标签组件主要用到的是JLabel，JLabel组件可以显示文本、图像。还可以设置标签内容的水平和垂直的对其方式。

10.3 Swing常用组件

- 10.3.4按钮组件
- 按钮组件在Swing中是较为常见的组件，它用于触发特定动作，其中包含JButton按钮、JRadioButton按钮和JCheckBox按钮等，它们都继承自AbstractButton抽象类。

10.3 Swing常用组件

- 1. JButton按钮
- Swing中的JButton按钮通常用于确定、保存、取消等操作


```java
import javax.swing.*;
import java.awt.event.*;

public class ClickCounter {
    static int clickCount=0;
    public static void main(String[] args) {

        JFrame frame = new JFrame("点击计数器");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 150);
        frame.setLayout(new java.awt.FlowLayout());

        JLabel countLabel = new JLabel("点击次数: 0");
        JButton button = new JButton("点我！");

        // 为按钮添加事件监听器
        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // 每次按钮被点击时，计数器加1并更新标签
                clickCount++;
                countLabel.setText("点击次数: " + clickCount);
            }
        });

        frame.add(button);
        frame.add(countLabel);
        frame.setVisible(true);
    }
}
```

引用类
计数
初始化窗体
初始化组件
实现点击数累加

10.3 Swing常用组件

- 2. JRadioButton按钮
- JRadioButton按钮被称为单选按钮组件，一般将多个单选按钮放置在按钮组中，使这些单选按钮表现出某种功能，当用户选中某个单选按钮后，按钮组中其他按钮将被自动取消。
- 但是JRadioButton组件本身并不具有这种功能，因此想实现JRadioButton按钮之间的互斥，需要使用ButtonGroup类。ButtonGroup是一个不可见的组件，不需要将其添加到容器中显示，只是在逻辑上表示一个单选按钮组。***将多个JRadioButton按钮添加到同一个单选按钮组中就能实现JRadioButton按钮的单选功能***。

10.3 Swing常用组件

- 3. JCheckBox按钮
- JCheckBox按钮被称为复选框按钮组件，与单选按钮不同的是，复选框可以进行多选设置，每一个复选框都提供“选中”与“不选中”两种状态。复选框由JCheckBox类的对象表示，它同样继承于AbstractButton抽象类

10.3 Swing常用组件

未将JRadiobutton放进ButtonGroup时，无法呈现单选特征

```
1  import java.awt.*;
2  import javax.swing.*;
3  public class RadioCheck {
4
5      public static void main(String[] args) {
6          JFrame jf = new JFrame("单选&复选按钮");
7          jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8          jf.setSize(300, 150);
9          jf.setLayout(new GridLayout(3,2,10,10));
10         for(int i=1;i<=3;i++) {
11             jf.add(new JRadioButton("单选按钮"+i));
12             jf.add(new JCheckBox("复选按钮"+i));
13         }
14         jf.setLocationRelativeTo(null);
15         jf.setVisible(true);
16
17     }
18
19 }
20
```

10.3 Swing常用组件

将JRadiobutton放进ButtonGroup进行逻辑分组后实现单选功能：

```
1  import java.awt.*;
2  import javax.swing.*;
3  public class RadioCheck {
4
5      public static void main(String[] args) {
6          JFrame jf = new JFrame("单选&复选按钮");
7          jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8          jf.setSize(300, 150);
9          jf.setLayout(new GridLayout(3,2,10,10));
10         ButtonGroup jbg = new ButtonGroup();
11         for(int i=1;i<=3;i++) {
12             JRadioButton jrb = new JRadioButton("单选按钮"+i);
13             jbg.add(jrb);// 先将按钮添加到 ButtonGroup 进行逻辑分组
14             jf.add(jrb); // 再将按钮添加到 JFrame显示
15             jf.add(new JCheckBox("复选按钮"+i));
16         }
17
18         jf.setLocationRelativeTo(null);
19         jf.setVisible(true);
20
21     }
22
23 }
```

10.3 Swing常用组件

- 10.3.5下拉框组件
- JComboBox组件被称为下拉框或者组合框组件，它将所有选项叠加在一起，默认显示的是第一个添加的选项。当用户单击下拉框时，会出现下拉式的选项列表，用户可以从中选择其中一项并显示。
- JComboBox下拉框组件分为可编辑和不可编辑两种形式，对于不可编辑的下拉框，用户只能选择现有的选项列表。对于可编辑的下拉框，用户既可以选择现有的选项列表，也可以自己输入新的内容。需要注意的是，自己输入的内容只能作为当前显示，并不会添加到下拉框的选项列表中。

10.3 Swing常用组件

例：用JComboBox显示笃行楼101-106的剩余空座位数

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.HashMap;
import java.util.Map;
public class JComboBoxExam extends JFrame {
    // 使用Map来存储教室编号和座位数的对应关系，模拟数据
    private static final Map<String, Integer> classroomSeats = new HashMap<>();
    static {
        classroomSeats.put("101", 12);
        classroomSeats.put("102", 10);
        classroomSeats.put("103", 0);
        classroomSeats.put("104", 28);
        classroomSeats.put("105", 40);
        classroomSeats.put("106", 20);
    }
    private JComboBox<String> classroomComboBox;
    private JLabel seatCountLabel;

    public JComboBoxExam() {
        setTitle("笃行楼1楼教室座位查询");
        setSize(400, 150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null); // 窗口居中
        setVisible(true);
        // 设置布局
        setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));

        // 创建教室选择标签和下拉框
        add(new JLabel("选择教室:"));
```

10.3 Swing常用组件

- 10.3.6菜单选项
- 1.下拉式菜单
- 创建下拉式菜单需要使用三个组件：JMenuBar（菜单栏）、JMenu（菜单）和JMenuItem（菜单项）。

10.3 Swing常用组件

- 10.3.6菜单选项
- （1）JMenuBar。JMenuBar表示一个水平的菜单栏，它用来管理一组菜单，不参与同用户的交互式操作。
- （2）JMenu。JMenu表示一个菜单，它用来整合管理菜单项，菜单可以使单一层次的结构，也可以使多层次的结构。
- （3）JMenuItem。JMenuItem表示一个菜单项，它是菜单系统中最基本的组件。在创建JMenuItem菜单项时，通常会使用JMenuItem(String text)这个构造方法为菜单项指定文本内容。

10.3 Swing常用组件

- 10.3.6菜单选项
- 2.弹出式菜单
- 前面讲解了下拉式菜单，还有一种菜单是弹出式菜单，如果要在Java中实现此菜单，可以使用JPopupMenu菜单组件，JPopupMenu弹出式菜单和下拉式菜单一样都是通过add()方法添加JMenuItem菜单项，但它默认是不可见的，如果要显示出来，则需调用它的show(Component invoker,int x,int y)方法，该方法中的参数invoker表示JPopupMenu菜单显示位置的参数组件，x和y表示invoker组件的坐标，显示的是JPopupMenu菜单的左上角坐标。

10.3 Swing常用组件

设计一个下拉菜单，有新建、打开，保存、退出4个子菜单，点击退出时弹出对话框询问是否要退出，确定则退出；

点击鼠标右键时会弹出“退出”菜单。

```java
import javax.swing.*;
import java.awt.event.*;

public class MenuExam extends JFrame {

    // 声明菜单组件
    private JMenuBar menuBar;
    private JMenu fileMenu;
    private JMenuItem newItem, openItem, saveItem, exitItem;
    private JPopupMenu popupMenu;
    private JMenuItem popupExitItem;

    public MenuExam() {
        // 设置窗口基本属性
        setTitle("简易菜单演示");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null); // 窗口居中

        // 创建菜单组件
        createMenuBar();
        createPopupMenu();

        setVisible(true);
    }

    private void createMenuBar() {
        // 创建菜单栏
        menuBar = new JMenuBar();

```

10.3 Swing常用组件

- 10.3.7创建Tree
- 树也是图形化用户界面中使用非常广泛的GUI组件，例如打开Windows资源管理器时就会看到目录树，在Swing中使用JTree对象来代表一棵树，JTree树中节点可以使用TreePath标识，该对象封装了当前节点及其所有的父节点。当一个节点具有子节点时，该节点具有展开和折叠两种状态。如果希望创建一棵树，可使用JTree类。

10.3 Swing常用组件

```
1  import javax.swing.*;
2  import javax.swing.tree.*;
3
4  public class CampusFacilitiesTree {
5
6      public static void main(String[] args) {
7          new CampusFacilitiesTree();
8      }
9
10     CampusFacilitiesTree() {
11         JFrame frame = new JFrame("校园设施一览");
12         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13         frame.setSize(300, 300); // 设置窗口大小
14
15         DefaultMutableTreeNode root = new DefaultMutableTreeNode("校园设施");
16
17         // 创建"教学楼"分支节点
18         DefaultMutableTreeNode teachingBuildings = new DefaultMutableTreeNode("教学楼");
19         teachingBuildings.add(new DefaultMutableTreeNode("知明楼"));
20         teachingBuildings.add(new DefaultMutableTreeNode("笃行楼"));
21         teachingBuildings.add(new DefaultMutableTreeNode("立诚楼"));
22         teachingBuildings.add(new DefaultMutableTreeNode("致广楼"));
23
24         // 创建"餐厅"分支节点
25         DefaultMutableTreeNode canteens = new DefaultMutableTreeNode("餐厅");
26         canteens.add(new DefaultMutableTreeNode("花香园"));
27         canteens.add(new DefaultMutableTreeNode("桃李园"));
28         canteens.add(new DefaultMutableTreeNode("桃苑"));
29
30         root.add(teachingBuildings);
```

10.4布局管理器

- 一个容器被创建后，它们有相应的默认布局管理器。JWindow、JFrame和JDialog的默认布局管理器是BorderLayout，JPanel和JApplet的默认布局管理器是FlowLayout。在java.awt包中提供了五种布局管理器，分别是FlowLayout（流式布局管理器）、BorderLayout（边界布局管理器）、GridLayout（网格布局管理器）、GridBagLayout（网格包布局管理器）和CardLayout（卡片布局管理器）。

10.4布局管理器

- 10.4.1 FlowLayout（流式布局管理器）
- FlowLayout（流式布局管理器）是最简单的布局管理器，在这种布局下，容器会将组件安装添加顺序从左到右放置，当达到容器的边界时，会自动将组件放在下一行的开始位置。这些组件可以按从左对齐、居中对齐或右对齐的方式排列。

10.4布局管理器

- 10.4.2 BorderLayout（边界布局管理器）
- BorderLayout（边界布局管理器）将一个窗体的版面分成东、西、南、北、中5个区域，可以直接将需要的组件放到这5个区域中。

10.4布局管理器

- 10.4.3 GridLayout（网格布局管理器）
- GridLayout布局管理器是以表格形式进行管理的，在使用此布局管理器时必须设置显示的行数和列数

10.4布局管理器

- 10.4.4 GridBagLayout（网格包布局管理器）
- GridBagLayout类是在GridLayout类基础上提供的更为复杂的布局管理器。与GridLayout布局管理器不同的是，GridBagLayout类允许容器中各个组件的大小不相同，还允许单个组件所在的显示区域占多个网格。

10.4布局管理器

- 10.4.5 CardLayout（卡片布局管理器）
- CardLayout布局管理器是将一些组件彼此重叠地进行布局，像一张张卡片叠放在一起一样，这样每次只会展现一个界面

10.4布局管理器

- 10.4.6取消布局管理器
- 容器被创建后，都会有一个默认的布局管理器。例如JWindow、JFrame和JDialog的默认布局管理器是BorderLayout，JPanel和JApplet的默认布局管理器是FlowLayout。如果不希望通过布局管理器来对容器进行布局，也可以调用容器的setLayout(null)方法，将布局管理器取消。

10.4布局管理器

```
1  import java.awt.*;
2  import javax.swing.*;
3  public class FlowLayoutDemo{
4      public static void main(String[] args) {
5          JFrame frame = new JFrame("Frame窗体");
6          frame.setLayout(new FlowLayout());
7          JButton button = null;
8          for (int i = 0; i < 8; i++) {
9              button = new JButton("按钮 - " + i);
10             frame.add(button);
11         }
12         frame.setSize(200, 150);
13         frame.setLocation(500, 200);
14         frame.setVisible(true);
15     }
16 }
17
```

10.5事件处理

- 10.5.1事件处理机制
- Swing组件中的事件处理专门用于相应用户的操作，例如，相应用户的单机鼠标、按下键盘等操作。在Swing事件处理的过程中，主要涉及三类对象：
  - （1）事件源（Event Source）：事件发生的场所，通常就是产生事件的组件，例如窗口、按钮、菜单等。
  - （2）事件对象（Event）:封装了GUI组件上发生的特定事件（通常就是用户的一次操作）。
  - （3）监听器（Listener）：负责监听事件源上发生的事件，并对各种事物做出相应处理的对象（对象中包含事件处理器）。

10.5事件处理

- 10.5.2Swing常用事件处理
- 在Swing中，提供了丰富的事件，这些事件大致可以分为窗体事件、鼠标事件、键盘事件、动作事件

- 窗体事件：

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4  public class WindowEventDemo {
5      public static void main(String[] args) {
6          JFrame frame = new JFrame("Frame窗体");
7          frame.setSize(300, 200);
8          frame.setLocation(500, 200);
9          frame.setVisible(true);
10         frame.addWindowListener(new WindowListener() {
11             public void windowOpened(WindowEvent e) {
12                 System.out.println("windowOpened-->窗口被打开");
13             }
14             public void windowIconified(WindowEvent e) {
15                 System.out.println("windowIconified-->窗口最小化");
16             }
17             public void windowDeiconified(WindowEvent e) {
18                 System.out.println("windowDeiconified-->窗口从最小化恢复");
19             }
20             public void windowDeactivated(WindowEvent e) {
21                 System.out.println("windowDeactivated-->取消窗口选中");
22             }
23             public void windowClosing(WindowEvent e) {
24                 System.out.println("windowClosing-->窗口正在关闭");
25                 ((Window) e.getComponent()).dispose();
26             }
27             public void windowClosed(WindowEvent e) {
28                 System.out.println("windowClosed-->窗口关闭");
29             }
30             public void windowActivated(WindowEvent e) {
```

Problems @ Javadoc Declaration Console ×

```
WindowEventDemo [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin
windowActivated-->窗口被选中
windowOpened-->窗口被打开
windowIconified-->窗口最小化
windowDeactivated-->取消窗口选中
```

10.5事件处理

- 鼠标事件：

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class MouthEventDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Frame窗口");
        JButton bt = new JButton("按钮");
        frame.add(bt);
        frame.setSize(300, 200);
        frame.setLocation(500, 200);
        frame.setVisible(true);
        bt.addMouseListener(new MouseListener() {
            public void mouseReleased(MouseEvent e) {
                System.out.println("mouseReleased-->鼠标松开");
            }
            public void mousePressed(MouseEvent e) {
                System.out.println("mousePressed-->鼠标按下");
            }
            public void mouseExited(MouseEvent e) {
                System.out.println("mouseExited-->鼠标离开组件");
            }
            public void mouseEntered(MouseEvent e) {
                System.out.println("mouseEntered-->鼠标进入组件");
            }
            public void mouseClicked(MouseEvent e) {
                int i = e.getButton();
                if (i == MouseEvent.BUTTON1) {
                    System.out.println("mouseClicked-->鼠标左键点击"
                            );
                }else if(i==MouseEvent.BUTTON3){
```

Problems Javadoc Declaration Console ×

```
MouthEventDemo [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\
mouseEntered-->鼠标进入组件
mouseExited-->鼠标离开组件
mouseEntered-->鼠标进入组件
mouseExited-->鼠标离开组件
mouseEntered-->鼠标进入组件
mouseExited-->鼠标离开组件
mouseEntered-->鼠标进入组件
mouseExited-->鼠标离开组件
```

10.5事件处理

- 键盘事件：

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class KeyEventDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Frame窗口");
        TextField tf = new TextField(10);
        frame.add(tf);
        frame.setSize(300, 200);
        frame.setLocation(500, 200);
        frame.setVisible(true);
        tf.addKeyListener(new KeyAdapter() {
            public void keyPressed(KeyEvent e) {
                System.out.println("keyPressed-->键盘"
                        + KeyEvent.getKeyText(e.getKeyCode()) + "键按下");
            }
            public void keyReleased(KeyEvent e) {
                System.out.println("keyReleased-->键盘"
                        + KeyEvent.getKeyText(e.getKeyCode()) + "键松开");
            }
            public void keyTyped(KeyEvent e) {
                System.out.println("keyTyped-->键盘输入的内容是："
                        + e.getKeyChar());
            }
        });
    }

}

```

Problems @ Javadoc Declaration Console ×

```
MouthEventDemo [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\
mouseEntered-->鼠标进入组件
mouseExited-->鼠标离开组件
```

10.5事件处理

- 动作事件：

```java
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ActionListenerDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("ActionListener 示例");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 150);

        JButton button = new JButton("点击我");

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // 这里是点击按钮后要执行的逻辑
                System.out.println("按钮被点击了！");
                JOptionPane.showMessageDialog(frame, "你好！这是一个动作事件。");
            }
        });

        frame.getContentPane().add(button);
        frame.setVisible(true);
    }
}
```

Problems Javadoc Declaration Console ×

<terminated> ActionListenerDemo [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v202304

```
按钮被点击了！
按钮被点击了！
按钮被点击了！
```