

Создание графического интерфейса пользователя GUI на языке Java

Языков Кирилл Александрович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

Статья посвящена разработке графического интерфейса пользователя (GUI) на языке Java, который является одним из самых популярных языков программирования для создания кроссплатформенных приложений. В ней рассматриваются основные концепции и инструменты, используемые для проектирования и реализации интерфейсов. Задача будет состоять в том, чтобы сделать понятный и простой интерфейс для любого пользователя. Каждый сможет понять для себя принцип взаимодействия пользователя с приложением, а программист – взаимодействие с кодом.

Ключевые слова: Графический интерфейс, Java, кроссплатформенные приложения.

Creating a graphical user interface GUI in Java

Yazykov Kirill Alexandrovich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article is devoted to the development of a graphical user interface (GUI) in Java, which is one of the most popular programming languages for creating cross-platform applications. It discusses the basic concepts and tools used to design and implement interfaces. The task will be to create a clear and simple interface for any user. Everyone will be able to understand for themselves the principle of user interaction with the application, and the programmer – interaction with the code.

Keywords: Graphical interface, Java, cross-platform applications.

1 Введение

1.1 Актуальность

Создание графического интерфейса пользователя (GUI) на языке Java остается актуальной и важной темой в современном программировании. В условиях стремительного развития технологий и увеличения числа вычислительных устройств, необходимость в интуитивно понятных и функциональных интерфейсах становится все более значимой. Java, как один из ведущих языков программирования, предлагает мощные инструменты и библиотеки, такие как Swing и JavaFX, которые позволяют разработчикам

создавать кроссплатформенные приложения с высококачественным пользовательским интерфейсом.

1.2 Обзор исследований

Руководство по освоению языка программирования Java с примерами, которые так или иначе помогают начинающему программисту в освоении тяжелых средств для создания приложения [1]. Для того чтобы начинающий программист мог понять по какому направлению стоит развивать навыки. А также рассмотреть различные технологии и дальнейшие перспективы в работе. [2]. Важным аспектом является создание двухмерных фигур и управления цвета в нем, каждая из технологий рассматривается подробно, от самых небольших до более профессиональных. Любой пользователь сможет управлять стилем, наложением и эффектами. [3]. Мониторинг устаревших инструментов, которые можно заменить дополнительными плагинами, библиотеками было проведено исследование, для того чтобы пользователь смог открыть для себя полезные возможности, о которых знает не каждый пользователь IDE [4].

1.3 Цель исследования

Цель данного исследования заключается в изучении и анализе методов и инструментов для разработки графического интерфейса пользователя на языке Java, с акцентом на создание удобных и интуитивно понятных интерфейсов. Исследование направлено на выявление лучших практик проектирования GUI, а также на оценку влияния различных библиотек, таких как Swing и JavaFX, на эффективность разработки.

2 Материалы и методы

Для реализации данного исследования будет использован IntelliJ IDEA (IDE для Java), библиотека awt.

3 Результаты

В данной работе был описан процесс создания небольшой программы с графическим интерфейсом на языке Java, что позволяет студентам, знакомым с основами языка, получить практический опыт в разработке GUI-приложений. Программа включает в себя основные элементы управления, такие как кнопки, текстовые поля и метки, что позволяет пользователю взаимодействовать с приложением. Процесс разработки начался с проектирования интерфейса, где были определены основные функции и элементы, необходимые для выполнения задач приложения. Результатом работы стала простая, но функциональная программа, демонстрирующая основные принципы создания GUI на Java. Этот опыт позволяет студентам лучше понять, как проектировать и реализовывать пользовательские интерфейсы, а также развивает навыки работы с библиотеками Java для создания интерактивных приложений.

Открываем IntelliJ IDEA и создаем новый проект: File -NewProject.

В старых версиях IntelliJ IDEA, папка src пустая, в новых версиях папка src содержит автоматически созданный класс Main с функцией main и готовой программой "Hello, World!".

Если установлена старая версия IntelliJ IDEA самостоятельно можно создать новый класс Main с функцией main и написать программу "Hello, World!".

Уже сейчас возможно запустить проект (Run). Внизу в терминале IDE можно увидеть сообщение “ Hello, World!“

На данном этапе уже есть работающая программа, но без поддержки GUI. А сейчас в той же папке src создадим GUI Form :

New -> GUI Form

Имя новой формы задайте MainWindow (Рис. 1).

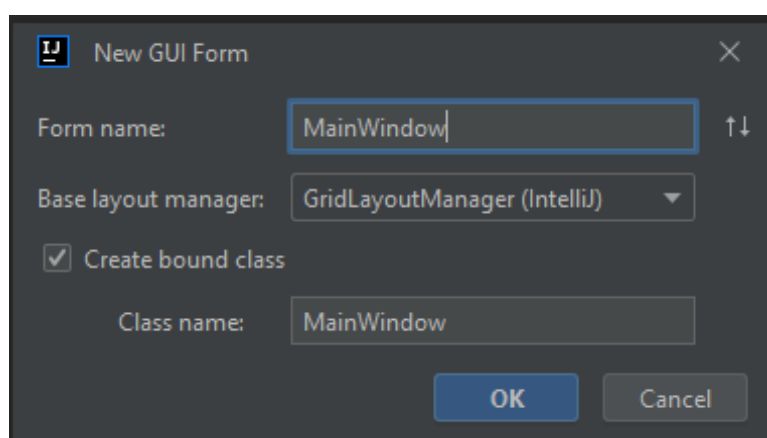


Рисунок 1 – Создание новой GUI формы

Открываем созданную GUI форму, нажимаем на JPanel и задаем его идентификатор в поле fieldname, в примере задано panel. После чего перетаскиваем на форму с правой стороны JTextField, JPasswordField и JButton (Рис. 2).

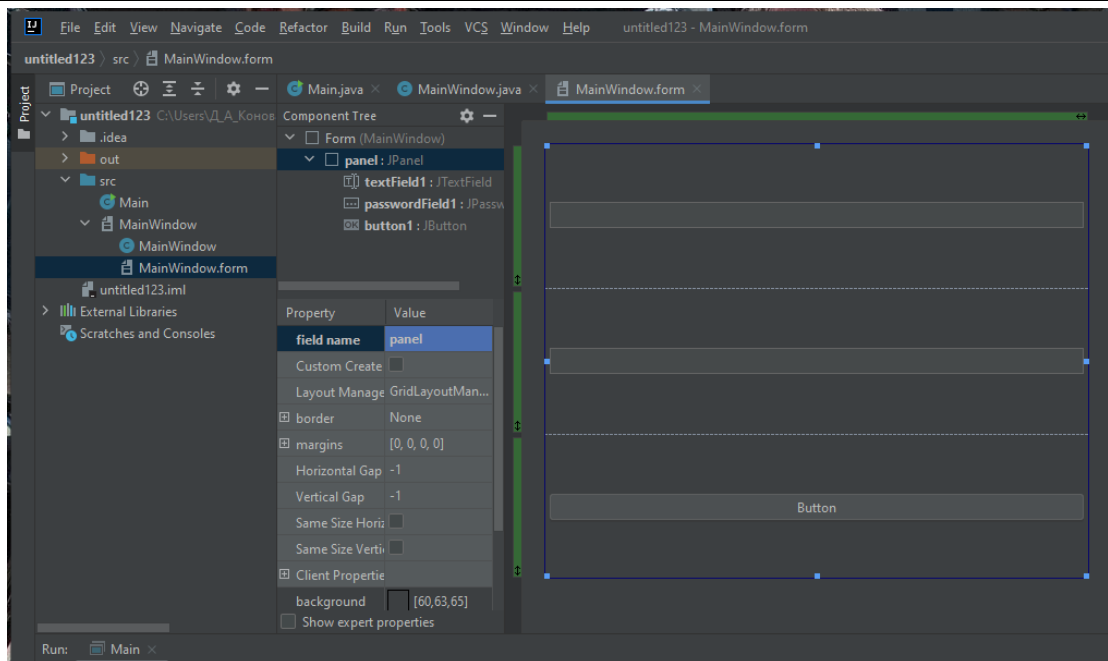


Рисунок 2 – Вставка значений в панели

Добавьте в строку:

`public class MainWindow extends JFrame{` //что означает – расширение функций GUI.

В данный момент имеем форму MainWindow и класс MainWindow расширенный с помощью JFrame.

Теперь необходимо определить все добавленные GUI элементы как содержание класса MainWindow `this.getContentPane().add(panel);` После чего содержание файла MainWindow.java будет изменено следующим образом:

```

import javax.swing.*;
public class MainWindow extends JFrame {
    private JTextField textField1;
    private JPasswordField passwordField1;
    private JButton button1;
    private JPanel panel;
    public MainWindow() {this.getContentPane().add(panel);
    }
}

```

Если попробовать запустить код, можно снова увидеть такое же сообщение “Hello, World!”. Дело в том, что были созданы класс и форма к нему, но не создана инстанция этого класса.

Изменим файл Main.java и добавим код создания GUI (Рис. 3):

```
import java.awt.*; //Подключаем библиотеку awt
public class Main {
    public static void main(String[] args) { // Начинаем программу
        MainWindow mainWindow = new MainWindow(); // Создаем
инстанцию класса MainWindow
        mainWindow.pack(); // Упаковываем все элементы с нашей формы
        mainWindow.setSize(new Dimension(200, 200)); // Изменяем
размеры окна
        mainWindow.setVisible(true); // Отображаем созданное окно
    }
}
```

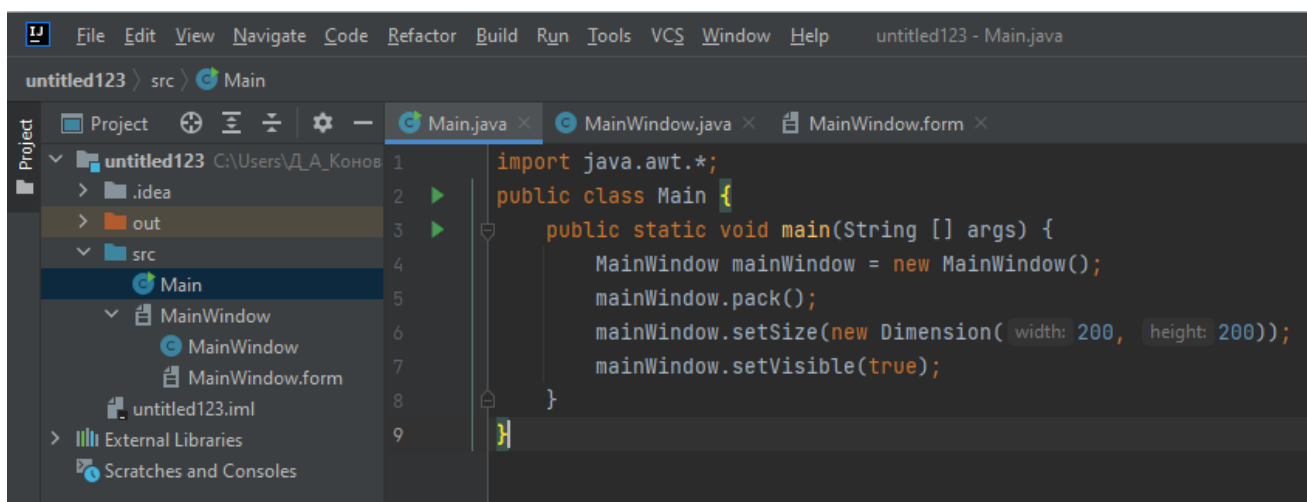


Рисунок 3 – Код для GUI

Если запустить код, то появится окно с кнопкой Button, но если нажать на нее, то ничего не произойдет, так как нет слушателя для события Events.

Кнопку необходимо привязать к действию или слушатель событий (EventListener) JButton должен быть имплементацией адаптера ActionListener, поэтому добавим следующий код в тело класса MainWindow.

Открыть форму MainWindow.form, правой кнопкой вызвать контекстное меню кнопки «Button» и в контекстном меню выбрать команду CreateListener (Рис. 4).

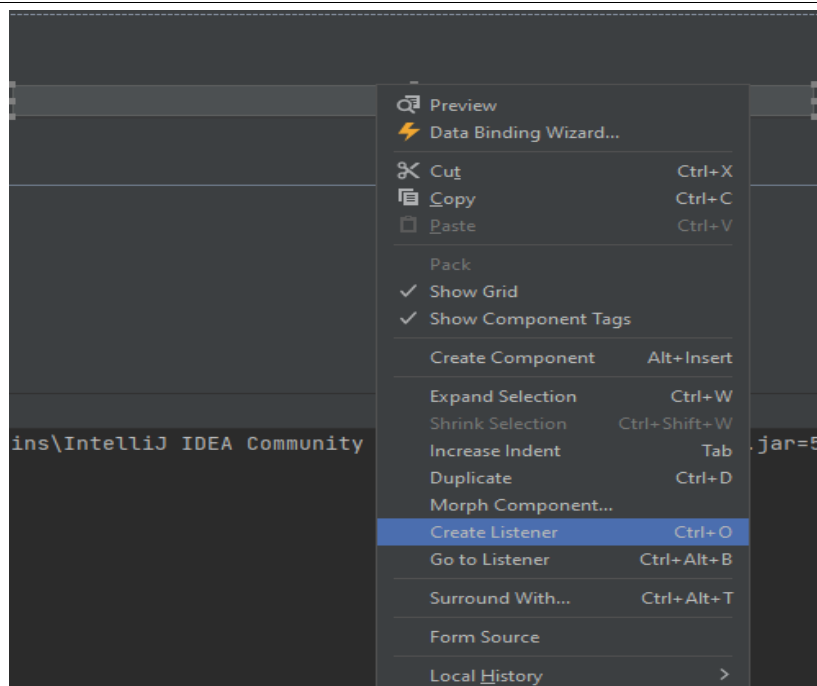


Рисунок 4 – Создание слушателя

В подменю следует выбрать пункт ActionListener. Затем подтвердить действия. В результате в код класса MainWindow автоматически добавится слушатель событий (Рис. 5).

```

4 public MainWindow(){
5     this.getContentPane().add(panel);
6     button1.addActionListener(new ActionListener() {
7         @Override
8         public void actionPerformed(ActionEvent actionEvent) {
9             if (textField1.getText().equals(passwordField1.getText())){
10                 JOptionPane.showMessageDialog( parentComponent: null, message: "Поздравайтесь");} else {
11                 JOptionPane.showMessageDialog( parentComponent: null, message: "Hello World!");}
12             }
13         }
14     });
15 }

```

Рисунок 5 – Визуализация слушателя в коде

Метод actionPerformed () будет обрабатывать все события кнопки button1, но для начала еще необходимо указать кнопке button1 какой класс будет обрабатывать, поэтому добавим следующий код в конструктор класса MainWindow:

```
button1.addActionListener(new MyButtonListener());
```

Чтобы обработчик не был бессмысленным требуется добавить следующий код в метод actionPerformed () (Рис. 6):

```

@Override
public void actionPerformed(ActionEvent actionEvent) {
    if (textField1.getText().equals(passwordField1.getText())) {
        JOptionPane.showMessageDialog(null, "Введите сообщение"); }
}

```

```
else { JOptionPane.showMessageDialog(null, "Привет  
пользователь");  
}  
}
```

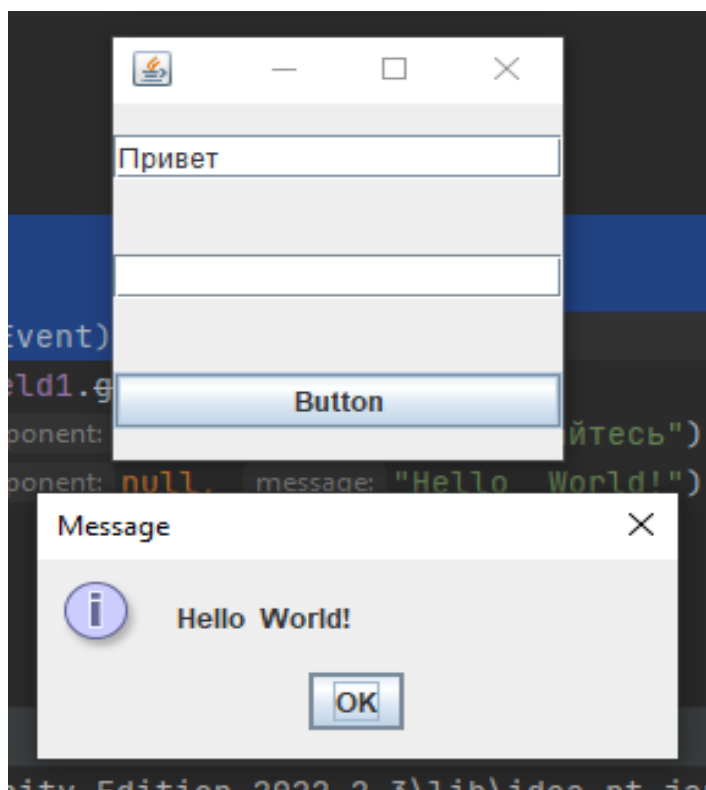


Рисунок 6 – Проверка работы кода

Выводы

Создание графического интерфейса пользователя (GUI) на языке Java является важной и актуальной задачей в области разработки программного обеспечения. Работа с GUI на Java требует понимания принципов проектирования интерфейсов, а также навыков программирования для реализации взаимодействия между элементами интерфейса и логикой приложения. В результате, разработка GUI не только улучшает пользовательский опыт, но и способствует созданию более качественных и конкурентоспособных продуктов.

Таким образом, освоение создания графических интерфейсов на Java открывает новые возможности для разработчиков, позволяя им адаптироваться к современным требованиям и тенденциям в области программирования. Это исследование подчеркивает важность GUI в взаимодействии пользователя с программным обеспечением и его влияние на успех приложений в условиях растущей конкуренции на рынке.

Библиографический список

1. Нимейер П., Леук Д. Программирование на Java. Litres, 2022.
2. Сафонов В. О. Java-технология: история, состояние и перспективы // Компьютерные инструменты в образовании. 2003. №. 2. С. 26-33.
3. Елепбергенова А. У., Канапьянова З. Н., Мурсакимова Г. А. Графические возможности JAVA // Редакционная коллегия. 2018.
4. An I. D. E. WatchDog For IntelliJ: An IDE Plugin To Analyze Software Testing Practices.