Конспект первой пары по JavaFX: Базовые основы FXML и простое приложение

Подсказка лектору

Октябрь 2025

1. Общие принципы урока

Цель: Ввести в основы JavaFX через FXML: настроить проект в IntelliJ IDEA, создать простое FXML-приложение (калькулятор суммы двух чисел), изучить базовые контролы и layouts. Фокус на практике: 30--40% теории, 60--70% демо в Scene Builder + IntelliJ.

Аудитория: Новички в JavaFX (знают базовый Java).

Формат: Теория кратко, демонстрации в IntelliJ + Scene Builder. Каждый шаг -- показ на экране с комментариями в FXML и пояснениями. Использовать комментарии XML для ключевых понятий.

Распределение времени: 80 мин.

2. Введение и настройка проекта (0--10 мин)

0--2 мин: Введение.
FXML -- декларативный XML для UI (как HTML). Позволяет создавать GUI на Java. К концу пары студенты создают FXML-приложение для суммирования чисел. Вербально объяснить и нарисовать диаграмму: Stage → Scene → Nodes.

2--8 мин: Настройка в IntelliJ.

1. File → New → Project → Import (открыть готовый Gradle-проект).
2. В `build.gradle` уже добавлена зависимость:

```
dependencies {
    implementation 'org.openjfx:javafx-controls:21'
}
```

3. Скачать Scene Builder и интегрировать: Right-click на FXML → Open in Scene Builder.

Демо: создать `sample.fxml` и main.

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.VBox?>
```

```
<VBox xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/
   fxml/1" spacing="10" alignment="CENTER">
   <Label text="Hello, FXML!" />
</VBox>
```

```
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class MainApp extends Application {
    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("/sample.
           fxml"));
        Scene scene = new Scene(root, 300, 200);
        stage.setScene(scene);
        stage.setTitle("Hello FXML");
        stage.show();
    }
}
```

3. База JavaFX: Stage, Scene, граф и FXML (10--30 мин)

10--15 мин: Stage → Scene → граф.
JavaFX строится как дерево: Stage содержит Scene, Scene -- root-узел (layout или control). Node -- элемент UI (текст, кнопка), родитель управляет положением и размером детей.

15--25 мин: FXML.
FXML -- декларативный XML для UI. Используется в Scene Builder. Плюсы: разделение UI/логики, легкое редактирование. Элементы: `<?import?>`, `xmlns`, `fx:controller`.

4. 3. Основные Layouts в FXML (30--50 мин)

30--40 мин: HBox и VBox.
Контейнеры для размещения контролов:

- `alignment="BOTTOM_LEFT"` -- выравнивание содержимого (Pos: TOP_LEFT, CENTER и др.)
- `HBox.hgrow="ALWAYS"` -- растяжка по ширине
- `VBox.vgrow="ALWAYS"` -- растяжка по высоте
- `spacing="10"` -- расстояние между детьми
- `padding="10"` -- внутренние отступы

- `VBox.margin="10"` / `HBox.margin="10"` -- внешние отступы для детей

Пример: `<VBox spacing="10" alignment="CENTER" padding="10"><Label text="Пример"/></VBox>`

Константы Pos для alignment:

Константа	Описание
BASELINE_CENTER	Позиционирование по базовой линии вертикально и по центру горизонтально.
BASELINE_LEFT	По базовой линии вертикально и слева горизонтально.
BASELINE_RIGHT	По базовой линии вертикально и справа горизонтально.
BOTTOM_CENTER	Снизу вертикально и по центру горизонтально.
BOTTOM_LEFT	Снизу вертикально и слева горизонтально.
BOTTOM_RIGHT	Снизу вертикально и справа горизонтально.
CENTER	По центру вертикально и горизонтально.
CENTER_LEFT	По центру вертикально и слева горизонтально.
CENTER_RIGHT	По центру вертикально и справа горизонтально.
TOP_CENTER	Сверху вертикально и по центру горизонтально.
TOP_LEFT	Сверху вертикально и слева горизонтально.
TOP_RIGHT	Сверху вертикально и справа горизонтально.

40--50 мин: Pane, GridPane и AnchorPane.

- Pane -- базовый, ручное расположение (layoutX/Y)
- GridPane -- таблицы (rowIndex, columnIndex, hgap, vgap, padding)
- AnchorPane -- якоря к краям (topAnchor, leftAnchor)

Примеры в FXML показаны выше.

5. Базовые Controls в FXML (50--60 мин)

50--55 мин: Label, Button, TextField.

- Label: `<Label fx:id="myLabel" text="Пример" font="Arial 14"/>`
- Button: `<Button fx:id="myButton" text="Клик" onAction="#handleClick"/>`
- TextField: `<TextField fx:id="myField" promptText="Ввод" prefWidth="100"/>`

55--60 мин: CheckBox, RadioButton, Tooltip.

- CheckBox: `<CheckBox fx:id="myCheck" text="Согласен"/>`

- RadioButton с ToggleGroup:

```
<HBox xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <fx:define>
            <ToggleGroup fx:id="group"/>
        </fx:define>
        <RadioButton text="Да" toggleGroup="$group"/>
        <RadioButton text="Нет" toggleGroup="$group"/>
    </children>
</HBox>
```

- Tooltip: использовать `Tooltip.install(myButton, new Tooltip("Текст"));`

6. Events и создание приложения в FXML (60--80 мин)

60--65 мин: Events.
Для Button: `onAction="#handleSum"` -- ссылка на метод в контроллере.
65--80 мин: Полное демо-приложение.
FXML (`sample.fxml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<VBox xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/
   fxml/1" fx:controller="com.example.SumController" spacing="10"
   alignment="CENTER">
   <Label text="Калькулятор␣суммы"/>
   <HBox spacing="5">
      <TextField fx:id="num1" promptText="Первое␣число" prefWidth="100"/>
      <Label text="+"/>
      <TextField fx:id="num2" promptText="Второе␣число" prefWidth="100"/>
   </HBox>
   <Button fx:id="sumButton" text="Сумма" onAction="#handleSum"/>
   <Label fx:id="resultLabel" text="Результат:␣0"/>
</VBox>
```

Контроллер (`SumController.java`):

```
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;

public class SumController {
    @FXML private TextField num1;
    @FXML private TextField num2;
    @FXML private Label resultLabel;

    @FXML
    private void handleSum(ActionEvent event) {
        try {
```

```
            double a = Double.parseDouble(num1.getText());
            double b = Double.parseDouble(num2.getText());
            resultLabel.setText("Результат: " + (a + b));
        } catch (NumberFormatException e) {
            resultLabel.setText("Ошибка: введите числа!");
        }
    }
}
```

Main -- как выше. Запуск: ввод 5 + 3 $\rightarrow$ ``Результат: 8.0''.