Catedra Calculatoare

Raport

PCD

Lucrarea de laborator nr.5

**Tema: Animatia scenelor cu ajutorul firelor de executie**

Varianta 7

A îndeplinit:st.gr. C-161 Marjina Alexandru

A controlat: lector superior Rotaru Lilia

**Chișinău-2018**

**Probleme propuse:** Spot publicitar, ce poate fi întrerupt şi repornit, ce caracterizează FCIM;

**Listingul programului:**

**Main:**

package sample;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
public class Main extends Application {  
  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
 Parent root = FXMLLoader.*load*(getClass().getResource("sample.fxml"));  
 primaryStage.setTitle("Hello World");  
 primaryStage.setScene(new Scene(root, 600, 600));  
 primaryStage.show();  
 }  
  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

**Controller:**

//package sample;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
import java.util.concurrent.CountDownLatch;  
import java.util.concurrent.TimeUnit;  
  
import javafx.animation.\*;  
import javafx.application.Platform;  
import javafx.concurrent.Service;  
import javafx.concurrent.Task;  
import javafx.event.ActionEvent;  
import javafx.event.EventHandler;  
import javafx.fxml.FXML;  
import javafx.geometry.Bounds;  
import javafx.scene.control.Button;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.shape.Sphere;  
import javafx.scene.text.Text;  
import javafx.scene.transform.Rotate;  
import javafx.scene.transform.Scale;  
import javafx.scene.transform.Translate;  
import javafx.util.Duration;  
import org.omg.CORBA.TIMEOUT;  
  
public class Controller {  
  
 @FXML  
 private ResourceBundle resources;  
  
 @FXML  
 private URL location;  
  
  
 @FXML  
 private Text tx\_FCIM;  
  
 @FXML  
 private Sphere tx\_ball;  
  
 @FXML  
 private Text tx\_UTM;  
  
 @FXML  
 private Text tx\_Calc;  
  
 @FXML  
 private Text tx\_Info;  
  
 @FXML  
 private Text tx\_MN;  
  
 @FXML  
 private Button btn\_Start;  
  
 @FXML  
 private Button btn\_Pause;  
  
 @FXML  
 private AnchorPane id\_pane;  
 public boolean paused=false;  
  
 public void pauseThread(){  
 paused=true;  
  
 }  
 public void resumeThread(){  
 paused=false;  
 }  
  
 @FXML  
 void initialize() {  
  
 Service<Void> service1=new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected synchronized Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 boolean paused=false;  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 Timeline time=new Timeline();  
 time.setCycleCount(Timeline.*INDEFINITE*);  
 time.setAutoReverse(true);  
 KeyValue kv1=new KeyValue(tx\_FCIM.xProperty(),200);  
 KeyFrame kf=new KeyFrame(Duration.*millis*(1000),kv1);  
 time.getKeyFrames().add(kf);  
// try {  
// Thread.sleep(1000);  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
 time.play();  
 } finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 Service<Void> service2=new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 Timeline time = new Timeline(new KeyFrame(Duration.*seconds*(0.1),  
 new EventHandler<ActionEvent>() {  
 @Override  
 public void handle(ActionEvent event) {  
 tx\_UTM.getTransforms().add(new Rotate(5,0,0));  
 }  
 }));  
 time.setCycleCount(Timeline.*INDEFINITE*);  
 time.play();  
 while (paused){  
 try {  
 this.wait();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 Service<Void> service3=new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 Timeline time=new Timeline();  
 time.setCycleCount(Timeline.*INDEFINITE*);  
 time.setAutoReverse(true);  
 KeyValue kv1=new KeyValue(tx\_Calc.yProperty(),250);  
 KeyFrame kf=new KeyFrame(Duration.*millis*(1000),kv1);  
 time.getKeyFrames().add(kf);  
 time.play();  
 }  
 finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 Service<Void> service4=new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 Timeline time=new Timeline();  
 time.setCycleCount(Timeline.*INDEFINITE*);  
 time.setAutoReverse(true);  
 KeyValue kv1=new KeyValue(tx\_Info.xProperty(),-100);  
 KeyValue kv2=new KeyValue(tx\_Info.yProperty(),100);  
 KeyFrame kf=new KeyFrame(Duration.*millis*(1000),kv1,kv2);  
 time.getKeyFrames().add(kf);  
 time.play();  
 while (paused){  
 try {  
 Thread.*sleep*(20);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 Service<Void> service5= new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 FadeTransition fadeTransition = new FadeTransition(Duration.*seconds*(0.1), tx\_MN);  
 fadeTransition.setFromValue(1.0);  
 fadeTransition.setToValue(0.0);  
 fadeTransition.setCycleCount(Animation.*INDEFINITE*);  
 fadeTransition.play();  
 }  
 finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 Service<Void> service6= new Service<Void>() {  
 @Override  
 protected Task<Void> createTask() {  
 return new Task<Void>() {  
 @Override  
 protected Void call() throws Exception {  
 CountDownLatch latch =new CountDownLatch(1);  
 Platform.*runLater*(new Runnable() {  
 @Override  
 public void run() {  
 try {  
 FadeTransition fadeTransition = new FadeTransition(Duration.*seconds*(0.1), tx\_ball);  
 fadeTransition.setFromValue(1.0);  
 fadeTransition.setToValue(0.0);  
 fadeTransition.setCycleCount(Animation.*INDEFINITE*);  
 Timeline time=new Timeline();  
 time.setCycleCount(Timeline.*INDEFINITE*);  
 time.setAutoReverse(true);  
 KeyValue kv1=new KeyValue(tx\_ball.radiusProperty(),-50);  
 KeyValue kv2=new KeyValue(tx\_ball.translateXProperty(),500);  
 KeyFrame kf=new KeyFrame(Duration.*millis*(1000),kv1,kv2);  
 time.getKeyFrames().add(kf);  
 fadeTransition.play();  
 time.play();  
  
 }  
 finally {  
 latch.countDown();  
 }  
 }  
 });  
 latch.await();  
 return null;  
 }  
 };  
 }  
 };  
 btn\_Start.setOnAction(event -> {  
 // notifyAll();  
 resumeThread();  
 service1.restart();  
 service2.restart();  
 service3.restart();  
 service4.restart();  
 service5.restart();  
 service6.restart();  
 System.*out*.println("A fost apasat start");  
 });  
 btn\_Pause.setOnAction(event -> {  
 pauseThread();  
 service1.cancel();  
 service4.cancel();  
 System.*out*.println("A fost apasat pause" );  
 // time.stop();  
 });  
  
 }  
}

**Sample.fxml:**

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.TitledPane?>  
<?import javafx.scene.effect.Light.Distant?>  
<?import javafx.scene.effect.Lighting?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.paint.RadialGradient?>  
<?import javafx.scene.paint.Stop?>  
<?import javafx.scene.shape.Sphere?>  
<?import javafx.scene.text.Font?>  
<?import javafx.scene.text.Text?>  
  
<TitledPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="600.0" prefWidth="600.0" text="Laborator5" textFill="#0b40ab" xmlns="http://javafx.com/javafx/8.0.141" xmlns:fx="http://javafx.com/fxml/1" fx:controller="Controller">  
 <content>  
 <AnchorPane fx:id="id\_pane" prefHeight="200.0" prefWidth="200.0" style="-fx-background-color: #835eff;">  
 <children>  
 <Text fx:id="tx\_FCIM" fill="#1e2680" layoutX="181.0" layoutY="342.0" strokeType="OUTSIDE" strokeWidth="0.0" text="FCIM" wrappingWidth="202.13671875">  
 <font>  
 <Font size="71.0" />  
 </font>  
 <effect>  
 <Lighting bumpInput="$null" diffuseConstant="0.0" specularConstant="1.74" specularExponent="18.14" surfaceScale="1.03">  
 <light>  
 <Light.Distant color="#b568ad" />  
 </light>  
 </Lighting>  
 </effect>  
 </Text>  
 <Sphere fx:id="tx\_ball" layoutX="80.0" layoutY="490.0" radius="50">  
 <effect>  
 <Lighting bumpInput="$null" diffuseConstant="1.05">  
 <light>  
 <Light.Distant azimuth="33.49" color="#141f9cca" />  
 </light>  
 </Lighting>  
 </effect>  
 </Sphere>  
 <Text fx:id="tx\_UTM" fill="#d70000d6" fontSmoothingType="LCD" layoutX="431.0" layoutY="118.0" strokeType="OUTSIDE" strokeWidth="0.0" text="UTM" wrappingWidth="135.306640625">  
 <font>  
 <Font size="63.0" />  
 </font>  
 </Text>  
 <Text fx:id="tx\_Calc" fill="#0dda1e" fontSmoothingType="LCD" layoutX="30.0" layoutY="166.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Calculatoare">  
 <font>  
 <Font name="Microsoft Himalaya" size="48.0" />  
 </font>  
 </Text>  
 <Text fx:id="tx\_Info" fontSmoothingType="LCD" layoutX="431.0" layoutY="186.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Informatica">  
 <font>  
 <Font name="Microsoft Himalaya" size="41.0" />  
 </font>  
 <fill>  
 <RadialGradient centerX="0.7111111111111111" centerY="0.6523809523809524" radius="0.5">  
 <stops>  
 <Stop color="#0fe11d" />  
 <Stop color="WHITE" offset="1.0" />  
 </stops>  
 </RadialGradient>  
 </fill>  
 </Text>  
 <Text fx:id="tx\_MN" layoutX="36.0" layoutY="242.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Microelectronica">  
 <font>  
 <Font name="Arial Narrow Bold Italic" size="41.0" />  
 </font>  
 <fill>  
 <RadialGradient centerX="0.8277777777777777" centerY="0.4095238095238095" radius="0.5">  
 <stops>  
 <Stop color="BLACK" />  
 <Stop color="#f84c4c" offset="1.0" />  
 </stops>  
 </RadialGradient>  
 </fill>  
 </Text>  
 <Button fx:id="btn\_Start" layoutX="14.0" layoutY="14.0" mnemonicParsing="false" prefHeight="32.0" prefWidth="60.0" style="-fx-background-color: #42f44b;" text="Start">  
 <font>  
 <Font name="System Italic" size="14.0" />  
 </font>  
 </Button>  
 <Button fx:id="btn\_Pause" layoutX="100.0" layoutY="13.0" mnemonicParsing="false" prefHeight="34.0" prefWidth="60.0" style="-fx-background-color: #f47a41;" text="Pause">  
 <font>  
 <Font size="14.0" />  
 </font>  
 </Button>  
 </children>  
 </AnchorPane>  
 </content>  
</TitledPane>

**Output:**





