Department of Computer Science & Engineering, SDMCET, Dharwad-2

AOOP Assignment Submission Report

[Submitted as part of CTA Assignment No-2]

Course:	Advanced Object-Oriented Programming	Course Code:	18UCSE508
Semester:	V	Division:	A

Submitted by:

USN:	2sd20cs024	Name:	Apoorva N

1. Problem Definition:

Write a Java program to build the GUI application using JavaFX for the following requirements:

a) Read user name and password using appropriate JavaFX controls.

b) Validate the input. If user name and password are matched with the assumed values, then display the welcome scene with proper text.

c) If user name and password don't match, then raise appropriate exception.Java

Program:

```
package application;



//Demonstrate a text field.
import javafx.application.*;
import javafx.scene.*; import
javafx.stage.*; import
javafx.scene.layout.*; import
javafx.scene.control.*;
import javafx.event.*; import
javafx.geometry.*;
 class InvalidException extends Exception
{

}
public class AS21 extends Application {
 TextField tf,tff;
 Label response;
 Label response1;
       public static void main(String[]
args) {
     // Start the JavaFX application by calling launch().
     launch(args);
 }
```

```
 // Override the start() method.
public void start(Stage myStage) {

      // Give the stage a title.
      myStage.setTitle("Read Uname and pasword");

      // Use a FlowPane for the root node. In this case,
      // vertical and horizontal gaps of 10.
      FlowPane rootNode = new FlowPane(50, 50);

      // Center the controls in the scene.
      rootNode.setAlignment(Pos.CENTER);

      // Create a scene.
      Scene myScene = new Scene(rootNode, 400, 600);

      // Set the scene on the stage.
      myStage.setScene(myScene);

      // Create a label that will report the contents of the
      // text field.
      response = new Label("Verify Name: ");
      //response1= new Label("Verify password: ");

      // Create a button that gets the text.
      Button btnGetUserName = new Button("Login If Valid"); Separator
separator = new Separator(); separator.setPrefWidth(180);



      response1= new Label("Verify password: ");
//          Button btnGetUserName = new Button("Get Password");

      Separator separator1 = new Separator();

      separator1.setPrefWidth(180);
```

```
        // Create a text field.
        tf = new TextField();
        tff= new TextField();

        // Set the prompt.
        tf.setPromptText("Enter UserName");
        tff.setPromptText("Enter password");

        // Set preferred column count.
        tf.setPrefColumnCount(30);
        tff.setPrefColumnCount(30);

        // Handle action events for the text field. Action
        // events are generated when ENTER is pressed while    //
the text field has input focus. In this case, the      // text
in the field is obtained and displayed.     tf.setOnAction(new
EventHandler<ActionEvent>() {          public void
handle(ActionEvent ae) {
response.setText("UserName: " + tf.getText());
             }
        });
```

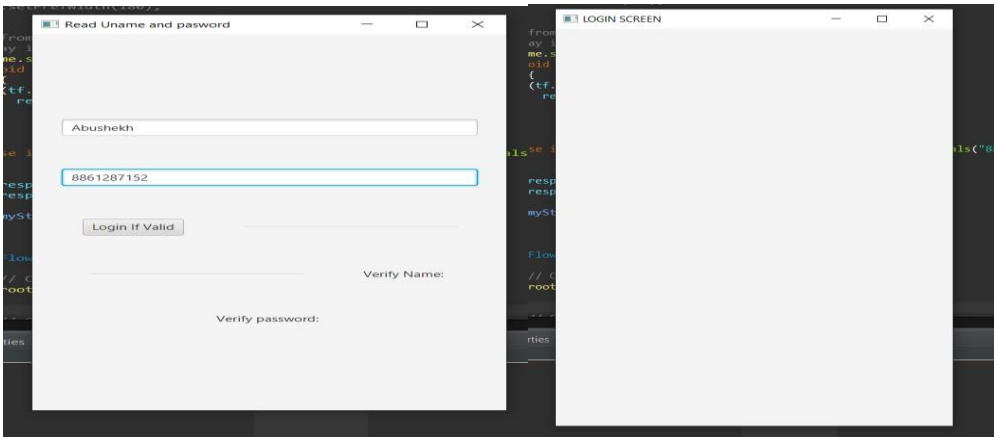
```
//    Separator separator = new Separator();
```

```
 //   separator.setPrefWidth(180);


      tff.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent ae) {

                  response1.setText("Password: " + tff.getText());
            }
      });


      //Separator separator1 = new Separator();
      //separator1.setPrefWidth(180);


      // Get text from the text field when the button is pressed
      // and display it.
      btnGetUserName.setOnAction(new EventHandler<ActionEvent>() {
            public void handle(ActionEvent ae) {
                  try {
                    if(tf.getText().equals(null) ||
tff.getText().equals(null)) {
                        response.setText("empty");



                  }

                   else if(tf.getText().equals("apoorva") &&
tff.getText().equals("1234567))
                  {

                    response.setText("User Name: " + tf.getText());
                    response1.setText("password: " + tff.getText());

                    myStage.setTitle("LOGIN SCREEN");
```

```
                    FlowPane rootNode = new FlowPane(50, 50);

                     // Center the controls in the scene.
          rootNode.setAlignment(Pos.CENTER);



                     // Create a scene.
                     Scene myScene = new Scene(rootNode, 400, 600);

                     // Set the scene on the stage.
      myStage.setScene(myScene);
                  }
                  else
                  {
                        response.setText("U entered a invalid name");
       response1.setText("U entered inavalid password");
                        throw new InvalidException();
                 }
              } catch(InvalidException i) {
              }

           }
        });

  // Separator separator = new Separator();
      separator.setPrefWidth(180);
      /* btnGet.setOnAction(new
      EventHandler<ActionEvent>() {    public void
      handle(ActionEvent ae) {

           }
      });*/
```

```
        // Use a separator to better organize the layout.


//      Separator separator1 = new Separator();
        separator1.setPrefWidth(180);



        // Add controls to the scene graph.
        rootNode.getChildren().addAll(tf,tff, btnGetUserName,
separator,separator1, response,response1);

        // Show the stage and its scene.
        myStage.show();
 }
}
```

Screen Shots of Execution:



2.Problem Definition:

Write a Java program to build the GUI application using JavaFX for the following requirements:

a) Create a Menu control to display the menu items: File, Edit & Help.

b) Create sub menus in the order: File → New, Open & Save. Edit → Cut, Copy & Paste.

The program must use Mnemonics and Accelerators (wherever appropriate) to Menu Items.

Java Program:

```
package application;
/* public class
AS22 {


}*/
//package application;
/* public class
MenuMnemonicsAcceleratorDemo {

}*/
//package application;

//Demonstrate Menus import
javafx.application.*; import
javafx.scene.*; import
javafx.stage.*; import
javafx.scene.layout.*; import
javafx.scene.control.*;
import javafx.scene.input.*;
import javafx.event.*;

public class AS22 extends Application {
 Label response;
  public static void main(String[] args)
{
     // Start the JavaFX application by calling launch().
     launch(args);
 }
```

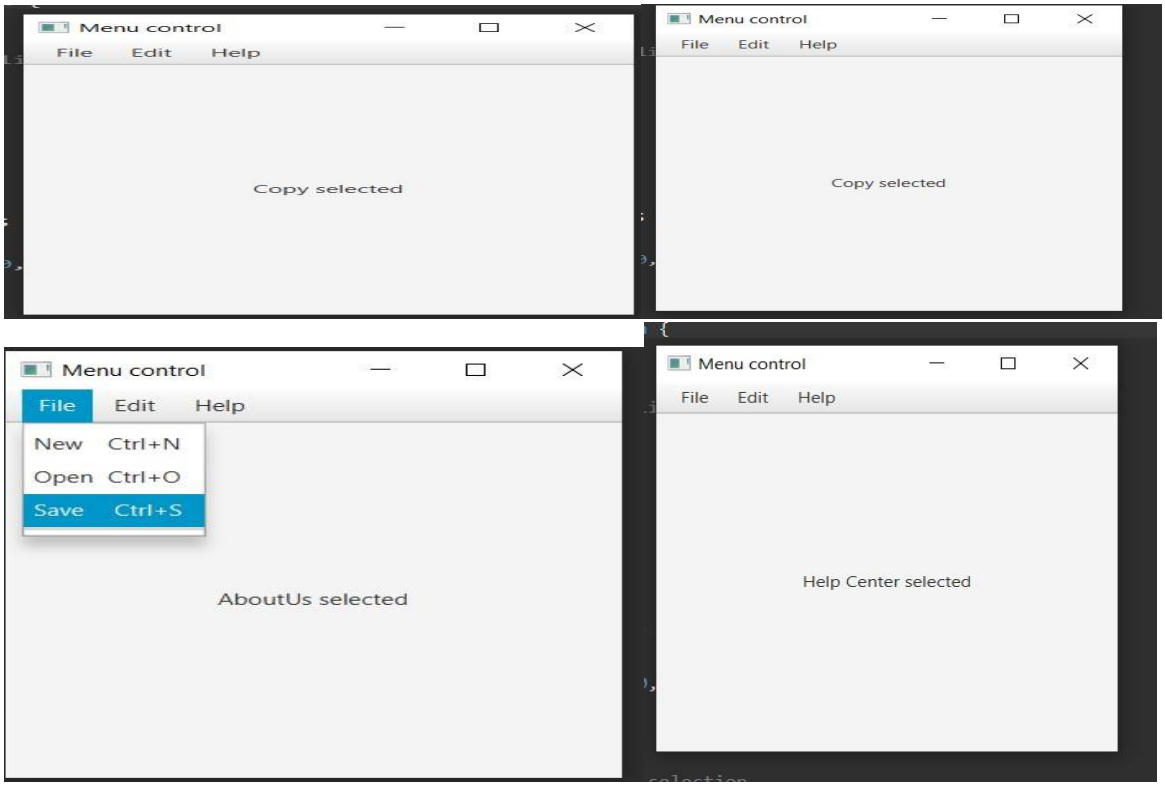
```
 // Override the start() method.
public void start(Stage myStage) {

      // Give the stage a title.
      myStage.setTitle("Menu control");

      // Use a BorderPane for the root node.
      BorderPane rootNode = new BorderPane();

      // Create a scene.
      Scene myScene = new Scene(rootNode, 300, 300);

      // Set the scene on the stage.
      myStage.setScene(myScene);

      // Create a label that will report the selection.
      response = new Label("Menu Demo");

      // Create the menu bar.
      MenuBar mb = new MenuBar();

      // Create the File menu.
      Menu fileMenu = new Menu("_File");  // now defines a mnemonic
      MenuItem New = new MenuItem("New");
      MenuItem open = new MenuItem("Open");
      MenuItem save = new MenuItem("Save");
      fileMenu.getItems().addAll(New,open,save, new
SeparatorMenuItem());

      // Turn on mnemonic
      fileMenu.setMnemonicParsing(true);

      // Add keyboard accelerators for the File menu.
```

```
        New.setAccelerator(KeyCombination.keyCombination("ctrl+N"));
        open.setAccelerator(KeyCombination.keyCombination("ctrl+O"));
        save.setAccelerator(KeyCombination.keyCombination("ctrl+S"));

        // Add File menu to the menu bar.
        mb.getMenus().add(fileMenu);

        // Create the Options menu.
        Menu editMenu = new Menu("Edit");

        // Create the Colors sub-menu.
        MenuItem cutMenu = new MenuItem("Cut");
        editMenu.getItems().add(cutMenu);


        // Create the Colors sub-menu.   MenuItem
    copyMenu = new MenuItem("Copy");
        editMenu.getItems().add(copyMenu);

        // Create the Priority sub-menu.
        MenuItem pasteMenu = new MenuItem("Paste");
        editMenu.getItems().add(pasteMenu);

        // Add a separator.
     // editMenu.getItems().add(new SeparatorMenuItem());


        // Add Options menu to the menu bar.
        mb.getMenus().add(editMenu);

        // Create the Help menu.
        Menu helpMenu = new Menu("Help");
        MenuItem aboutus = new MenuItem("AboutUs");
        helpMenu.getItems().add(aboutus);
```

```
        MenuItem  helpcenter= new MenuItem("Help Center");
        helpMenu.getItems().add(helpcenter);



        // Add Help menu to the menu bar.
        mb.getMenus().add(helpMenu);

         // Create one event handler that will handle menu action events.
        EventHandler<ActionEvent> MEHandler = new
EventHandler<ActionEvent>() {
              public void handle(ActionEvent ae) {
                    String name = ((MenuItem) ae.getTarget()).getText();
                    // If Exit is chosen, the program is terminated.
                    if (name.equals("Exit"))
        Platform.exit();
                    response.setText(name + " selected");
              }
        };

        // Set action event handlers for the menu items.
        New.setOnAction(MEHandler);
        open.setOnAction(MEHandler);
save.setOnAction(MEHandler);
        cutMenu.setOnAction(MEHandler);
        copyMenu.setOnAction(MEHandler);
        pasteMenu.setOnAction(MEHandler);
        helpMenu.setOnAction(MEHandler);
        aboutus.setOnAction(MEHandler);

        // Add the menu bar to the top of the border pane and
        // the response label to the center position.
        rootNode.setTop(mb);
        rootNode.setCenter(response);
```

```
        // Show the stage and its scene.
        myStage.show();
    }
}
```

Screen Shots of Execution:



3.Problem Definition:

Write a Java program to build the GUI application using JavaFX for the following requirements:

a) Create Context menu involving the menu items in the order: New & View.

b) Create sub menus for the above main context menu: New → File, Folder & Image.

View → Large, Medium & Small.

The context menu must be displayed on right-click of the mouse button.

Program:

```
package application;
/* public class
AS23 {
```

```
}
*/
//package application;
/* public class
ContextMenuDemo {

}*/

//package application;

//Demonstrate Menus import
javafx.application.*; import
javafx.scene.*; import
javafx.stage.*; import
javafx.scene.layout.*; import
javafx.scene.control.*;
import javafx.event.*; import
javafx.geometry.Pos; public
class AS23 extends
Application {
 Label response;
  public static void main(String[] args)
{
     // Start the JavaFX application by calling launch().
     launch(args);
 }

 // Override the start() method.
public void start(Stage myStage) {

     // Give the stage a title.
     myStage.setTitle("Demonstrate Menus");

     // Use a BorderPane for the root node.
```

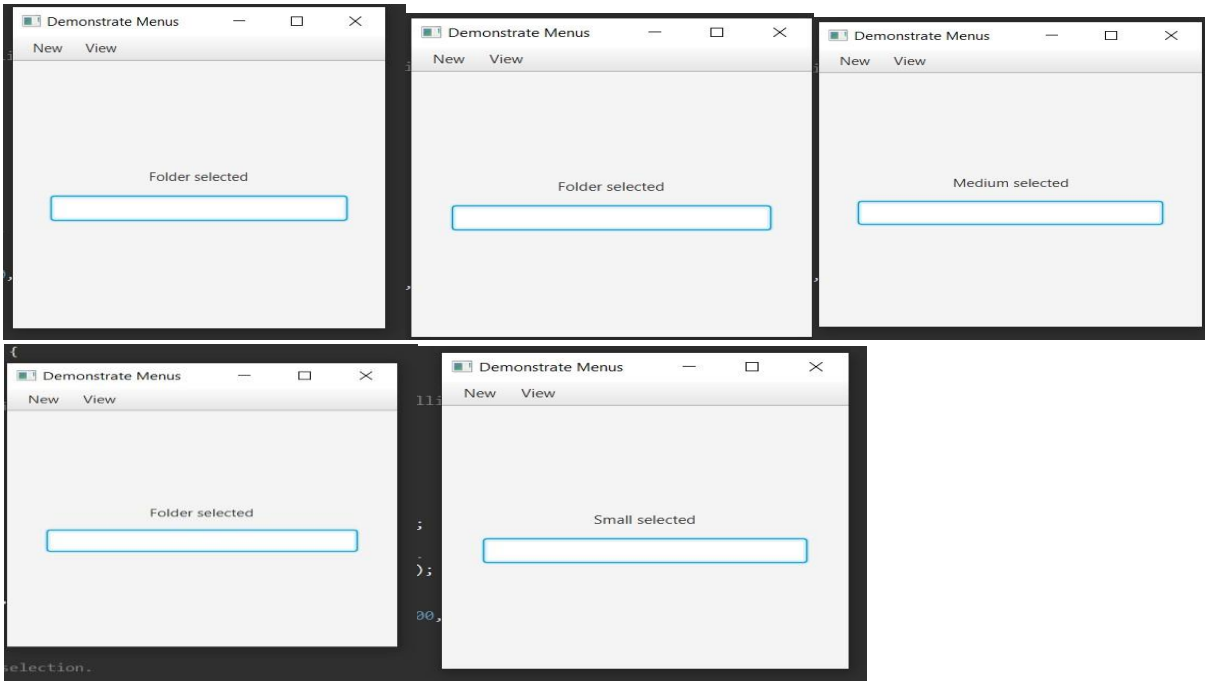
```
        BorderPane rootNode = new BorderPane();

        // Create a scene.
        Scene myScene = new Scene(rootNode, 300, 300);

        // Set the scene on the stage.
        myStage.setScene(myScene);

        // Create a label that will report the selection.
        response = new Label("Menu Demo");

        // Create the menu bar.
        MenuBar mb = new MenuBar();

        // Create the New menu.
        Menu NewMenu = new Menu("New");

        //create the view submenu
        MenuItem file = new MenuItem("File");
        MenuItem folder = new MenuItem("Folder");
        MenuItem image = new MenuItem("image");
        NewMenu.getItems().addAll(file,folder,image, new
SeparatorMenuItem());



        // Add New menu to the menu bar.
        mb.getMenus().add(NewMenu);

        // Create the view  menu.
        Menu viewMenu = new Menu("View");

        // Create the view sub-menu.
        MenuItem large = new MenuItem("Large");
        MenuItem medium = new MenuItem("Medium");
        MenuItem small = new MenuItem("Small");
```

```
      viewMenu.getItems().addAll(large,medium,small,new
SeparatorMenuItem());



      // Add a separator.
     // viewMenu.getItems().add(new SeparatorMenuItem());

      // Add view menu to the menu bar.
      mb.getMenus().add(viewMenu);

      // Create the context menu items
      MenuItem cut = new MenuItem("Cut");
      MenuItem copy = new MenuItem("Copy");
      MenuItem paste = new MenuItem("Paste");

      // Create a context (i.e., popup) menu that shows edit options.
      final ContextMenu editMenu = new ContextMenu(cut, copy, paste);

      // Create one event handler that will handle menu action events.
      EventHandler<ActionEvent> MEHandler = new
EventHandler<ActionEvent>() {
            public void handle(ActionEvent ae) {
                  String name = ((MenuItem) ae.getTarget()).getText();
                  // If Exit is chosen, the program is terminated.
                  if (name.equals("Exit"))
      Platform.exit();
                  response.setText(name + " selected");
            }
      };

      // Set action event handlers for the menu items.
      file.setOnAction(MEHandler);
      folder.setOnAction(MEHandler);
      image.setOnAction(MEHandler);
```

```
        large.setOnAction(MEHandler);
        medium.setOnAction(MEHandler);
        small.setOnAction(MEHandler);
        cut.setOnAction(MEHandler);
        copy.setOnAction(MEHandler);
        paste.setOnAction(MEHandler);

        // Create a text field and set its column width to 20.
        TextField tf = new TextField();
        tf.setPrefColumnCount(20);

        // Add the context menu to the textfield.
        tf.setContextMenu(editMenu);

        // Add the menu bar to the top of the border pane and
        // the response label to the center position.
        rootNode.setTop(mb);

        // Create a flow pane that will hold both the response
    // label and the text field.
        FlowPane fpRoot = new FlowPane(10, 10);

        // Center the controls in the scene.
        fpRoot.setAlignment(Pos.CENTER);

        // Add both the label and the text field to the flow pane.
        fpRoot.getChildren().addAll(response, tf);

        // Add the flow pane to the center of the border layout.
        rootNode.setCenter(fpRoot);

        // Show the stage and its scene.
        myStage.show();

    }
```

```
}
```

Screen Shots of Execution:



4.Problem Definition:

Write a JavaFX program that produces the following output when executed and displays Dialog Box on click of Register button

Java Program:

```
package application;
```

```
/* public class
AS24 {

}*/
//package application;

  import javafx.beans.*;
import
javafx.collections.*;
import
javafx.application.*;
import javafx.scene.*;
import javafx.stage.*;
import
javafx.scene.layout.*;
import
javafx.scene.control.*;
import javafx.event.*;
import javafx.geometry.*;
import java.io.IOException;
 public class AS24 extends Application
{
  public static void main(String[] args)
{
     // Start the JavaFX application by calling launch().
     launch(args);
 }

 // Override the start() method.
public void start(Stage myStage) {

     // Give the stage a title.
     myStage.setTitle("Registration Form");
```

```
    //Label for name
        Label nameLabel=new Label("Name");
        //Text Field for Name
        TextField nameText=new TextField();


        //Label for gender
        Label genderLabel=new Label("gender");
        //Toggle group of radio button
        ToggleGroup groupGender=new ToggleGroup();          RadioButton
maleRadio=new RadioButton("male");
maleRadio.setToggleGroup(groupGender);
        RadioButton femaleRadio=new RadioButton("female");
femaleRadio.setToggleGroup(groupGender);


        //Label for date of birth
        Label dobLabel=new Label("Date of birth");

        //date picker to choose date
        DatePicker datePicker=new DatePicker();

        //Label for Location
        Label stateLabel=new Label("Select Your State");

        //Choice box for location
        ChoiceBox stateChoiceBox=new ChoiceBox();
stateChoiceBox.getItems().addAll(
                "Andhra Pradesh","Arunachal
Pradesh","Assam","Bihar","Chhattisgarh","Goa","Gujarat","Haryana"
                ,"Himachal Pradesh","Jammu and
kashmir","Ladakh","Jharkhand","Karnataka","Kerala","Madhya Pradesh"

,"Maharashtra","Manipur","Meghalaya","Mizoram","Nagaland","Odisha","Pu
njab","Rajasthan","Sikkim"
                ,"Tamil
```

```
Nadu","Telangana","Tripura","Uttarakhand","Uttar Pradesh","West
Bengal"
        );

        //Label for technologies known
        Label qualificationLabel=new Label("Select Your
Qualifiaction");

        //Check box for education
        CheckBox UGCheckBox=new CheckBox("UG");
UGCheckBox.setIndeterminate(false);
        CheckBox PGCheckBox=new CheckBox("PG");
        PGCheckBox.setIndeterminate(false);
        CheckBox PhDCheckBox=new CheckBox("PhD");
        PhDCheckBox.setIndeterminate(false);


        //Label for register
        Button buttonRegister=new Button("Register");
Label valLabel=new Label();      buttonRegister.setOnAction(new
EventHandler<ActionEvent>() {          public void
handle(ActionEvent ae) {

                if(nameText.getText().equals("")){

                     valLabel.setText("unsuccessfull");
                }
            else {

               myStage.setTitle("Registartion Successfull");


                FlowPane rootNode = new FlowPane(50, 50);
```

```
                    // Center the controls in the scene.
rootNode.setAlignment(Pos.CENTER);
                    Label status=new Label("Registration Status");



                       Separator separator = new Separator();
      separator.setPrefWidth(100);

                        Label msg=new Label("Employee registration is
Successful!!!");



                       Button btnVal=new Button("ok");
                  // Create a scene.
                 Scene myScene = new Scene(rootNode, 400, 300);

rootNode.getChildren().addAll(status,separator,msg,btnVal);



                  // Set the scene on the stage.
                 myStage.setScene(myScene);
              }
            }
       });

         //Crating a Grid Pane
         GridPane gridPane=new GridPane();

         //Setting size for pane
gridPane.setMinSize(500,300);

         //Setting the padding
gridPane.setPadding(new Insets(10,10,10,10));
```

```
        //Setting the vertical and horizontal gaps between the columns
gridPane.setVgap(5);         gridPane.setHgap(5);

        //Setting the grid alignment
gridPane.setAlignment(Pos.CENTER);

        //Arranging all the nodes in the grid
gridPane.add(nameLabel,0,0);         gridPane.add(nameText,1,0);

gridPane.add(genderLabel,0,2);
gridPane.add(maleRadio,1,2);
gridPane.add(femaleRadio,2,2);

gridPane.add(dobLabel,0,1);
gridPane.add(datePicker,1,1);

gridPane.add(qualificationLabel,0,3);
gridPane.add(UGCheckBox,1,3);
gridPane.add(PGCheckBox,2,3);
gridPane.add(PhDCheckBox,3,3);

          gridPane.add(stateLabel,0,5);
gridPane.add(stateChoiceBox,1,5);

gridPane.add(buttonRegister,2,7);
gridPane.add(valLabel, 2, 8);



      // Create a scene.
      Scene myScene = new Scene(gridPane);


      // Set the scene on the stage.
      myStage.setScene(myScene);
```

```
        // Show the stage and its scene.
                    myStage.show();
    }

}
```

Screen Shots of Execution: