1. **Tạo chương trình javaFX có file FXML và sử dụng Scene builder trong netbeans.**

* Vào new project chọn javaFX và tạo ra JavaFX FXML application.

1. **Mô hình MVVM trong javafx.**

* Model: là phần mà chúng ta sử dụng để xử lí dữ liệu, lấy dữ liệu từ database hoặc cập nhật dữ liệu, xử lí dữ liệu.
* View: Phần mà người dùng nhìn thấy và thao tác trên đó.
* View-Model: phần trung gian giữa model và view. Xử lí các events xảy ra trong view của người dùng, giao tiếp với model để lấy dữ liệu lên hoặc cập nhật dữ liệu sang cho model.

1. **Cấu trúc đơn giản của một phần model**

public void start(Stage primaryStage) {

try {

Parent root = FXMLLoader.*load*(getClass().getResource("/Main.fxml"));

Scene scene = new Scene(root);

primaryStage.setScene(scene);

primaryStage.show();

} catch (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

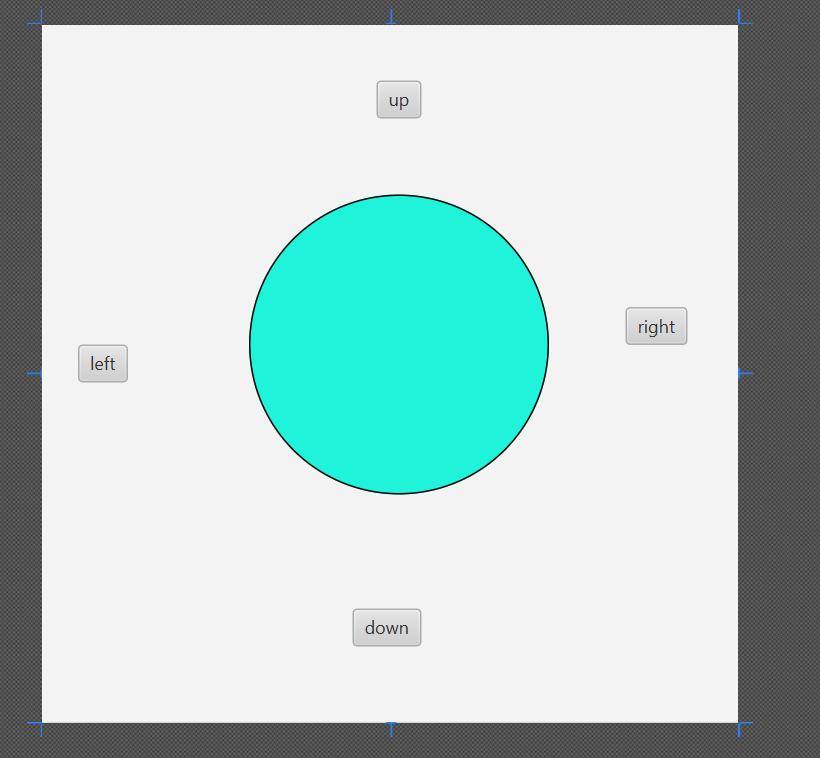
}

}

* Trong ví dụ này thì chúng ta không cần phải tạo root bằng một nhóm hay gì hết, mà root hay toàn bộ nội dung cho sence sẽ được sence builder lo hết. Chúng ta chỉ cần add nội dung từ sence builder vào trong root là được.

1. **Tạo giao diện kéo thả với scene builder.**

* Bước 1: tìm đường dẫn đến file .fxml và mở nó với ứng dụng scene builder.
* Bước 2: thực hiện các thao tác cài đặt vị trí của các controll cho nó.

****

(Trong ví dụ này tạo 3 nút và 1 hình tròn.)

* Tất cả sự thay đổi trên scene builder đều được đồng bộ ở trên file fxml. Và đặc bietj chỉ được thay đổi trên scene builder không được thay đổi ở trên file .fxml. File .fxml được theo dõi nhằm mục đích hiểu được cách tổ chức và sử dụng tên các thuộc tính.

1. **Xử lí sự kiện khi chúng ta nhấn vào nút nào thì in ra hướng di chuyển.**

* Đầu tiên cần phải thêm file controller (view-model) vào bên trong scene builder. (Trong thanh công cụ ở bên trái chọn phần controller).
* Nhấp vào một đối tượng bên thanh công cụ bên phải chọn phần code. Trong phần này sẽ hiển thị các sự kiện được áp dụng cho đối tượng. Chúng ta chọn một sự kiện ví dụ như (di chuột, kéo, nhấp chuột,…) và chọn một hàm trong controller để xử lí cho sự kiện này. (phần hàm trong controller được nói đến ở phần bên dưới.)

1. **Định nghĩa hàm trong controller.**

* Trong lớp được chọn là controller, chúng ta định nghĩa các hàm để xử lí sự kiện cho các controlls. (Các controlls sẽ dựa vào các hàm này mà sự kiện sẽ được xảy ra).

1. **Đồng bộ đối tượng từ scenebuilder và controller.**

JavaFX và Scene builder cung cấp cơ chế để controller làm việc với các đối tượng trong scene builder về việc thay đổi kích thước, kiểu, vị trí,… một cách trực tiếp.

* B1: khai báo đối tượng cần xử lí ở bên trong controller.

Ví dụ: trong scene builder có một 3 shapes(hình tròn, hình chữ nhât, hình vuông). Chúng ta muốn thay đổi 3 hình thì khai báo cả 3 trong controller hoặc hai hình thì khai báo 2 trong controller.

* B2: đặt fx-id của đối tượng trong scene builder bằng với đối tượng đã được khai báo trong controller.

(nhấp vào đối tượng được chọn, trong thanh công cụ bên phải chọn fx-id để chọn tên đối tượng được khai báo trong controller.)

Sau khi đã hoàn thành xong thì bất cứ khi nào đối tượng trong controller được kích hoạt thay đổi thì đối tượng scene builder tham chiếu tới nó cũng bị thay đổi theo.

1. **Trick lỏ khi chạy file.**

* Debug toàn bộ project trước.
* Sau khi debug toàn bộ project thì chúng ta sẽ run file model để chạy

**VD: tạo chương trình có bốn nút, nhấn nút nào thì shape sẽ di chuyển theo hướng đó.**

**A screenshot of a computer

Description automatically generated**

1. **Model.**

|  |
| --- |
| **/\***  **\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license**  **\* Click nbfs://nbhost/SystemFileSystem/Templates/javafx/FXML.java to edit this template**  **\*/**  **package javafxapplication5;**  **import javafx.application.Application;**  **import javafx.fxml.FXMLLoader;**  **import javafx.scene.Parent;**  **import javafx.scene.Scene;**  **import javafx.stage.Stage;**  **/\*\***  **\***  **\* @author user**  **\*/**  **public class JavaFXApplication5 extends Application {**    **@Override**  **public void start(Stage stage) throws Exception {**  **Parent root = FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"));**  **Scene scene = new Scene(root);**  **// String css = this.getClass().getResource("css.css").toExternalForm();**  **// scene.getStylesheets().add(css);**    **stage.setScene(scene);**  **stage.show();**  **}**  **/\*\***  **\* @param args the command line arguments**  **\*/**  **public static void main(String[] args) {**  **launch(args);**  **}**    **}** |

1. **View**

|  |
| --- |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<?import javafx.scene.control.Button?>**  **<?import javafx.scene.layout.AnchorPane?>**  **<?import javafx.scene.shape.Circle?>**  **<AnchorPane id="AnchorPane" prefHeight="465.0" prefWidth="464.0" style="-fx-background-color: white;" xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1" fx:controller="javafxapplication5.FXMLDocumentController">**  **<children>**  **<Button layoutX="223.0" layoutY="37.0" mnemonicParsing="false" onAction="#up" text="up" />**  **<Button layoutX="24.0" layoutY="213.0" mnemonicParsing="false" onAction="#left" text="left" />**  **<Button layoutX="207.0" layoutY="389.0" mnemonicParsing="false" onAction="#down" text="down" />**  **<Button layoutX="389.0" layoutY="188.0" mnemonicParsing="false" onAction="#right" text="right" />**  **<Circle fx:id="mycircle" fill="#21ffe4" layoutX="238.0" layoutY="213.0" radius="100.0" stroke="BLACK" strokeType="INSIDE" />**  **<Circle fill="DODGERBLUE" layoutX="167.0" layoutY="238.0" radius="100.0" stroke="BLACK" strokeType="INSIDE" />**  **</children>**  **</AnchorPane>** |

1. **View Model.**

|  |
| --- |
| **/\***  **\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license**  **\* Click nbfs://nbhost/SystemFileSystem/Templates/javafx/FXML2.java to edit this template**  **\*/**  **package javafxapplication5;**  **import java.net.URL;**  **import java.util.ResourceBundle;**  **import javafx.event.ActionEvent;**  **import javafx.fxml.FXML;**  **import javafx.fxml.Initializable;**  **import javafx.scene.control.Label;**  **import javafx.scene.shape.Circle;**  **/\*\***  **\***  **\* @author user**  **\*/**  **public class FXMLDocumentController implements Initializable {**    **@FXML**  **private Circle mycircle;**  **private double x;**  **private double y;**  **@FXML**  **private Label label;**    **@FXML**  **public void up(ActionEvent e)**  **{**  **mycircle.setCenterY(y = y - 1);**  **}**  **@FXML**  **public void right(ActionEvent e)**  **{**  **mycircle.setCenterX(x = x + 1);**  **}**  **@FXML**  **public void down(ActionEvent e)**  **{**  **mycircle.setCenterY(y = y + 1);**  **}**  **@FXML**  **public void left(ActionEvent e)**  **{**  **mycircle.setCenterX(x = x - 1);**  **}**    **@Override**  **public void initialize(URL url, ResourceBundle rb) {**  **// TODO**  **}**    **}** |