



## 关于实验课

---

1. 使用腾讯课堂上课，如遇到技术故障将改用腾讯会议；
2. 为方便考勤，请同学们将昵称改成“学号-真实姓名”；
3. 上课不定时发起签到，请同学们不要迟到早退。



哈爾濱工業大學(深圳)  
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

規格嚴格



功夫到家

1920 — 2017

# 面向对象的软件构造导论

## 实验五：Swing和多线程

2022春

哈尔滨工业大学（深圳）



# 本学期实验总体安排

实验项目	一	二	三	四	五	六
学时数	2	2	2	4	2	4
实验内容	飞机大战 功能分析	单例模式 工厂模式	Junit 单元测试	策略模式 数据访问 对象模式	Swing 多线程	模板模式 观察者模式
分数	4	6	4	6	6	14
提交内容	UML类图、 代码	UML类图、 代码	单元测试 代码 测试报告	UML类图、 代码	代码	项目代码、 实验报告

实验课程共**16**个学时，**6**个实验项目，总成绩为**40**分。



# 目录

---

01

实验目的

02

实验任务

03

实验步骤

04

实验要求

05

作业提交



# 实验目的

---

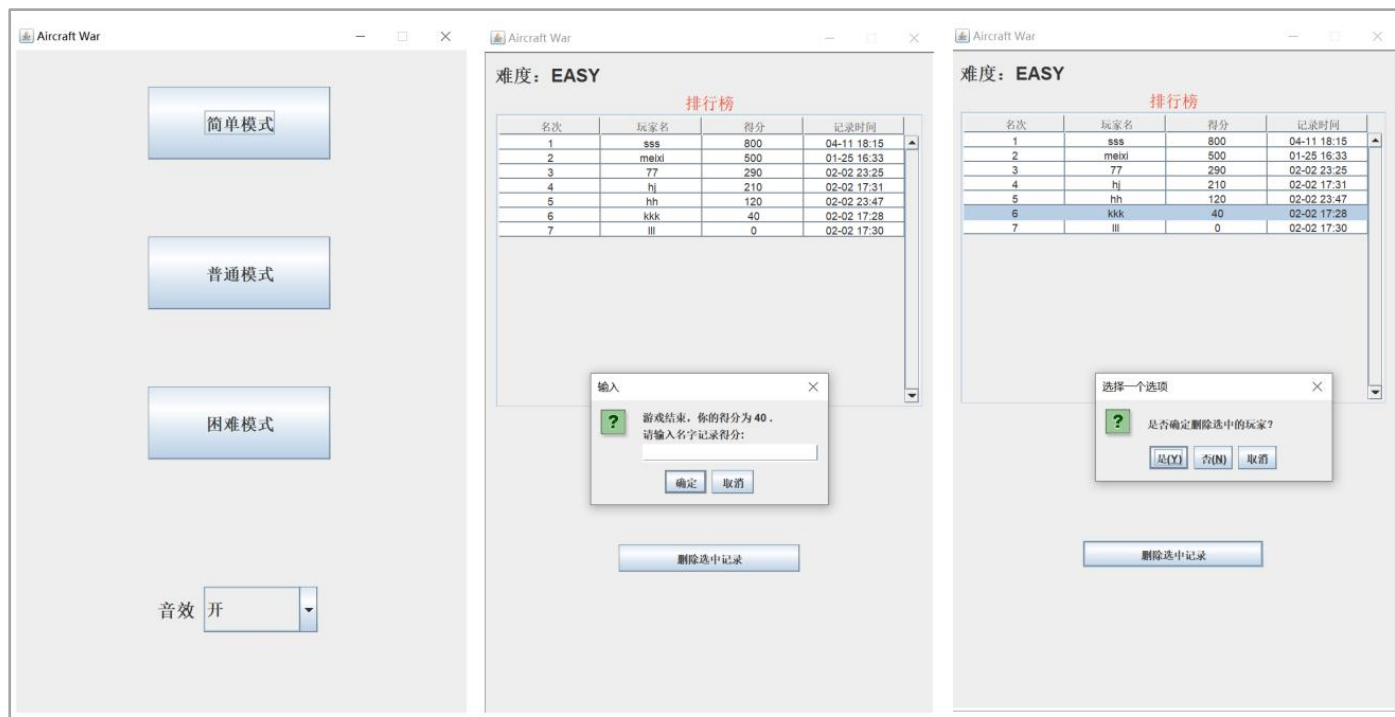
- 掌握Java图形界面程序设计的基本方法，熟悉Java Swing中的容器、常用组件和布局管理器的使用，了解Java事件处理机制，掌握事件处理机制的基本用法；
- 理解Java多线程的概念和生命周期，掌握多线程的实现方法。



# 实验任务

1. 使用Java Swing类库完成游戏中的**难度选择**、**音效开关**界面。
2. 使用Java Swing类库完成**得分排行榜**界面，要求可记录玩家该局得分，并可删除玩家历史得分。

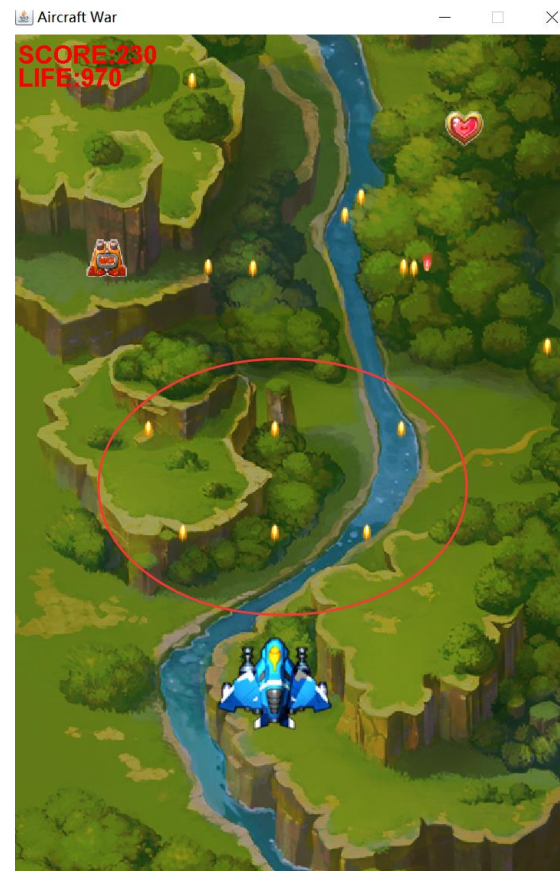
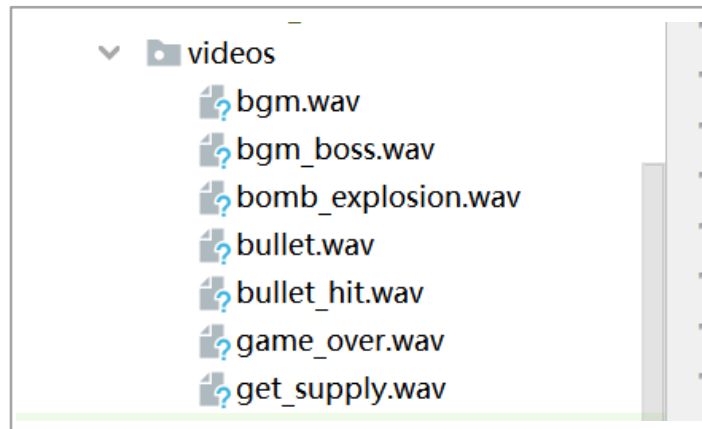
界面与下图接近即可：





# 实验五任务

3. 使用Runnable接口实现多线程，完善火力道具功能。
4. 继承Thread类实现多线程，完成游戏背景音乐、子弹击中敌机、炸弹爆炸、道具生效、Boss机出场、游戏结束时的音效控制。





## 1 Java Swing

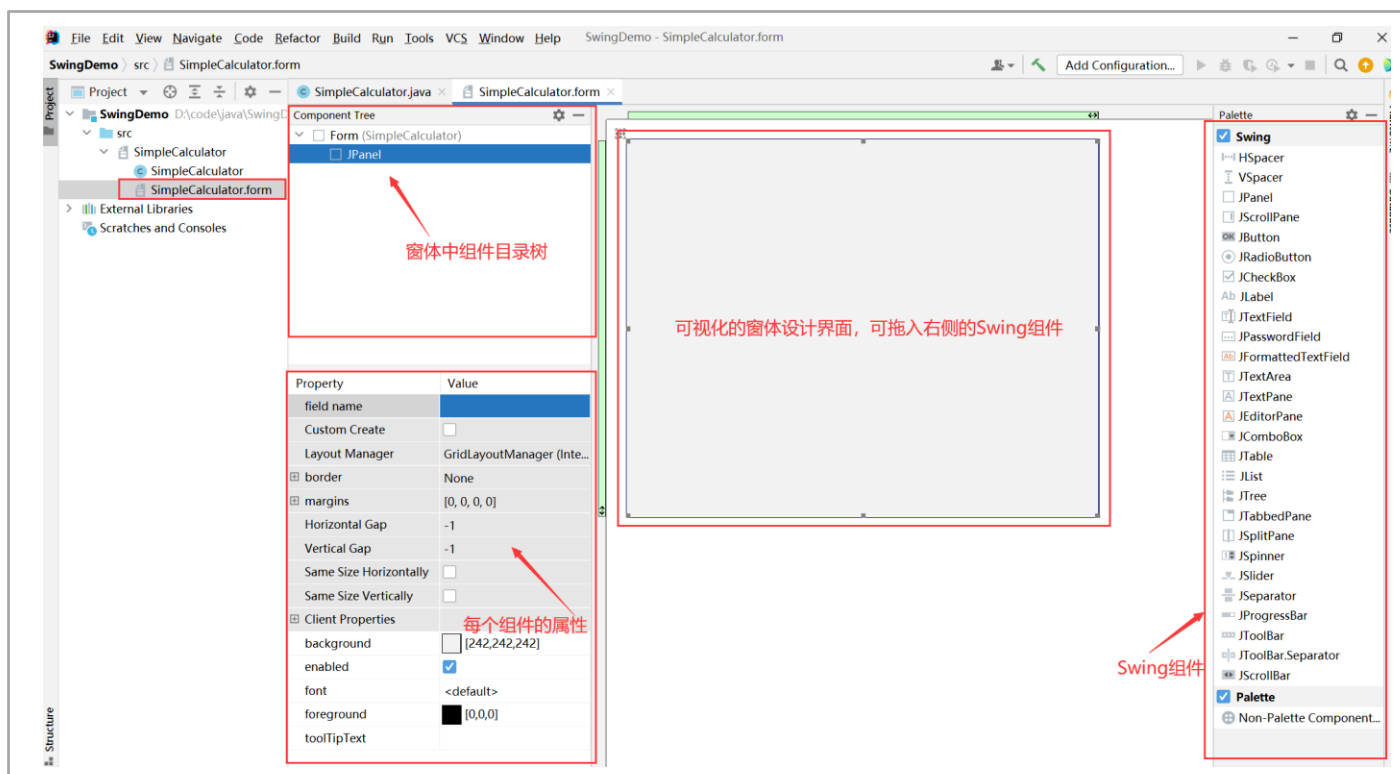
- 是 Java 为图形界面应用开发提供的一组工具包;
- 组件采用MVC模式设计, 实现 GUI 组件的显示逻辑和数据逻辑的分离;
- IntelliJ IDEA提供的Swing GUI Designer可以方便的进行图形界面编程。



# 实验步骤

## 1 Java Swing - Swing GUI Designer

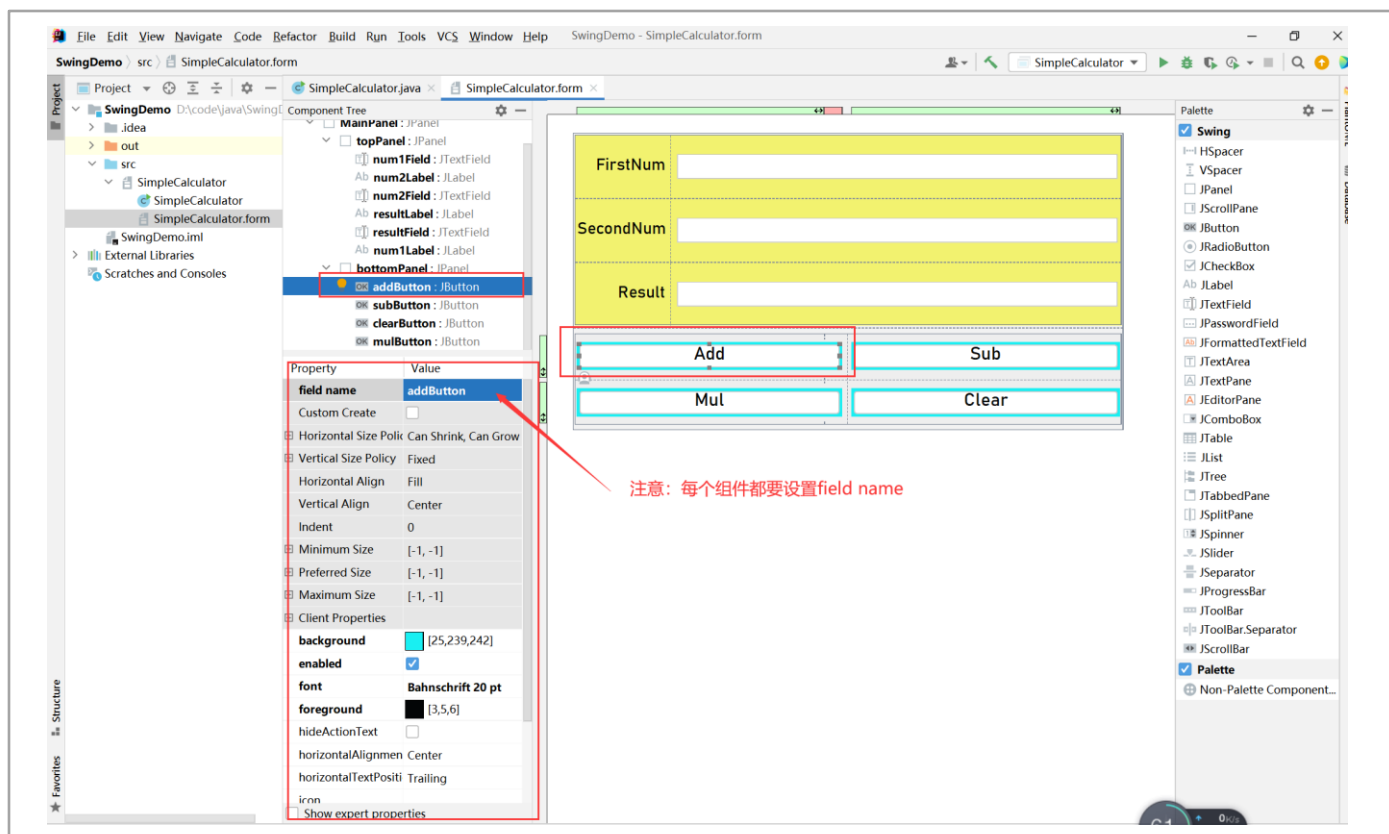
- ①选中项目src目录，右键新建一个**GUI Form**，命名为：SimpleCalculator。
- ②自动生成两个文件：**SimpleCalculator.form** 用于图形界面设计；  
**SimpleCalculator.java** 用于操作组件对象和运行。



# 实验步骤

## 1 Java Swing - Swing GUI Designer

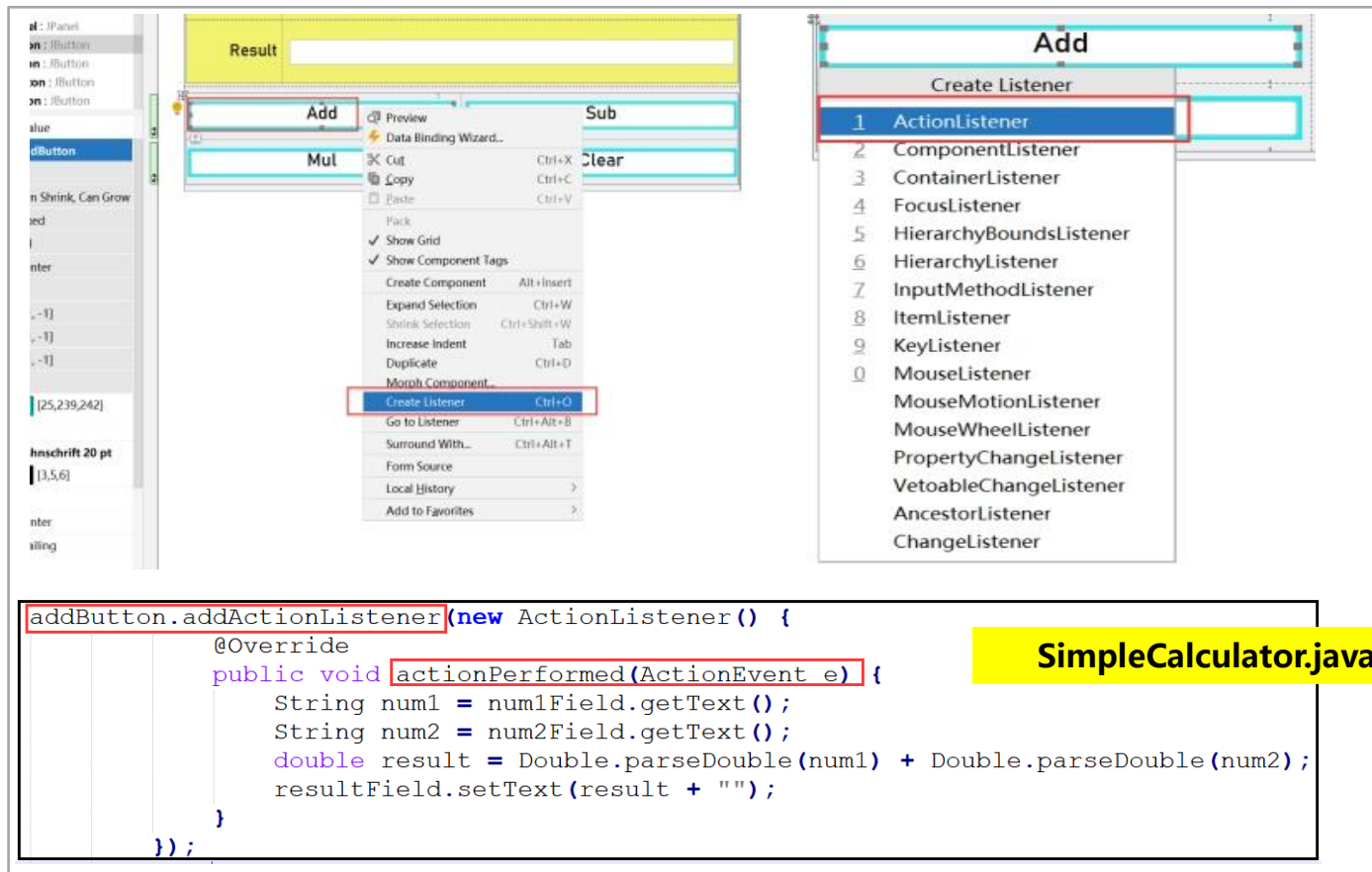
③ 在窗口设计界面中拖入所需要的Swing组件并合理布局。**注意每个组件都要设置field name。**右键选择Preview选项可查看窗体设计的效果。



# 实验步骤

## 1 Java Swing - Swing GUI Designer

④ 添加按钮事件，右键Add按钮，选择Create Listener选项，为其添加 **ActionListener**。在actionPerformed()函数体中添加事件处理代码。



The screenshot illustrates the process of adding an event listener to a button in the Java Swing GUI Designer. The 'Add' button is selected, and the context menu is open, with 'Create Listener' highlighted. The 'Create Listener' dialog is shown, listing various listener interfaces, with 'ActionListener' selected. Below the dialog, the Java code for the ActionListener is displayed, showing the implementation of the actionPerformed method.

```
addButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        String num1 = num1Field.getText();  
        String num2 = num2Field.getText();  
        double result = Double.parseDouble(num1) + Double.parseDouble(num2);  
        resultField.setText(result + "");  
    }  
});
```

**SimpleCalculator.java**



# 实验步骤

## 1 Java Swing - Swing GUI Designer

⑤ 添加Form Main函数，并运行程序。

The image consists of two side-by-side screenshots from an IDE, likely IntelliJ IDEA, illustrating the steps to generate a main method and run a Java Swing application.

**Left Screenshot:** Shows a code editor with a Java class named `SimpleCalculator`. The code includes a constructor and a `main` method. A context menu is open over the `main` method, with the `Generate...` option highlighted. A red arrow points to the right side of the code editor with the text "此处右键" (Right-click here). Below the code editor, a "Generate" dialog box is open, showing a list of options. The `Form main()` option is highlighted.

**Right Screenshot:** Shows the same code editor with the `main` method. A context menu is open over the `main` method, with the `Run 'SimpleCalculator.main()'` option highlighted. Below the code editor, a window titled "SimpleCalculator" is displayed. The window has a yellow background and contains two text input fields labeled "FirstNum" and "SecondNum", and a text output field labeled "Result". The "FirstNum" field contains the value "55", the "SecondNum" field contains the value "88", and the "Result" field contains the value "143.0". At the bottom of the window, there are four buttons: "Add", "Sub", "Mul", and "Clear".



# 实验步骤

## 1 Java Swing - Swing GUI Designer

⑤ 添加Form Main函数，并运行程序。

```
};
mulButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String num1 = num1Field.getText();
        String num2 = num2Field.getText();
        double result = Double.parseDouble(num1) * Double.parseDouble(num2);
        resultField.setText(String.valueOf(result));
    }
});
//end constructor

//end SimpleCalculator

public static void main(String[] args) {
    JFrame frame = new JFrame( title: "SimpleCalculator");
    frame.setContentPane(new SimpleCalculator().MainPanel);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.pack();
    frame.setVisible(true);
}
//end SimpleCalculator
```

Constructor  
Getter  
Setter  
Getter and Setter  
equals() and hashCode()  
toString()  
Override Methods... Ctrl+O  
Delegate Methods...  
Test...  
Copyright  
**Form main()**

SecondNum 88  
Result 143.0  
Add Sub  
Mul Clear



# 实验步骤

## 1 Java Swing - JTable

**JTable** 是将数据以表格的形式显示给用户看的一种组件，它包括行和列。

① 添加所需组件（JScrollPane和JTable）并合理布局。

The screenshot shows the Java Swing IDE interface. On the left, the **Component Tree** panel displays a hierarchy of components: **Form (SimpleTable)** contains **MainPanel : JPanel**, which contains **topPanel : JPanel**. **topPanel** contains **tableScrollPane : JScrollPane** (highlighted with a red box) and **scoreTable : JTable** (also highlighted with a red box). Below **scoreTable** are **headerLabel : JLabel** and **bottomPanel : JPanel**, which contains **deleteButton : JButton**. The **Properties** window below shows the properties for the selected **tableScrollPane** component. On the right, a visual representation of the window titled **成绩表** (Grade Table) is shown. It features a table with two columns: **Shape** and **Color**. The table contains two rows: **round** with **red** and **square** with **green**. Below the table is a button labeled **删除** (Delete).

Property	Value
field name	tableScrollPane
Custom Create	<input type="checkbox"/>
border	None
Horizontal Size Policy	Can Shrink, Can Grow, Want...
Vertical Size Policy	Can Shrink, Can Grow, Want...
Horizontal Align	Fill
Vertical Align	Fill
Indent	0
Minimum Size	[-1, -1]
Preferred Size	[-1, -1]
Maximum Size	[-1, -1]
Client Properties	
background	<input type="checkbox"/> [242,242,242]



# 实验步骤

## 1 Java Swing - JTable

② 在构造函数中创建DefaultTableModel对象，装载数据。

```
public SimpleTable() {  
  
    String[] columnName = {"学号", "姓名", "成绩"};  
    String[][] tableData = {"001", "Lily", "78"}, {"002", "Jane", "89"}, {"003", "Alex", "67"},  
        {"004", "Macy", "83"}, {"005", "Nancy", "66"}, {"006", "John", "99"};  
  
    // 表格模型  
    DefaultTableModel model = new DefaultTableModel(tableData, columnName){  
        @Override  
        public boolean isCellEditable(int row, int col){  
            return false;  
        }  
    };  
  
    // JTable并不存储自己的数据，而是从表格模型那里获取它的数据  
    scoreTable.setModel(model);  
    tableScrollPane.setViewportView(scoreTable);  
}
```

JTable表格与数据通过TableModel分离，JTable并不存储自己的数据，而是从TableModel那里获取数据。



# 实验步骤

## 1 Java Swing - JTable

### ③ 添加“删除”按钮事件和Form Main函数。

```
deleteButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        int row = scoreTable.getSelectedRow();  
        System.out.println(row);  
        if (row != -1) {  
            model.removeRow(row);  
        }  
    }  
});
```

运行程序：

学号	姓名	成绩
001	Lily	78
002	Jane	89
003	Alex	67
004	Macy	83
005	Nancy	66
006	John	99



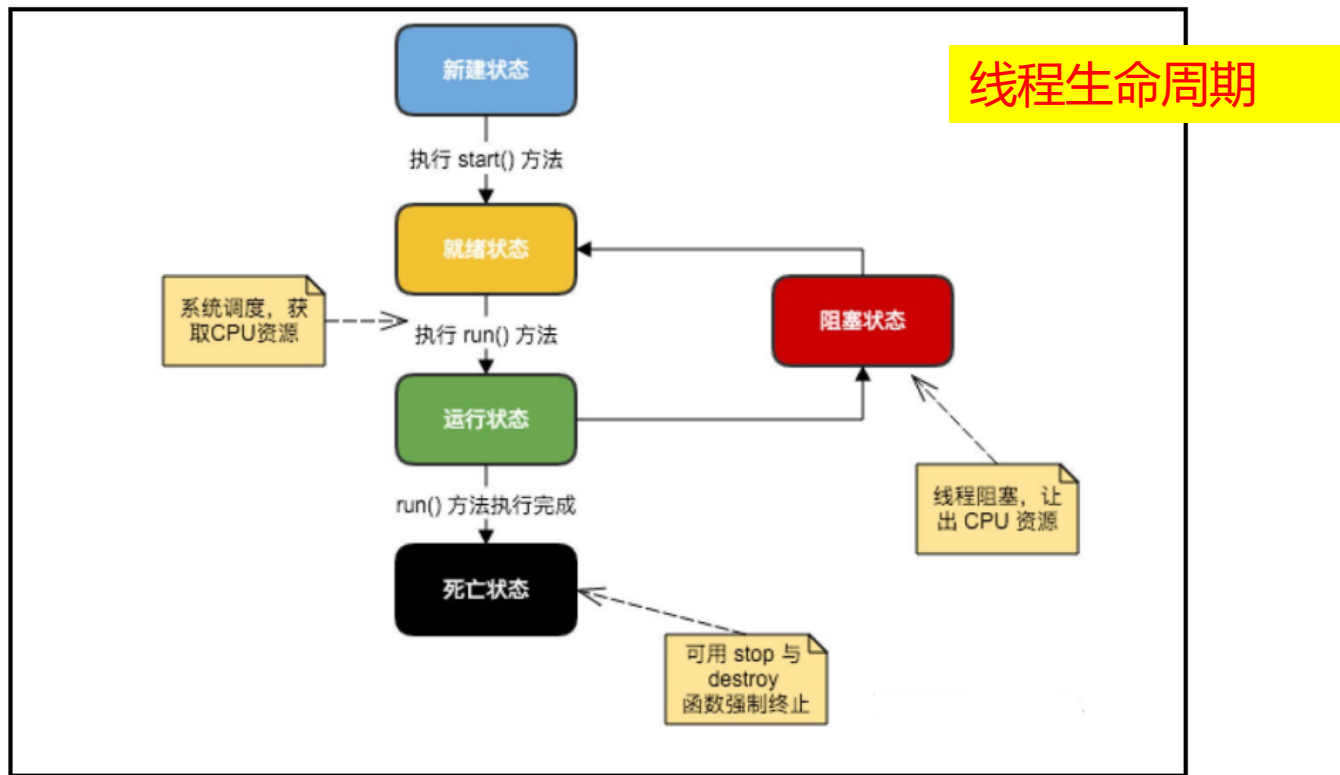


# 实验步骤

## 2

## Java 多线程编程

飞机大战游戏中，**火力道具生效**、**音效多处控制**等功能需要用多线程来完成。Java 给多线程编程提供了内置的支持。线程是一个动态执行的过程，它也有一个从产生到死亡的过程。





# 实验步骤

## 2

## Java 多线程编程

- ① 使用 **Runnable 接口** 实现多线程, 由于Runnable 是函数式接口, 可使用 lambda表达式简写。

```
public class RunnableTest {  
    public static void main(String[] args) {  
  
        Runnable r = () -> {  
            try {  
                for (int i = 0; i < 3; i++) {  
                    System.out.println("【" + Thread.currentThread().getName() +  
                        "】线程执行, 当前的循环次数: " + i);  
  
                    Thread.sleep(2000);  
                }  
            } catch (InterruptedException e) {  
                e.printStackTrace();  
            }  
        };  
  
        // 启动线程  
        new Thread(r, "线程1").start();  
        new Thread(r, "线程2").start();  
    }  
}
```

【线程 1】线程执行, 当前的循环次数: 0

【线程 2】线程执行, 当前的循环次数: 0

【线程 1】线程执行, 当前的循环次数: 1

【线程 2】线程执行, 当前的循环次数: 1

【线程 1】线程执行, 当前的循环次数: 2

【线程 2】线程执行, 当前的循环次数: 2



# 实验步骤

## 2

## Java 多线程编程

- ② 继承 Thread 类实现多线程，继承类必须重写 run() 方法，该方法是新线程的入口点。

本实验已提供MusicThread类，该类继承Thread类，重写了run()方法，用于启动音频播放。

```
@Override
public void run() {
    InputStream stream = new ByteArrayInputStream(samples);
    play(stream);
}
```



# 实验步骤

## 2

## Java 多线程编程

- ② 继承 Thread 类实现多线程，继承类必须重写 run() 方法，该方法是新线程的入口点。

使用方法：

```
public class ThreadTest {  
  
    public static void main(String[] args) {  
        new MusicThread("src/bgm.wav").start();  
    }  
}
```

**注意：**在飞机大战游戏中，还需实现循环播放、停止播放音频的功能。



# 实验步骤

## 2

## Java 多线程编程

提示：可利用线程等待做页面切换。



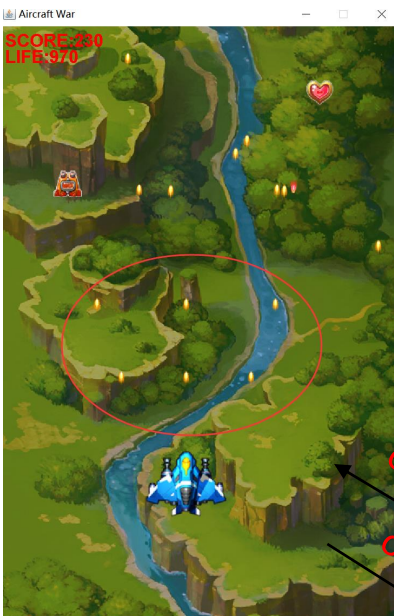


# 实验步骤

## 2

## Java 多线程编程

提示：可利用线程等待做页面切换。



2

Main

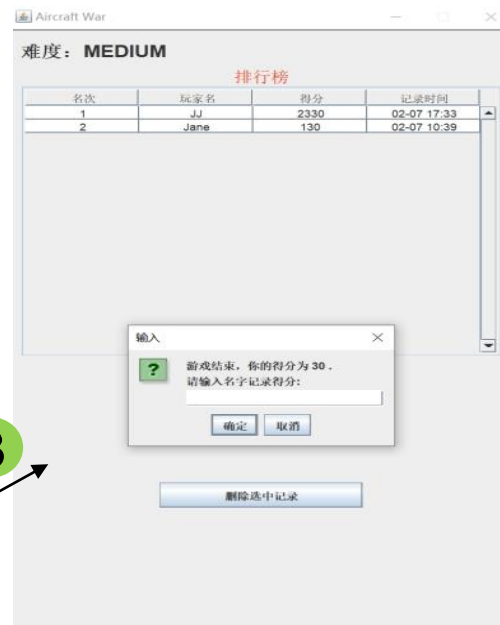
```
create menuPanel
frame.setContentPane(menuPanel)
frame.setVisible(true)
...
wait
...
frame.remove(menuPanel)
create gamePanel
frame.setContentPane(gamePanel)
frame.setVisible(true)
...
wait
...
frame.remove(gamePanel)
create boardPanel
frame.setContentPane(boardPanel)
frame.setVisible(true)
```

1

object.wait()

object.notify()

3





# 实验要求

---

本次实验提交版本需完成以下功能：

- ✓ 游戏开始显示**难度选择和音效设置界面**，根据玩家选择显示相应难度的**游戏地图**。（本实验**无需细化**三种不同游戏难度，替换地图即可，实验六进一步完善。）
- ✓ 游戏结束后显示**得分排行榜界面**，可输入玩家姓名并保存得分记录，可删除玩家某次历史得分。
- ✓ 若音效开启，游戏中**循环播放**游戏背景音乐，游戏结束后**停止播放**。子弹击中敌机、炸弹爆炸、道具生效、游戏结束时有相应的音效。**Boss敌机**出场时循环播放其背景音乐，坠毁后停止播放。
- ✓ 火力道具生效时，英雄机由直射切换为散射弹道并**持续**一段时间，结束后**恢复**原状态。



# 作业提交

---

- 提交内容

把整个项目目录压缩成zip包提交。

- 截止时间

实验课后一周内提交至HITsz Grader 作业提交平台，  
具体截止日期参考平台发布。





**同学们  
请开始实验吧！**