
Interfejs graficzny (GUI) w JavaFX

Piotr Harmuszkiewicz, Tomasz Gajda

The background features abstract, colorful shapes in shades of blue, green, and purple. There are several overlapping circles and ellipses, some with solid outlines and others with dotted outlines, creating a modern, artistic feel.

Spis treści

01. Podstawy

Czym jest JavaFX?
Wstęp teoretyczny

02. Instrukcja

Jak skonfigurować
środowisko?

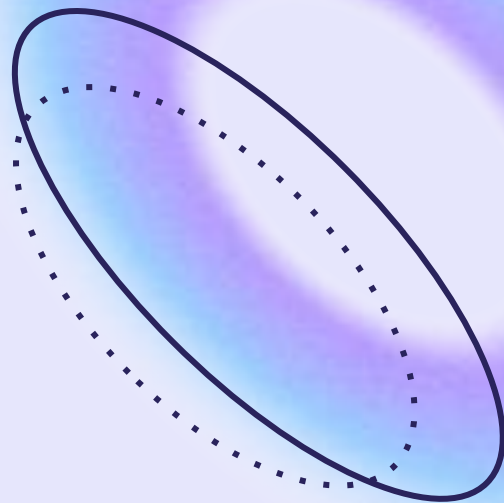
03. Przykłady

Praktyczne wykorzystanie
JavaFX

01.

Podstawy

Czym jest JavaFX i do czego służy?



Czym jest JavaFX?

Jest to biblioteka służąca do tworzenia graficznych interfejsów użytkownika w języku Java. JavaFX została stworzona i dodana do Java 8 aby zastąpić wcześniejszą bibliotekę Swing jako domyślną bibliotekę do tworzenia interfejsów aplikacji okienkowych.



Chris Oliver

Zalety JavaFX

- JavaFX jest dostępna w wiodących systemach operacyjnych dla komputerów stacjonarnych.
- Daje możliwość tworzenia aktualnych interfejsów użytkownika.
- Opiera się na jasnym i przejrzystym języku.
- Zapewnia wszystkie profesjonalne narzędzia Java wymagane do debugowania, analizowania, profilowania i rejestrowania aplikacji klienckiej.
- Umożliwia prostą instalację.



Brutalna rzeczywistość

Branża w większości ustandaryzowała **JavaScript (TypeScript)** jako technologię dla interfejsów użytkownika. Nawet aplikacje desktopowe są teraz pisane w JS.

Najpopularniejsze są napisane przy użyciu **JavaScript'a i frameworka Electron**.

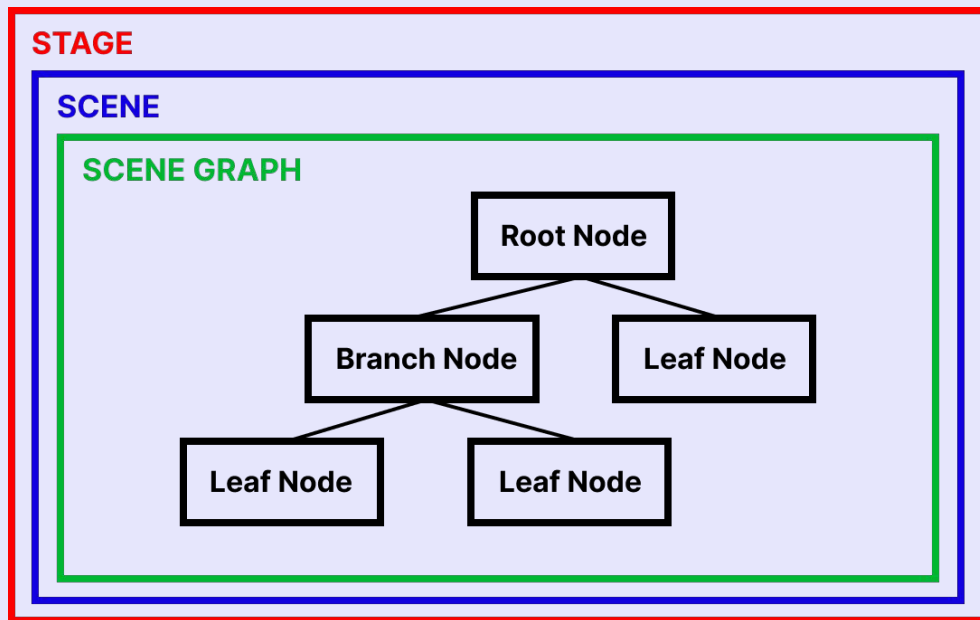




Wstęp

Teoretyczny

Struktura okienka JavaFX



Aplikacja JavaFX składa się z obiektów:

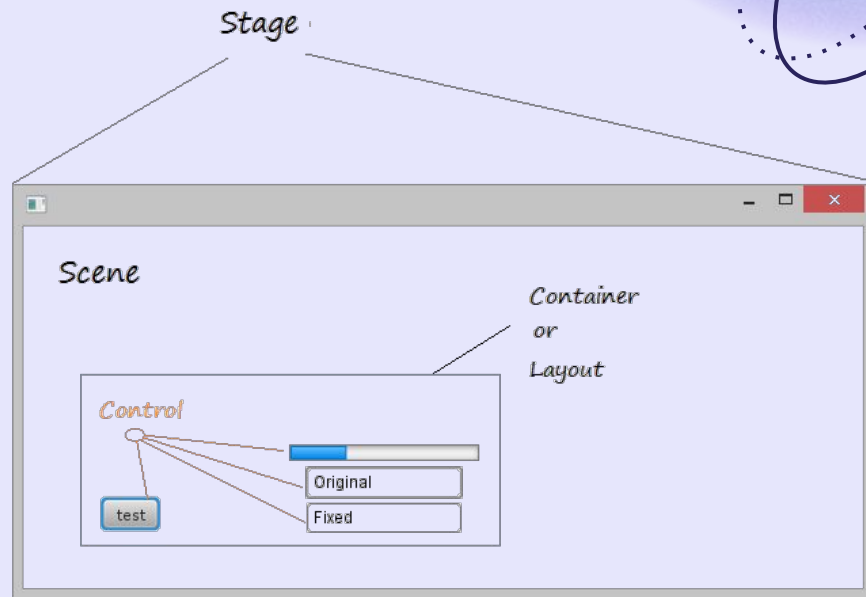
- **Stage** (okno),
- **Scene** (jego wnętrze)

oraz **drzewa kontrolek**.

Stage

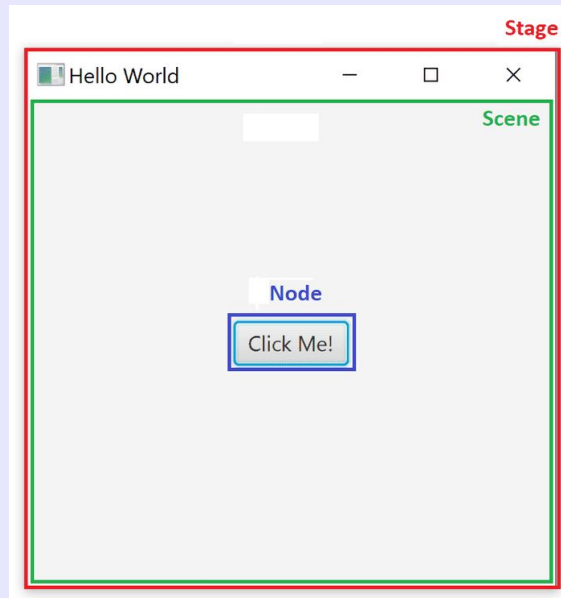
Stage zawiera wszystkie obiekty aplikacji JavaFX.

Utworzony obiekt stage jest przekazywany jako argument do metody start() klasy **Application**.



Scena

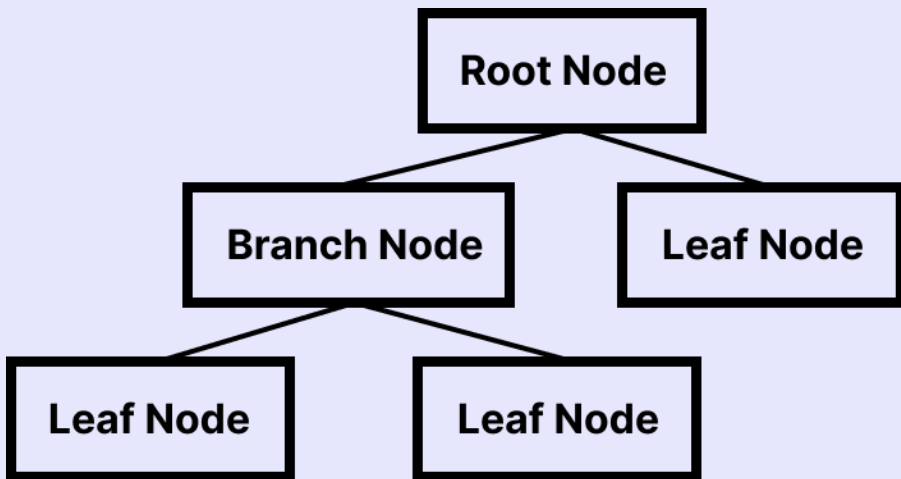
Scena reprezentuje fizyczną zawartość aplikacji JavaFX. Klasa **Scene** z pakietu **javafx.scene** reprezentuje obiekt sceny.



SceneGraph i węzły

SceneGraph to podobna do drzewa struktura danych (hierarchiczna) przedstawiająca zawartość sceny.

Węzeł (node) jest wizualnym/graficznym obiektem SceneGraph.



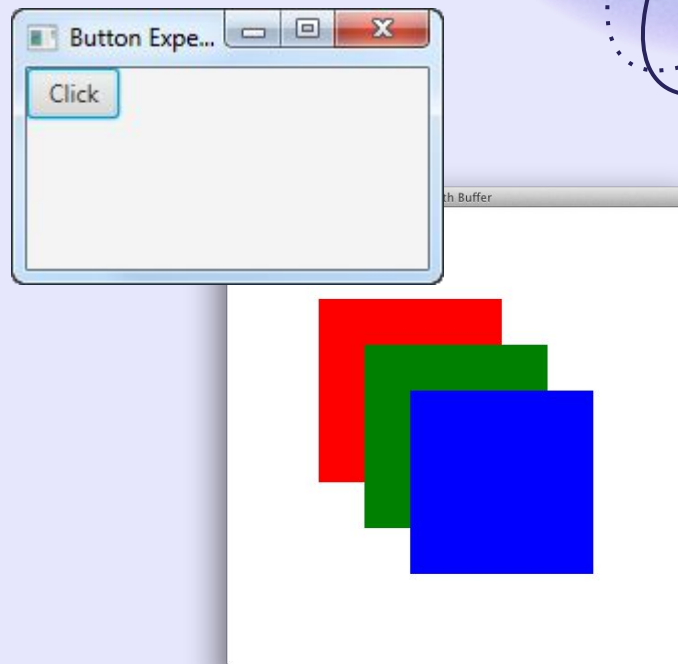
Rodzaje węzłów

Węzeł może zawierać:

- Obiekty geometryczne (graficzne) (2D i 3D)
- Elementy sterujące interfejsu użytkownika
- Kontenery (okienka układu)
- Elementy multimedialne

Możemy je podzielić też na:

- Root Node
- Branch Node
 - Grupa
 - Region
 - WebView
- Leaf Node



Tworzenie aplikacji

Klasa **Application** pakietu `javafx.application` jest punktem wejścia aplikacji w JavaFX. Aby stworzyć aplikację, musimy ją odziedziczyć i zaimplementować jej abstrakcyjną metodę **start()**. W tej metodzie musimy napisać cały kod grafiki JavaFX.

```
public class JavafxExample extends Application {  
    @Override  
    public void start(Stage primaryStage) throws Exception {  
        /*  
         * Kod aplikacji JavaFX.  
         * (Stage, scene, SceneGraph)  
         */  
    }  
    public static void main(String args[]){  
        launch(args);  
    }  
}
```

Metoda `start()`

W ramach metody **`start()`**, aby stworzyć podstawową aplikację JavaFX, musimy wykonać poniższe kroki:

1. Przygotować SceneGraph z wymaganymi node'ami,
2. Przygotować scenę o wymaganych wymiarach i dodać do niej SceneGraph (Root Node),
3. Przygotować stage i dodać scenę do stage'a, wyświetlając na końcu zawartość stage'a.



1. Przygotowanie SceneGraph

Zgodnie z opisanymi krokami, musimy przygotować SceneGraph z wymaganymi węzłami. Ponieważ węzeł główny (Root Node) jest pierwszym węzłem, musimy go utworzyć. Jako węzeł główny możemy wybrać grupę (**Group**), region lub WebView.

```
Group root = new Group();
```



2. Przygotowanie sceny

Scenę możemy utworzyć za pomocą jej klasy z paczki **javafx.scene**. Podczas tworzenia instancji, obowiązkowe jest przekazanie obiektu root do konstruktora.

```
Scene scene = new Scene(root);
```

Możemy także przekazać dwa parametry typu double reprezentujące wysokość i szerokość sceny.

```
Scene scene = new Scene(root, 600, 300);
```



3. Przygotowanie stage'a

Stage to kontener aplikacji JavaFX, który zapewnia okno dla aplikacji. Jest reprezentowany przez klasę Stage pakietu **javafx.stage**. Obiekt tej klasy jest przekazywany jako parametr metody **start()** klasy **Application**.

```
// Ustawianie tytułu stage'a
primaryStage.setTitle("Przykładowa aplikacja");

// Ustawianie sceny w stage'u
primaryStage.setScene(scene);

// Wyświetlenie sceny
primaryStage.show();
```

Cykl życia aplikacji

Klasa JavaFX Application ma trzy metody cyklu życia:

- **start()** – Metoda punktu wejścia, w której ma zostać zapisany kod graficzny JavaFX,
- **stop()** – Pusta metoda, którą można nadpisać, tutaj możemy napisać logikę zatrzymania aplikacji,
- **init()** – Pusta metoda, którą można nadpisać, ale nie można w tej metodzie tworzyć sceny ani stage'a.

Za każdym razem, gdy uruchamiana jest aplikacja JavaFX, wykonywane są następujące czynności

- Zostanie utworzona instancja klasy aplikacji.
- Wywoływana jest metoda **init()**.
- Wywoływana jest metoda **start()**.
- Launcher czeka na zakończenie działania aplikacji i wywołuje metodę **stop()**.



Zamykanie aplikacji

Po zamknięciu ostatniego okna aplikacji aplikacja JavaFX jest zamykana **niejawnie**. Możemy wyłączyć to zachowanie, przekazując „**False**” do statycznej metody **setImplicitExit()**, która powinna być wywoływana z kontekstu statycznego.

Aplikację JavaFX można zakończyć jawnie za pomocą metod **Platform.exit()** lub **System.exit(int)**.



Tworzenie tekstu

Możemy utworzyć tekst, tworząc instancję klasy **Text** pakietu **javafx.scene.text** w następujący sposób:

```
Text text = new Text();
```

Klasa **Text** zawiera właściwość o nazwie **text** typu **string**, która reprezentuje tekst, który ma zostać utworzony. Po utworzeniu instancji klasy **Text** należy ustawić wartość tej właściwości za pomocą metody **setText()**:

```
String text = "Przykładowy tekst";  
Text.setText(text);
```

Możemy także ustawić pozycję tekstu, określając wartości właściwości **x** i **y** przy użyciu ich odpowiednich metod ustawiających, a mianowicie **setX()** i **setY()**:

```
text.setX(50);  
text.setY(50);
```



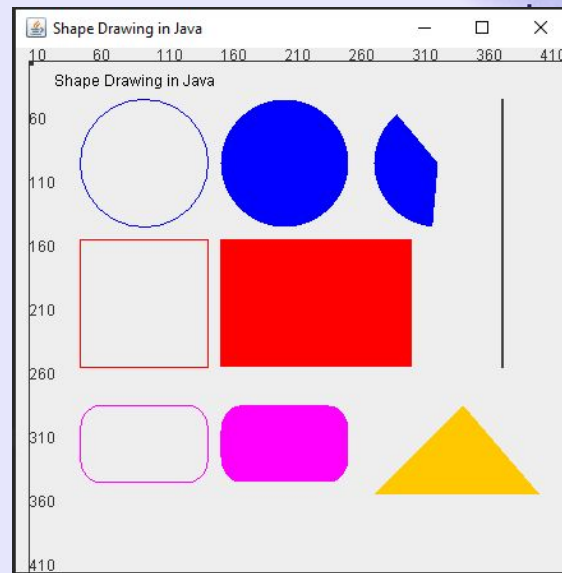
Tworzenie tekstu



Obsługa kształtów

Korzystając z biblioteki JavaFX, możemy rysować:

- Kształty, takie jak prostokąt, okrąg, elipsa, wielokąt itp.
- Elementy ścieżki, takie jak linia, linia pozioma, linia pionowa, krzywa sześcienna itp.
- Oprócz tego możemy również narysować kształt 2D, analizując ścieżkę SVG.



Tworzenie kształtów

By utworzyć kształt, musimy:

- Utworzyć wystąpienie odpowiedniej klasy o wymaganym kształcie,

```
Line line = new Line();
```

- Ustawić właściwości kształtu,

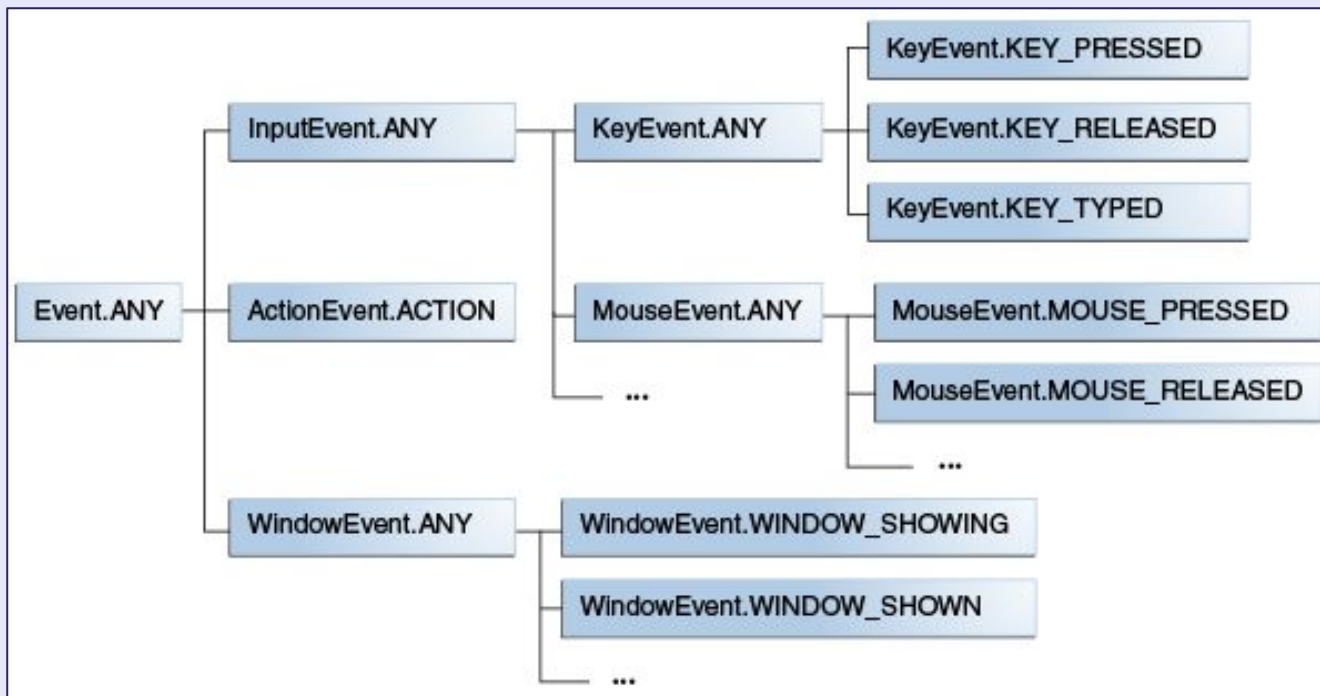
```
line.setStartX(150.0f);  
line.setStartY(140.0f);  
line.setEndX(450.0f);  
line.setEndY(140.0f);
```

- Dodać obiekt kształtu do grupy.

```
Group root = new Group(line);
```



Obsługa zdarzeń



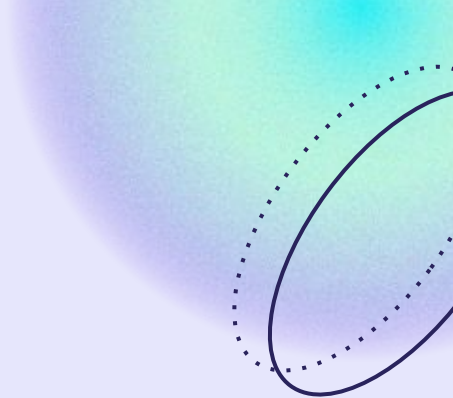
Obsługa zdarzeń

Zdarzenia można ogólnie podzielić na dwie następujące kategorie:

- Zdarzenia pierwszoplanowe (foreground events),
- Zdarzenia w tle (background events)

JavaFX posiada również mechanizm znany jako **procedura obsługi zdarzeń**. Pozwala nam napisać kod który jest wykonywany po wystąpieniu zdarzenia. W JavaFX każde zdarzenie ma:

- Target (Cel),
- Source (Źródło),
- Type (Typ)



Obsługa zdarzeń - przykład

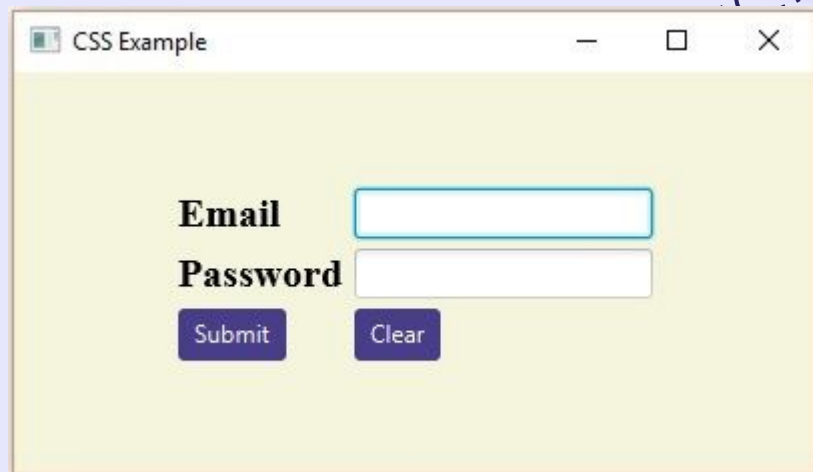
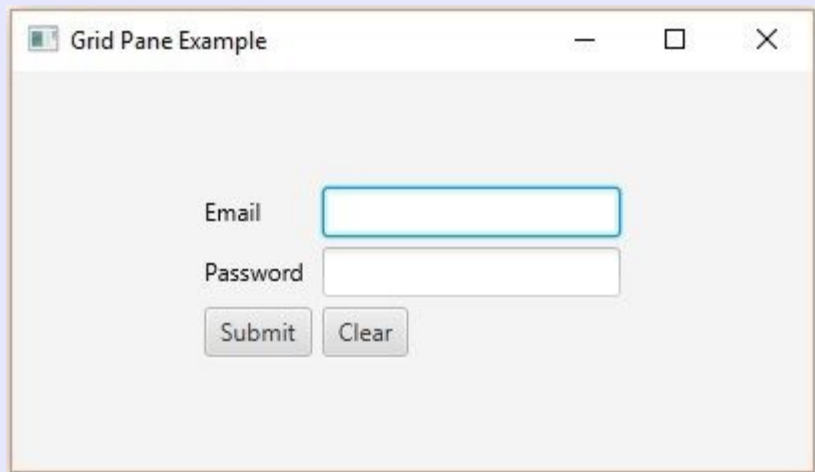


Obsługa zdarzeń - kod

```
// Tworzenie funkcji obsługującej zdarzenie
EventHandler<MouseEvent> eventHandler = new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent e) {
        System.out.println("Przykładowa wiadomość");
        circle.setFill(Color.DARKSLATEBLUE);
    }
};

// Dodawanie filtru
Circle.addEventFilter(MouseEvent.MOUSE_CLICKED, eventHandler);
```

CSS



```
Scene scene = new Scene(new Group(), 500, 400);  
scene.getStylesheets().add("path/styleSheet.css");
```

```
button1.setStyle(<kod CSS>);  
button2.setStyle(<kod CSS>);
```

FXML

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

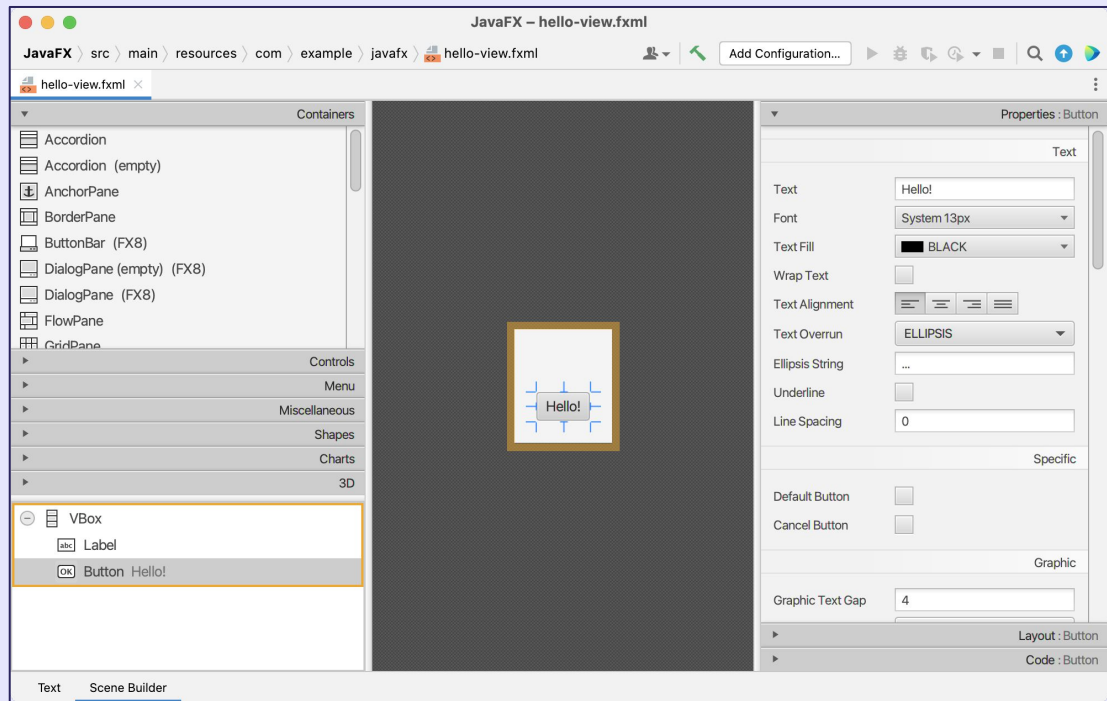
<VBox>
  <children>
    <Label text="Hello world FXML"/>
  </children>
</VBox>
```

Ładowanie pliku FXML

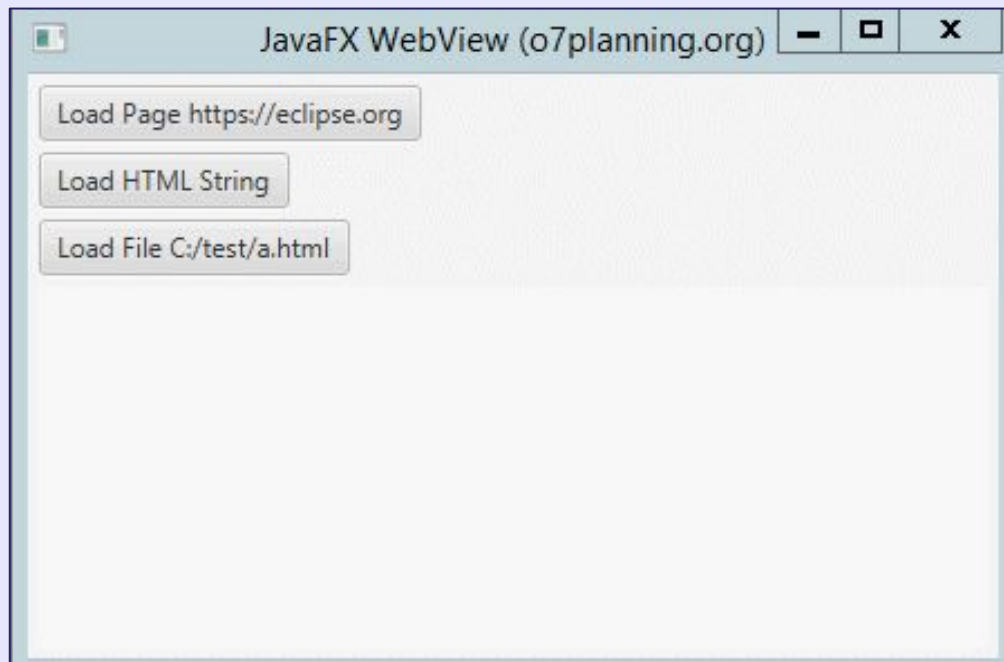
```
@Override
public void start(Stage primaryStage) throws Exception {
    FXMLLoader loader = new FXMLLoader();
    loader.setLocation(new URL("file:///C:/data/hello-world.fxml"));
    VBox vbox = loader.<VBox>load();

    Scene scene = new Scene(vbox);
    primaryStage.setScene(scene);
    primaryStage.show();
}
```

SceneBuilder



WebView



Interesujące źródła nauki

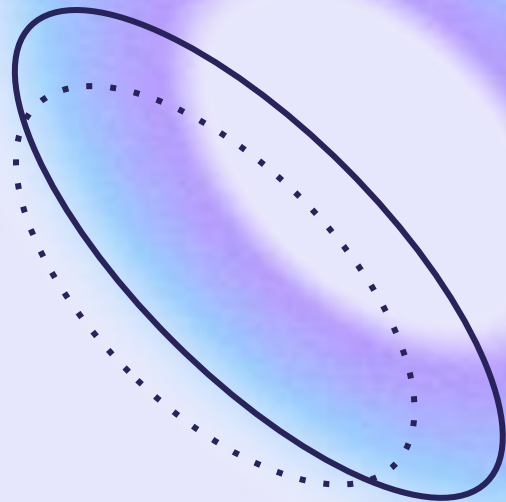
- **Tutorialspoint (ENG)** - popularne źródło nauki zawierające wiele dokładnych informacji
<https://www.tutorialspoint.com/javafx/>
- **Jenkov (ENG)** - portal z kursami o różnych narzędziach, głównie opartych na Javie
<https://jenkov.com/tutorials/javafx/>
- **Bro Code (ENG)** - 4-godzinny film prezentujący wszystkie najważniejsze założenia JavaFX
https://www.youtube.com/watch?v=9XJicRt_Fal
- **Zaczniij Programować (PL)** - seria filmów na temat JavaFX w języku polskim
<https://www.youtube.com/watch?v=rEzzJaoC6uo>
- **Dokumentacja (ENG)** - oficjalna dokumentacja JavaFX
<https://openjfx.io/openjfx-docs/>



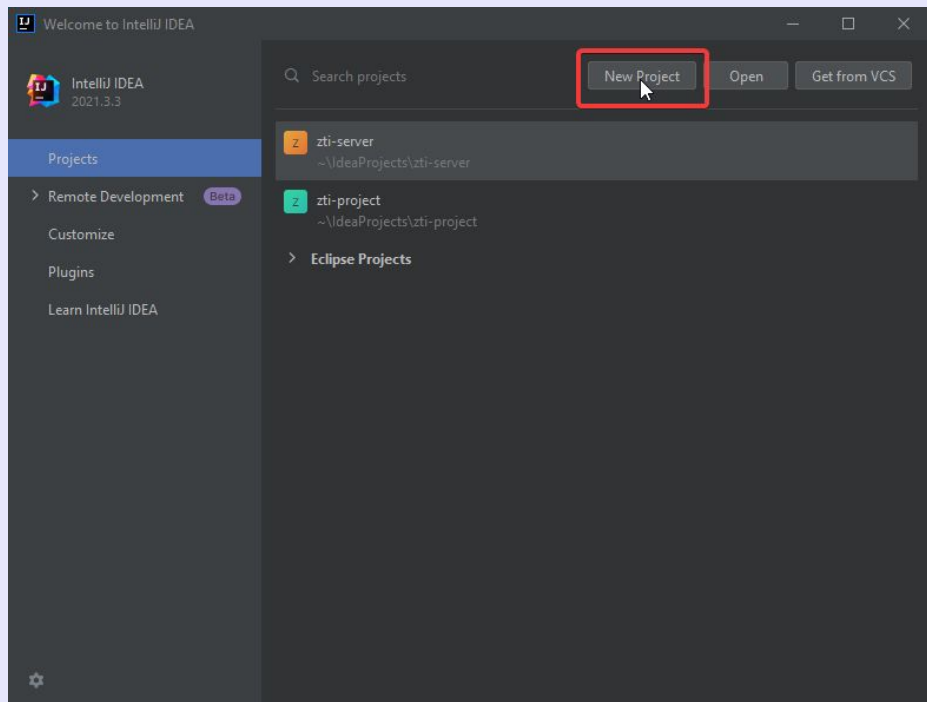
02.

Instrukcja

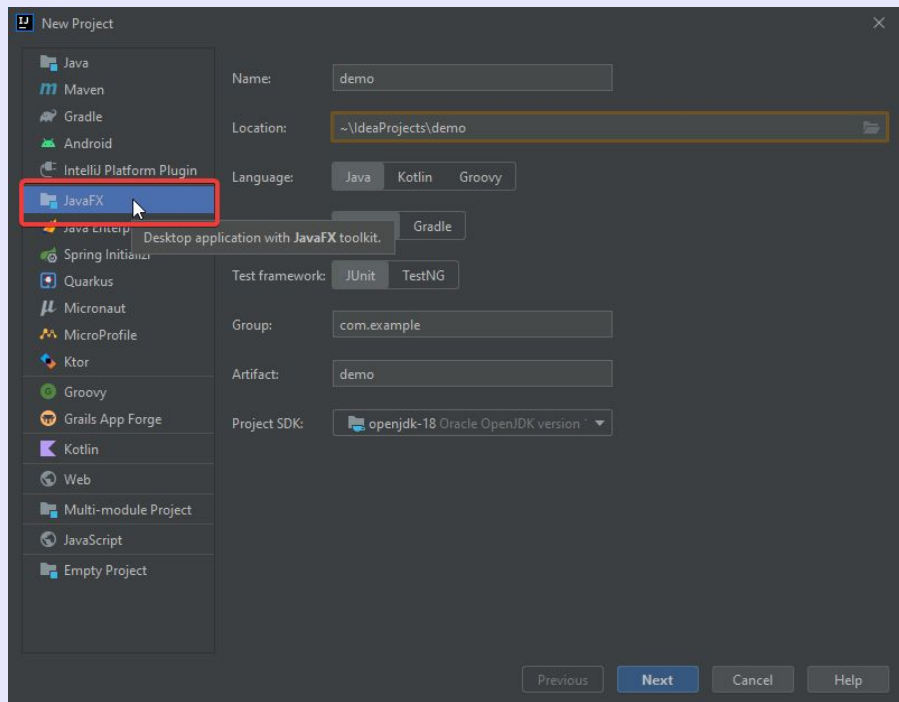
Jak skonfigurować środowisko? (IntelliJ)



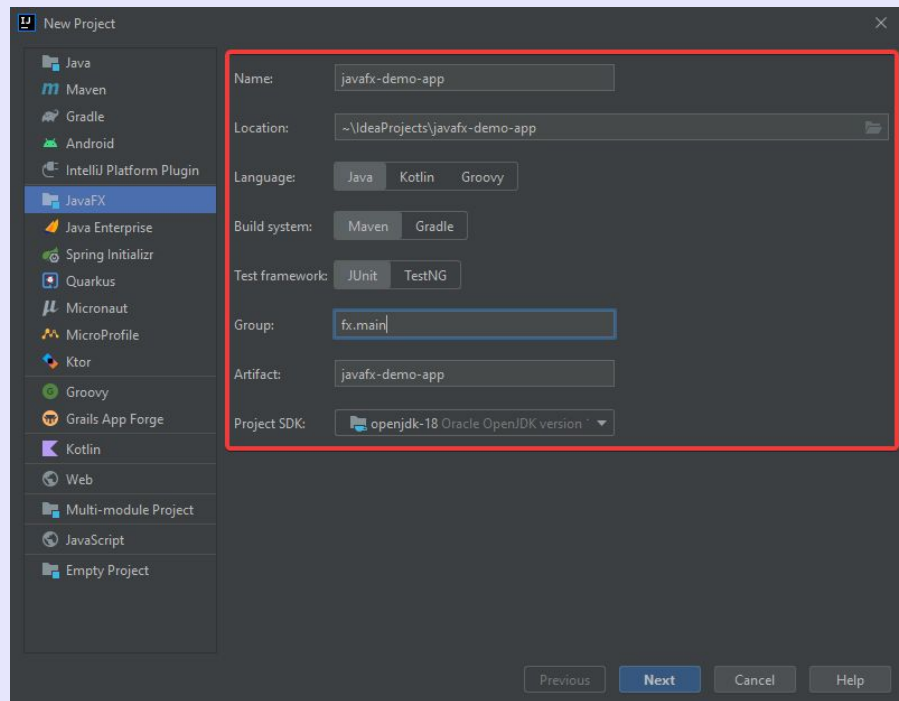
1. Tworzymy nowy projekt



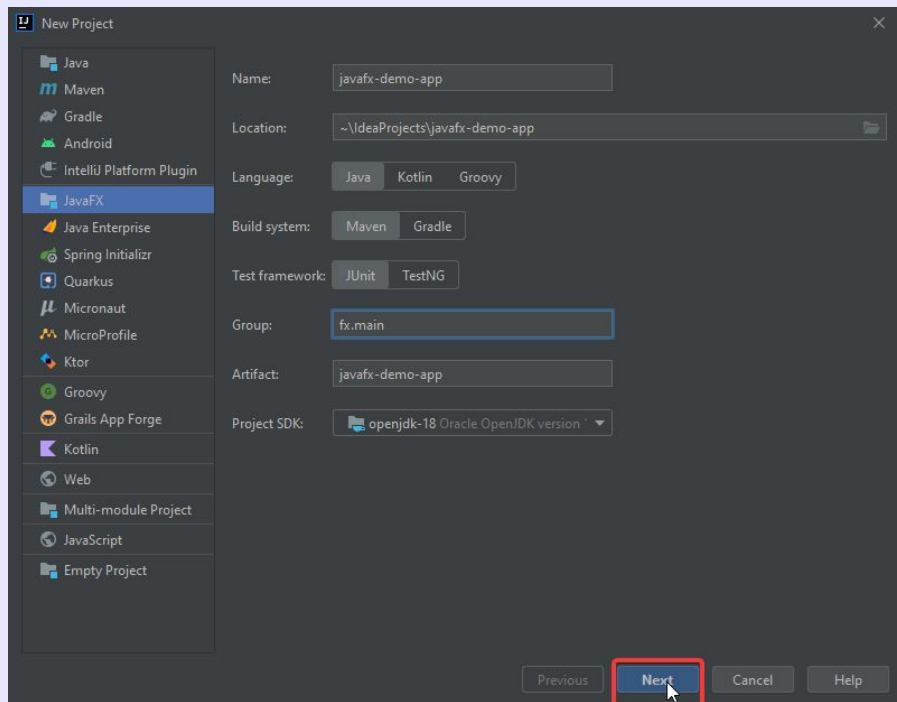
2. Wybieramy JavaFX



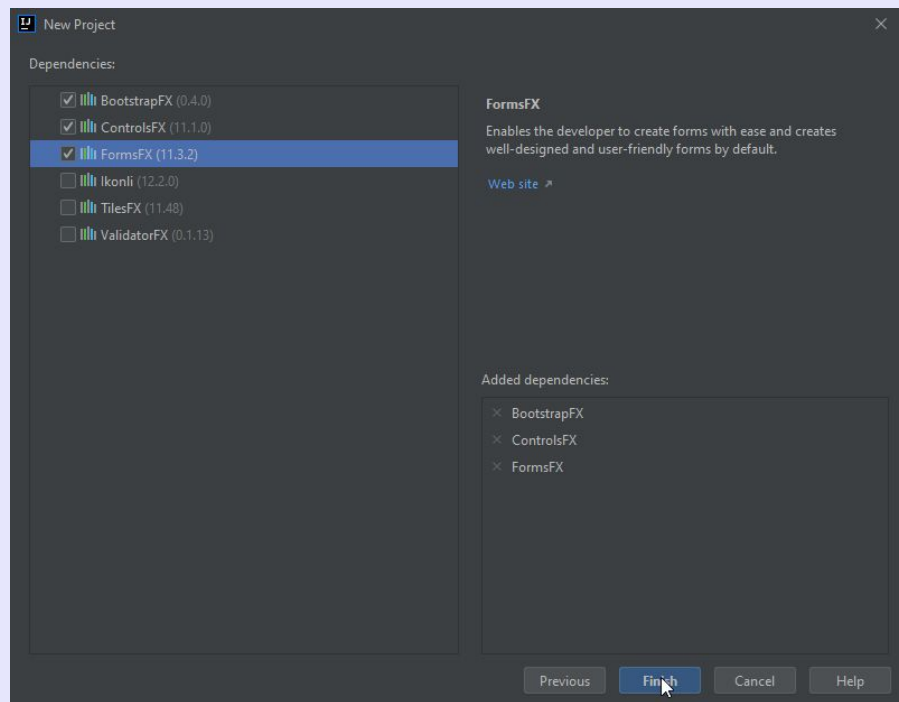
3. Wypełniamy dane



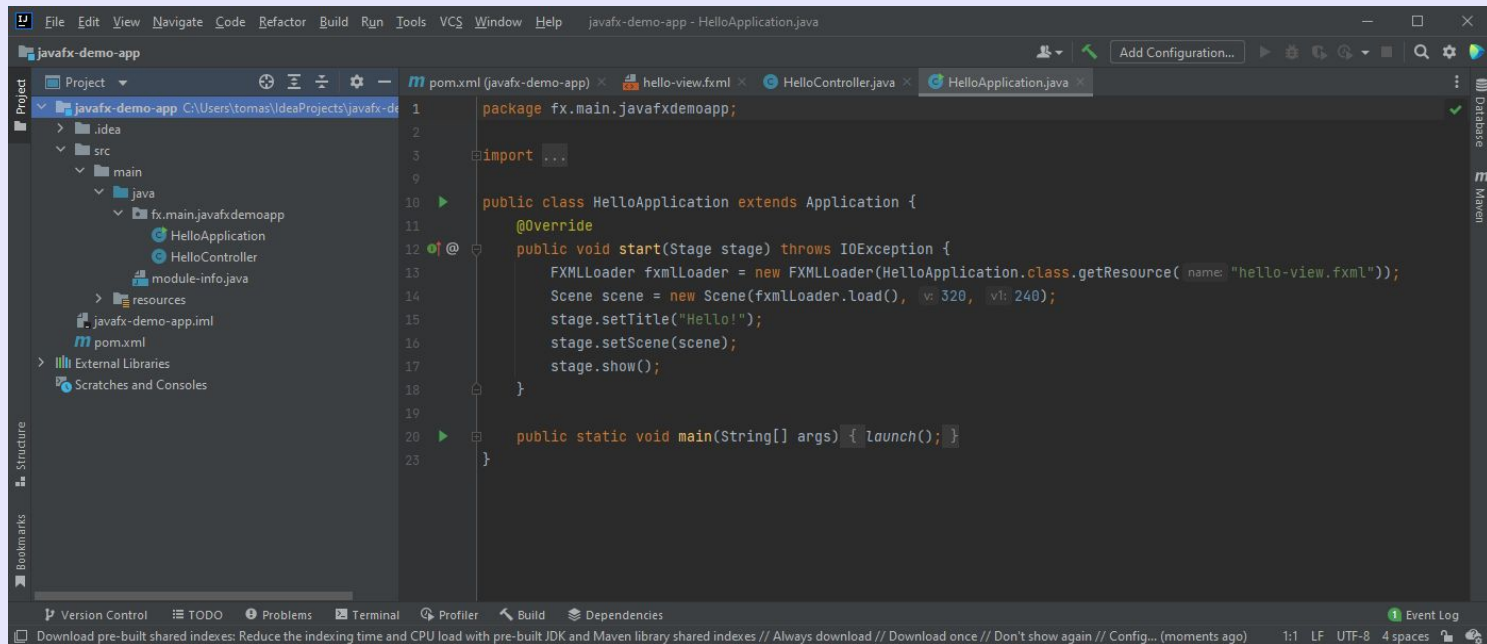
4. Klikamy “Next”



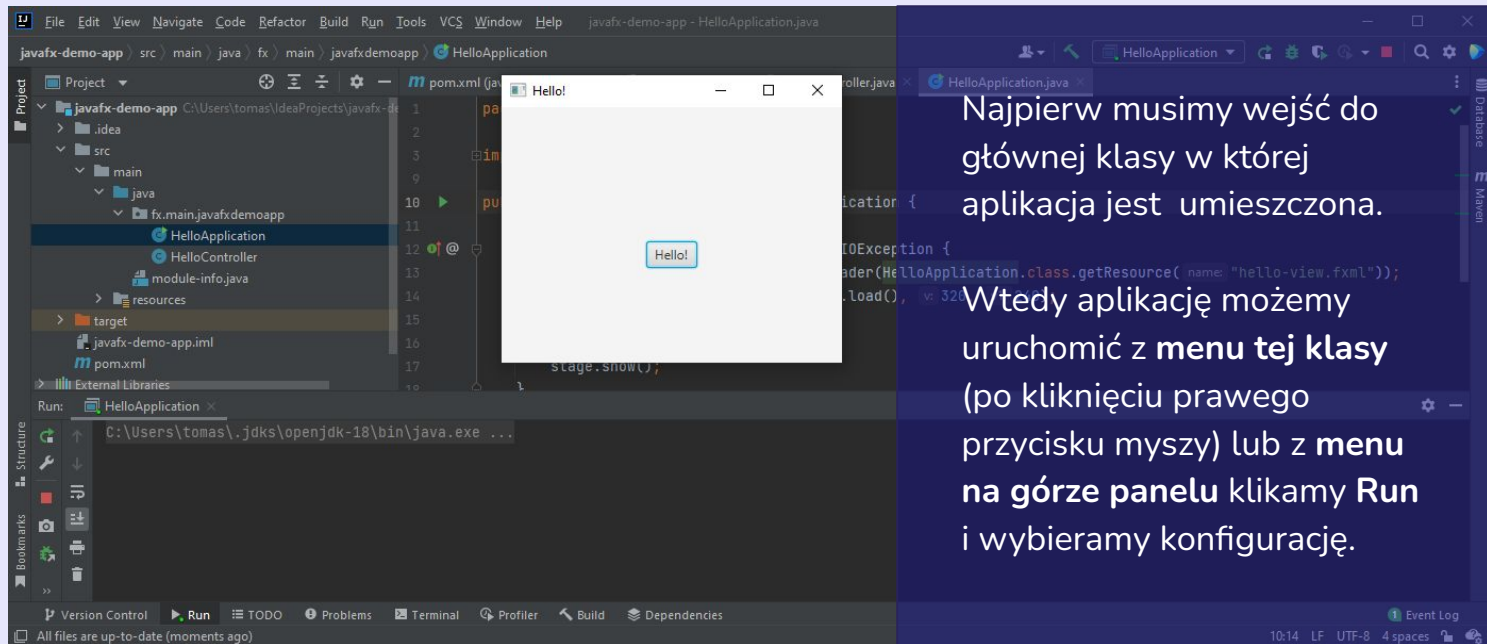
5. Wybieramy zależności



6. Projekt jest gotowy



7. Uruchomienie

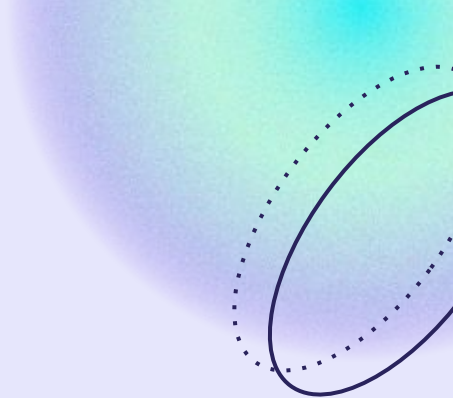


Najpierw musimy wejść do głównej klasy w której aplikacja jest umieszczona.

Wtedy aplikację możemy uruchomić z **menu tej klasy** (po kliknięciu prawego przycisku myszy) lub z **menu na górze panelu** klikamy **Run** i wybieramy konfigurację.

Inne konfiguracje

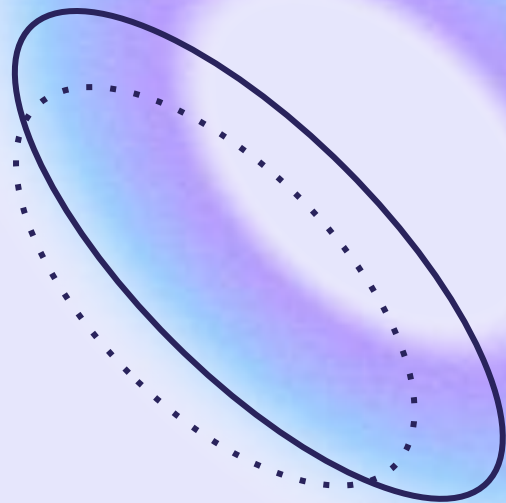
- Eclipse - Tutorialkart (ENG)
<https://www.tutorialkart.com/javafx/install-javafx-in-eclipse-ide/>
- IntelliJ - Bro Code (ENG)
<https://www.youtube.com/watch?v=Ope4icw6bVk>
- Eclipse - Java Coding Community (ENG)
<https://www.youtube.com/watch?v=bC4XB6JAaoU>
- Eclipse - Zaczniij Programować (PL)
https://www.youtube.com/watch?v=qcDNZB_5rgc



03.

Przykłady

Wykorzystanie JavaFX w praktyce.



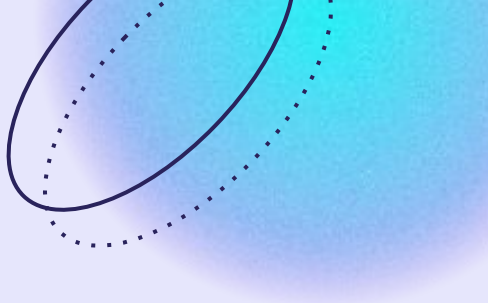
Dzięki za uwagę!

Czy macie jakieś pytania?

Repozytorium z materiałami:

<https://github.com/nerooc/zti-seminar-javafx>

Piotr Harmuszkiewicz, Tomasz Gajda



Źródła

- <https://www.tutorialspoint.com/javafx/index.htm>
- <https://openifx.io/>
- <https://www.javatpoint.com/javafx-tutorial>
- <https://jenkov.com/tutorials/javafx/index.html>
- <https://zetcode.com/gui/javafx/events/>
- https://www.youtube.com/watch?v=9XJicRt_Fal
- <https://www.youtube.com/watch?v=rEzzJaoC6uo>