实验描述
实验过程
生日数据输入的界面
绘制生物节律曲线
切换用户功能

邓语苏 信息学院 计科

21级 2023.7.7

实验描述

1. 根据生物节律曲线的定义，用户输入生日后，绘制生物节律曲线（使用JavaFX）；
2. 能保存用户生日，并允许切换用户——即保存多个生日并进行切换。

实验过程

生日数据输入的界面

- 创建基础框架
- 修改按钮css风格
- 检查输入数据(生日和姓名)的正确性

```
// 创建生日输入界面
private GridPane createInputPane() {
    // 创建一个网格面板
    GridPane inputPane = new GridPane();
    inputPane.setAlignment(Pos.CENTER); // 设置面板的对齐方式为居中
    inputPane.setHgap(10); // 设置水平间距为10个像素
    inputPane.setVgap(10); // 设置垂直间距为10个像素

    // 创建姓名标签和文本框
    Label nameLabel = new Label("姓名:");
    TextField nameField = new TextField();

    // 创建生日标签和文本框
    Label birthdayLabel = new Label("生日:(年-月-日)");
    TextField birthdayField = new TextField();

    // 创建提交按钮
    Button submitButton = new Button("提交");
    submitButton.setStyle("-fx-background-color: lightblue; -fx-text-fill:
black; -fx-border-color: black; -fx-border-width: 2px; -fx-border-radius:
5px; -fx-effect: dropshadow(three-pass-box, rgba(0,0,0,0.3), 5, 0, 0, 1);");

    // 注册提交按钮的点击事件处理程序
    submitButton.setOnAction(e -> {
        String name = nameField.getText(); // 获取姓名文本框中的内容
        String birthdayStr = birthdayField.getText(); // 获取生日文本框中的内容
```

```
25
26          // 检查姓名和生日是否都不为空
27          if (!name.isEmpty() && !birthdayStr.isEmpty()) {
28              LocalDate birthday = LocalDate.parse(birthdayStr); // 将生日字符串
    解析为 LocalDate 对象
29              birthdays.put(name, birthday); // 将姓名和生日存储到一个集合中
30              drawBioRhythm(name, birthday); // 调用方法绘制生物节律图表
31          }
32      });
33
34      // 将姓名标签、姓名文本框、生日标签、生日文本框和提交按钮添加到网格面板中
35      inputPane.add(nameLabel, 0, 0);
36      inputPane.add(nameField, 1, 0);
37      inputPane.add(birthdayLabel, 0, 1);
38      inputPane.add(birthdayField, 1, 1);
39      inputPane.add(submitButton, 0, 2, 2, 1); // 指定按钮跨两列和一行
40
41      return inputPane; // 返回创建的网格面板
42  }
43
```

绘制生物节律曲线

- 由生日计算生物节律的数据散点
- 添加到Series中
- 将曲线加入图表中

```
1   // 绘制生物节律曲线
2   private void drawBioRhythm(String name, LocalDate birthday) {
3       // 创建一个 XYChart.Series 对象，用于表示生物节律曲线的数据系列
4       XYChart.Series<Number, Number> series = new XYChart.Series<>();
5       series.setName(name); // 设置数据系列的名称为姓名
6
7       LocalDate currentDate = LocalDate.now(); // 获取当前日期
8       long daysSinceBirth = ChronoUnit.DAYS.between(birthday, currentDate); //
    计算从生日到当前日期的天数
9
10      // 循环绘制30个数据点，表示30天的生物节律
11      for (int i = 0; i <= 30; i++) {
12          double x = i; // x轴坐标为循环变量i的值
13          double y = Math.sin(2 * Math.PI * (daysSinceBirth + i) / 23); // 使用
    生物节律的示例计算公式计算y轴坐标，可以根据需要进行修改
14          series.getData().add(new XYChart.Data<>(x, y)); // 将数据点添加到数据系
    列中
15      }
16
17      chart.getData().add(series); // 将数据系列添加到图表中显示
18  }
19
```

切换用户功能

- 创建按钮，绘制框架
- 绘制选中用户的生物节律曲线

- 返回主页面

```
// 切换用户
private void switchUser(Stage primaryStage) {
    // 创建切换用户界面
    GridPane switchUserPane = new GridPane();
    switchUserPane.setAlignment(Pos.CENTER); // 设置面板的对齐方式为居中
    switchUserPane.setHgap(10); // 设置水平间距为10个像素
    switchUserPane.setVgap(10); // 设置垂直间距为10个像素

    // 创建生日输入界面
    GridPane inputPane = createInputPane();

    // 创建曲线绘制界面
    NumberAxis xAxis = new NumberAxis();
    NumberAxis yAxis = new NumberAxis();
    chart = new LineChart<>(xAxis, yAxis);
    chart.setTitle("节律曲线图");

    // 创建切换用户按钮
    Button switchUserButton = new Button("切换用户");
    switchUserButton.setStyle("-fx-background-color: lightblue; -fx-text-fill: black; -fx-border-color: black; -fx-border-width: 2px; -fx-border-radius: 5px; -fx-effect: dropshadow(three-pass-box, rgba(0,0,0,0.3), 5, 0, 0, 1);");

    // 注册切换用户按钮的点击事件处理程序
    switchUserButton.setOnAction(e -> switchUser(primaryStage));

    // 创建主要布局容器
    GridPane mainPane = new GridPane();
    mainPane.setAlignment(Pos.CENTER); // 设置中央对齐
    mainPane.setHgap(10); // 设置水平间距
    mainPane.setVgap(10); // 设置垂直间距
    mainPane.setPadding(new Insets(10)); // 设置内边距
    mainPane.add(inputPane, 0, 0); // 在第一行第一列添加inputPane
    mainPane.add(chart, 0, 1);
    mainPane.add(switchUserButton, 0, 2); // 第三行--切换用户按钮

    AtomicReference<String> selectedUser = new AtomicReference<>("");

    int row = 1;
    for (String user : birthdays.keySet()) {
        Button userButton = new Button(user);
        GridPane.setConstraints(userButton, 0, row);
        switchUserPane.getChildren().add(userButton);
        userButton.setOnAction(e -> {
            selectedUser.set(userButton.getText());
        });
        row++;
    }

    Button switchButton = new Button("切换");
    switchButton.setOnAction(e -> {
        String user = selectedUser.get();
```

```
        if (!user.isEmpty()) {
            LocalDate birthday = birthdays.get(user);
            chart.getData().clear(); // 清空图表中的数据系列
            drawBioRhythm(user, birthday); // 绘制选中用户的生物节律曲线
        }

        primaryStage.getScene().setRoot(mainPane); // 切换回主要布局容器

    });

    switchUserPane.getChildren().add(switchButton);

    Scene switchUserScene = new Scene(switchUserPane, 300, 200); // 创建切换用户界面的场景
    primaryStage.setScene(switchUserScene); // 设置主舞台的场景为切换用户界面
}

```