



Lab 9 GUI Programming

Lý thuyết và ngôn ngữ hướng đối tượng
(bài tập)



Lab's Objectives

- In this lab, you will practice with:
 - Create a simple GUI application with AWT
 - Create a simple GUI application with Swing
 - Work with JavaFX

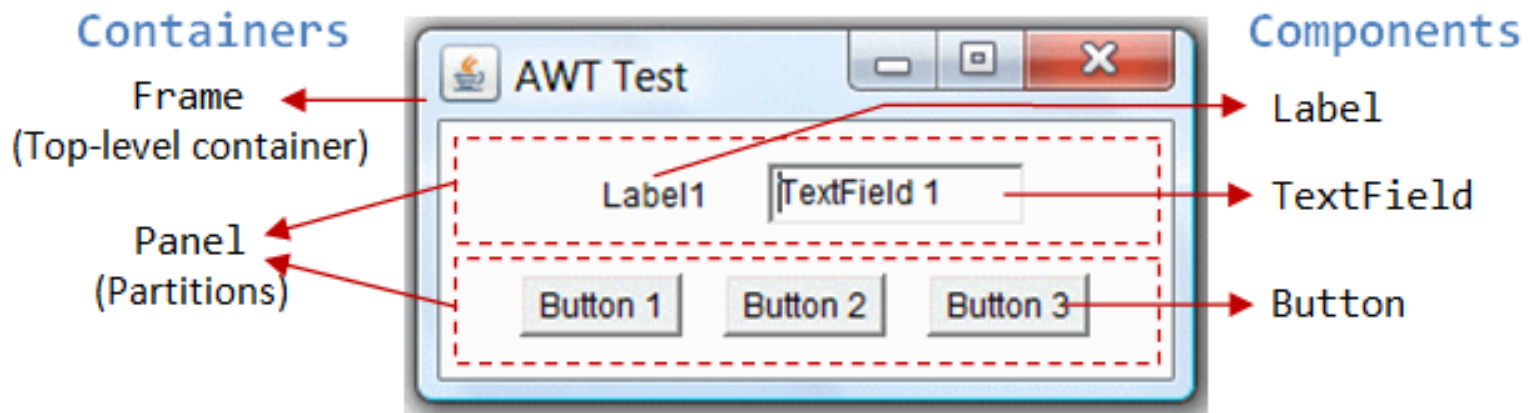


Java APIs cho lập trình đồ họa

- AWT (Abstract Windowing Toolkit)
 - Được giới thiệu trong JDK 1.0
 - Không nên dùng, dùng Swing thay thế
- Swing:
 - Mở rộng AWT
 - Tích hợp vào Java từ JDK 1.2
- JavaFX:
 - Thư viện Java, phát triển ứng dụng đa nền tảng (Desktop, mobile, TV, tablet)
- Các thư viện khác:
 - Eclipse's Standard Widget Toolkit (SWT)
 - Google Web Toolkit (GWT)
 - 3D Graphics API: Java OpenGL (JOGL), Java3D.

GUI application with AWT

- AWT Packages: **java.awt** and **java.awt.event** - are commonly-used
- Containers and Components



```
Panel pnl = new Panel();  
Button btn = new Button("Press");  
pnl.add(btn);  
// Panel is a container  
// Button is a component  
// The Panel container adds  
// a Button component
```

```

import java.awt.*;
import java.awt.event.*;

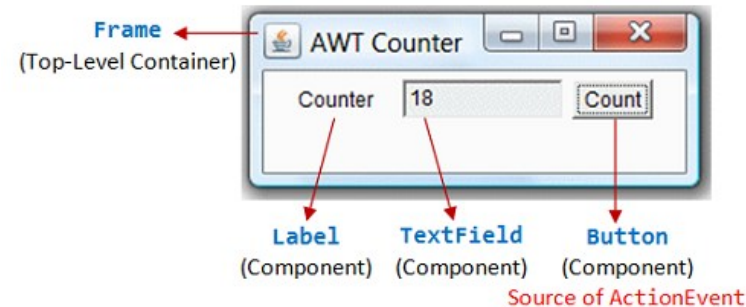
public class AWTCounter extends Frame implements ActionListener {
    private Label lblCount;
    private TextField tfCount;
    private Button btnCount;
    private int count = 0;

    public AWTCounter () {
        setLayout(new FlowLayout());
        lblCount = new Label("Counter"); add(lblCount);
        tfCount = new TextField(count + "", 10);
        tfCount.setEditable(false); add(tfCount);
        btnCount = new Button("Count"); add(btnCount);
        btnCount.addActionListener(this);
        setTitle("AWT Counter");
        setSize(250, 100);
        setVisible(true);
    }

    public static void main(String[] args) {
        AWTCounter app = new AWTCounter();
    }

    @Override
    public void actionPerformed(ActionEvent evt) {
        ++count;
        tfCount.setText(count + "");
    }
}

```

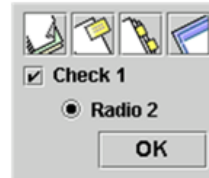
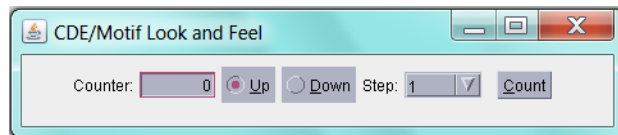
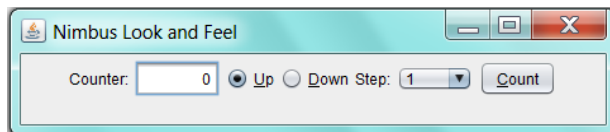
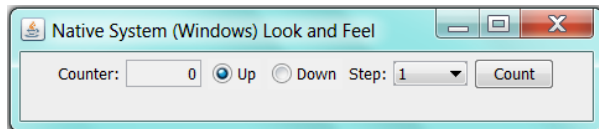




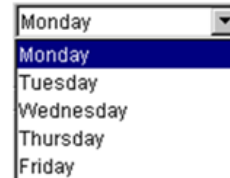
GUI application with Swing

- Swing API: for advanced graphical programming.
- Accessibility API: provides assistive technology for the disabled.
- Java 2D API: for high quality 2D graphics and images.
- Pluggable look and feel supports.
- Drag-and-drop support between Java and native applications.

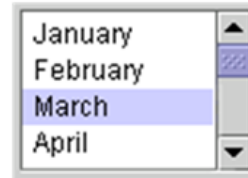
GUI application with Swing



Buttons



Combo Box



List



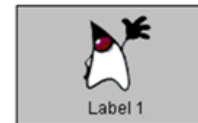
TextField



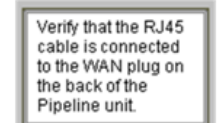
Slider



Menu



Label



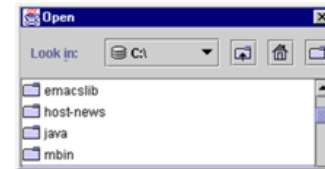
Text Area



ToolTip



Progress Bar



File Chooser



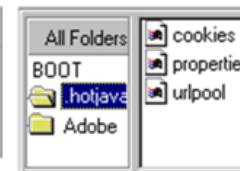
Color Chooser



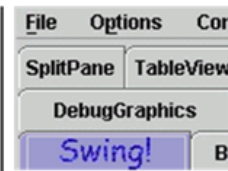
Table



Tree



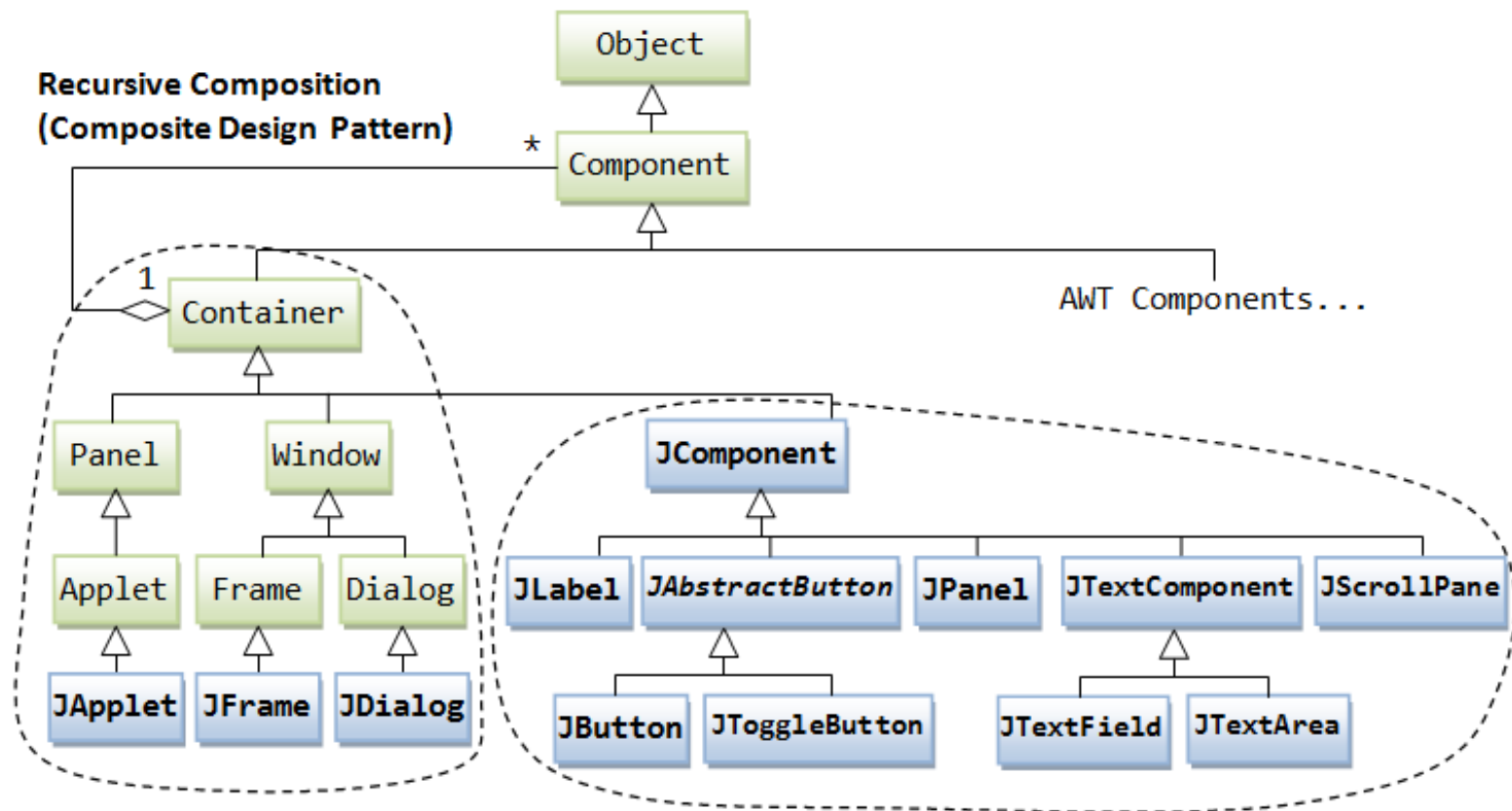
Split Pane



Tabbed Pane

GUI application with Swing

■ Swing's Components

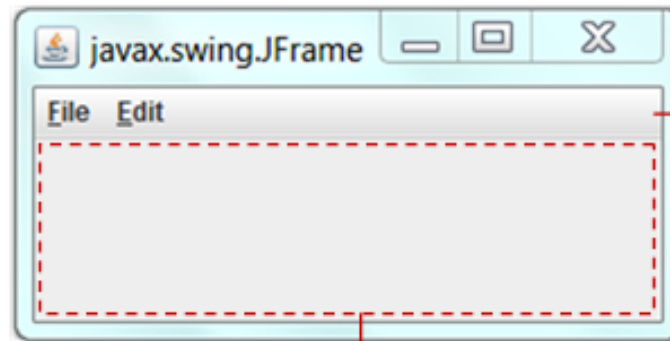


GUI application with Swing

■ Swing's Top-Level and Secondary Containers

- JFrame
- JDialog
- JApplet

`javax.swing.JFrame`



Menu Bar
(Optional)

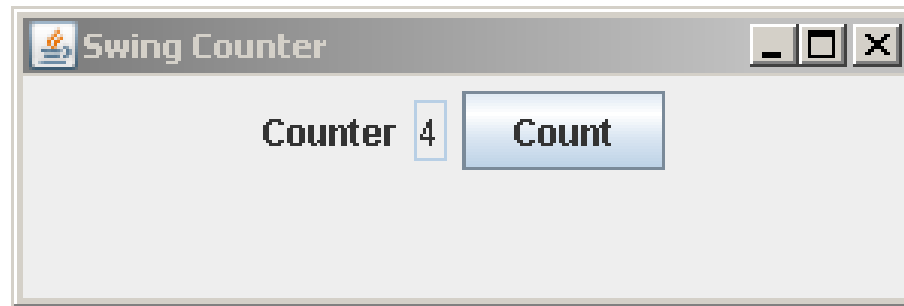
Content Pane

```
Container cp = aJFrame.getContentPane();  
aJFrame.setContentPane(aPanel);
```

```
public class SwingDemo extends JFrame {  
    public SwingDemo() {  
  
        Container cp = getContentPane();  
        cp.setLayout(new FlowLayout());  
        cp.add(new JLabel("Hello, world!"));  
        cp.add(new JButton("Button"));  
        .....  
    }  
    .....  
}
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class SwingCounter extends JFrame {
    private JTextField tfCount;
    private JButton btnCount;
    private int count = 0;
    public SwingCounter() {
        Container cp = getContentPane();
        cp.setLayout(new FlowLayout());
        cp.add(new JLabel("Counter"));
        tfCount = new JTextField("0");
        tfCount.setEditable(false);
        cp.add(tfCount);
        btnCount = new JButton("Count");
        cp.add(btnCount);
        btnCount.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent evt) {
                ++count; tfCount.setText(count + "");
            }
        });
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Swing Counter");
        setSize(300, 100);
        setVisible(true);
    }
}
```

```
public static void main(String[] args) {  
    SwingUtilities.invokeLater(new Runnable() {  
        @Override  
        public void run() {  
            new SwingCounter();  
        }  
    });  
}
```





JavaFX – Tính năng (Features)

- Viết bằng Java, dùng được trong các ngôn ngữ thực thi trên máy ảo Java (Java, Groovy và JRuby)
- Hỗ trợ FXML (tương tự HTML), giúp dễ dàng định nghĩa giao diện người dùng
- Scene Builder: JavaFX cung cấp ứng dụng Scene Builder trên các nền tảng khác nhau, cho phép LTV kéo thả khi thiết kế giao diện
- Tương thích với Swing: trong ứng dụng JavaFX có thể nhúng các thành phần Swing
- Built-in UI controls: JavaFX cung cấp các control đa dạng để phát triển ứng dụng
- CSS like Styling: thiết kế giao diện với các tính năng giống như trong CSS
- ...



Cài đặt JavaFX

- Trang chủ JavaFX: <https://openjfx.io/>
- Trang download thư viện JavaFX:
<https://gluonhq.com/products/javafx/>
- Download, giải nén, copy các file trong thư mục lib, add vào build path của project
- Lưu ý khi chạy chương trình trên IDE Eclipse
 - Vào runtime configuration, cấu hình VM arguments:
 - `--module-path ${project_classpath:REPLACE_ME_WITH_YOUR_PROJECT_NAME}`
 - `--add-modules javafx.controls,javafx.fxml`
 - Bỏ chọn: "Use the -XstartOnFirstThread argument when launching with SWT"



Tiện ích JavaFX trên Eclipse

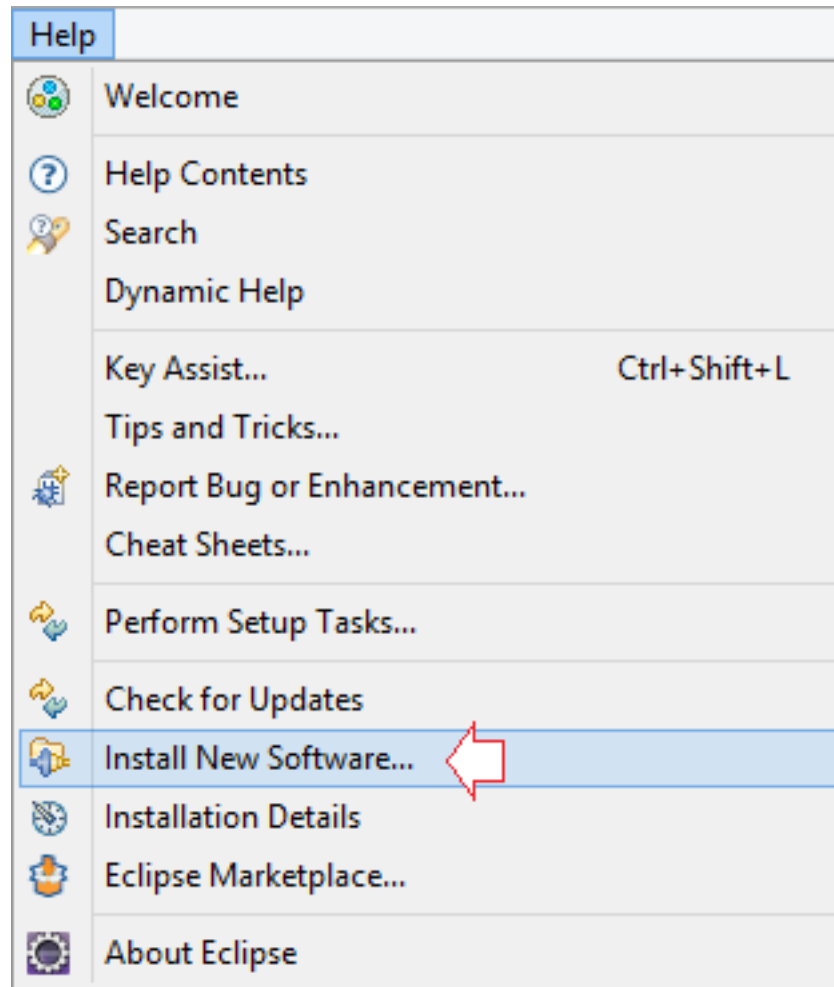
- e(fx)clipse

- <https://www.eclipse.org/efxclipse/releases.html>
- Là một Eclipse plugin
- Công cụ hỗ trợ lập trình JavaFX trên Eclipse

- JavaFX Scene Builder

- <https://www.oracle.com/java/technologies/javafxscenebuilder-1x-archive-downloads.html>
- Công cụ độc lập, đa nền tảng, thiết kế trực quan giao diện cho ứng dụng **JavaFX**.
- Cho phép kéo thả các thành phần giao diện người dùng, thay đổi thuộc tính, áp dụng style
- Đầu ra: file FXML dùng trong ứng dụng JavaFX

Cài đặt e(fx)clipse



Cài đặt e(fx)clipse

Available Software
Select a site or enter the location of a site.

Work with: ▼ Add...

Find more software by working with the ["Available Software Sites"](#) preferences.

type filter text

Name	Version
<input type="checkbox"/> ⓘ There is no site selected.	

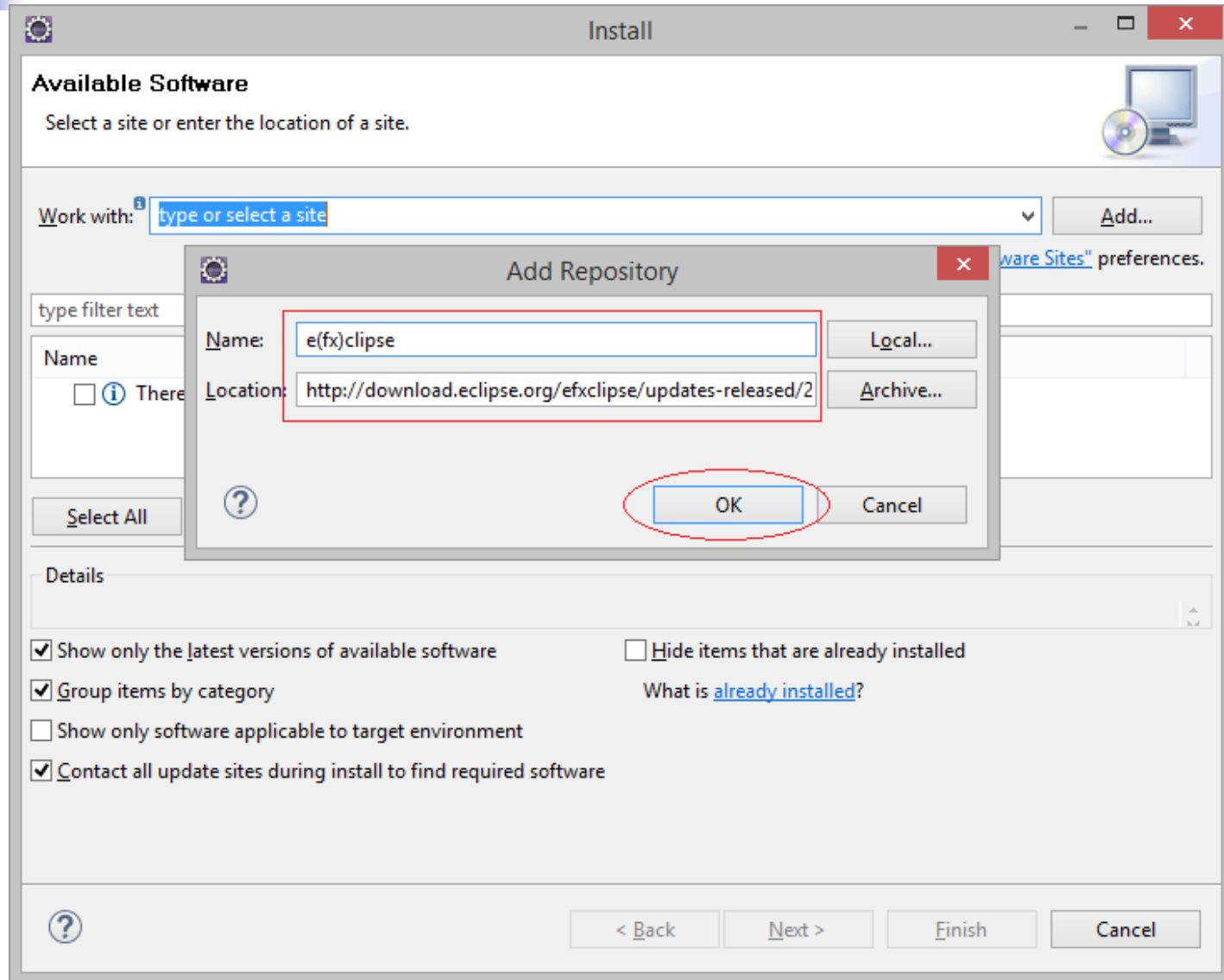
Select All Deselect All

Nhập vào:
<http://download.eclipse.org/efxclipse/updates-released/3.0.0/site>

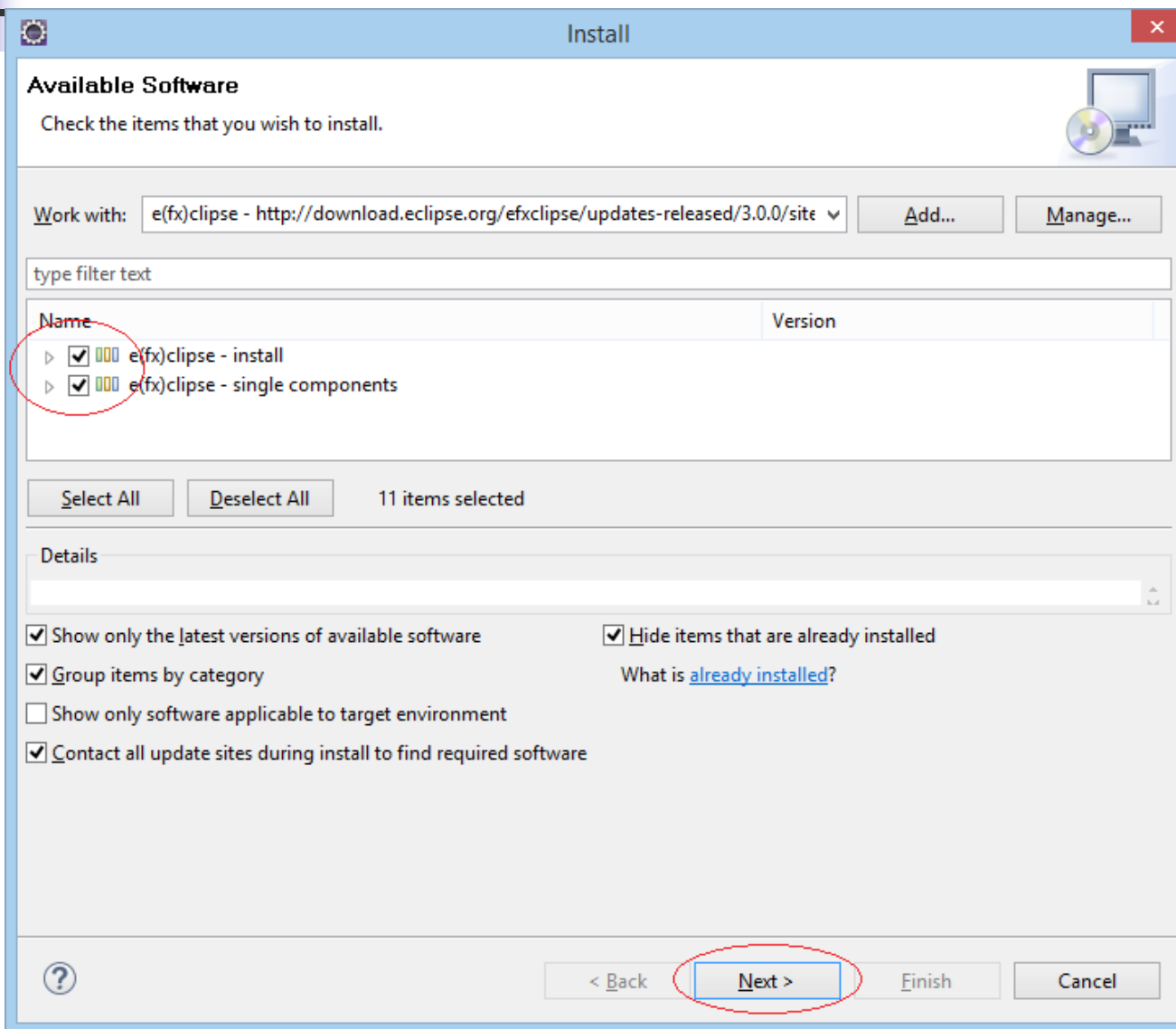
Xem các Phiên bản mới nhất tại:
<https://www.eclipse.org/efxclipse/releases.html>

? < Back Next > Finish Cancel

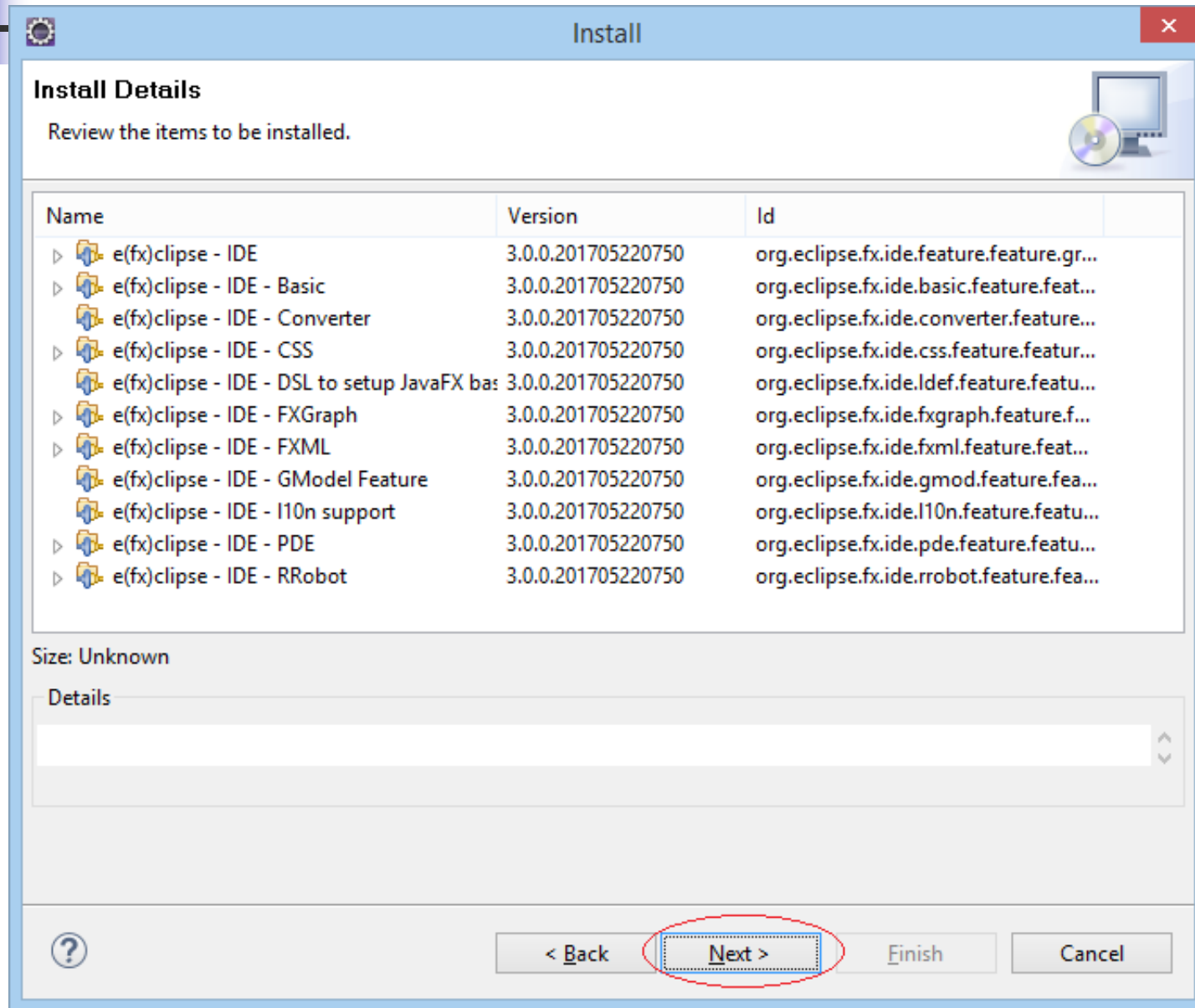
Cài đặt e(fx)clipse



Cài đặt e(fx)clipse

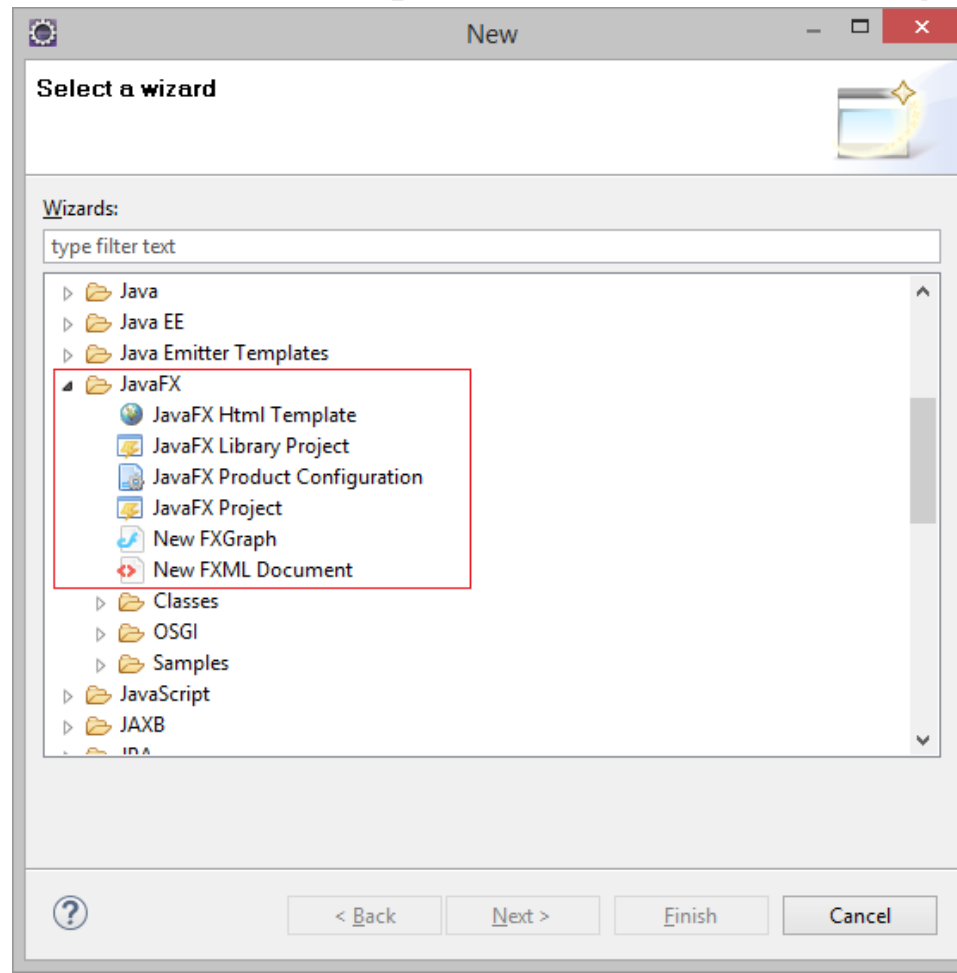


Cài đặt e(fx)clipse



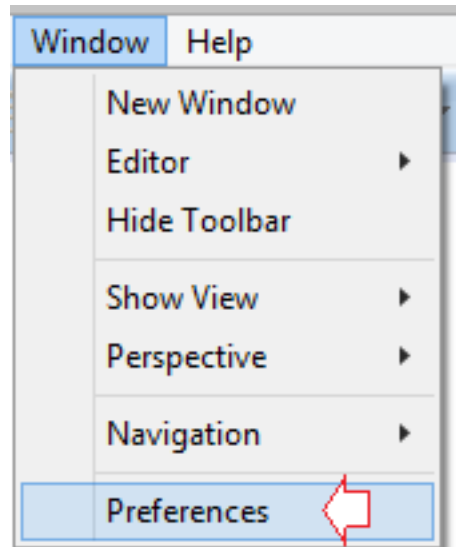
Cài đặt e(fx)clipse

- Sau khi cài đặt và khởi động lại **Eclipse**, vào menu **File/New/Others ...** sẽ thấy các **Wizard** cho phép lập trình **JavaFX**

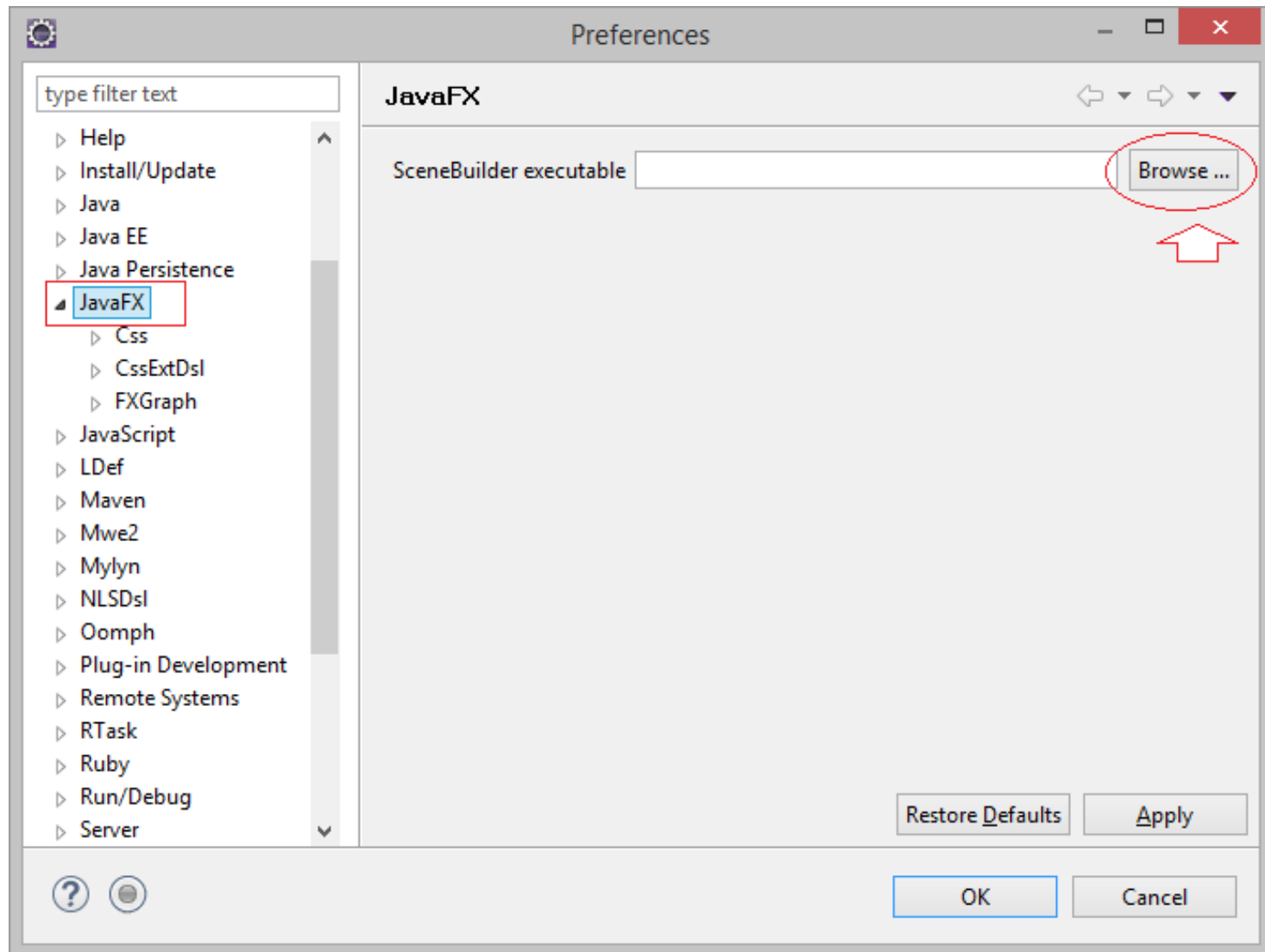


Tích hợp JavaFX Scene Builder vào Eclipse

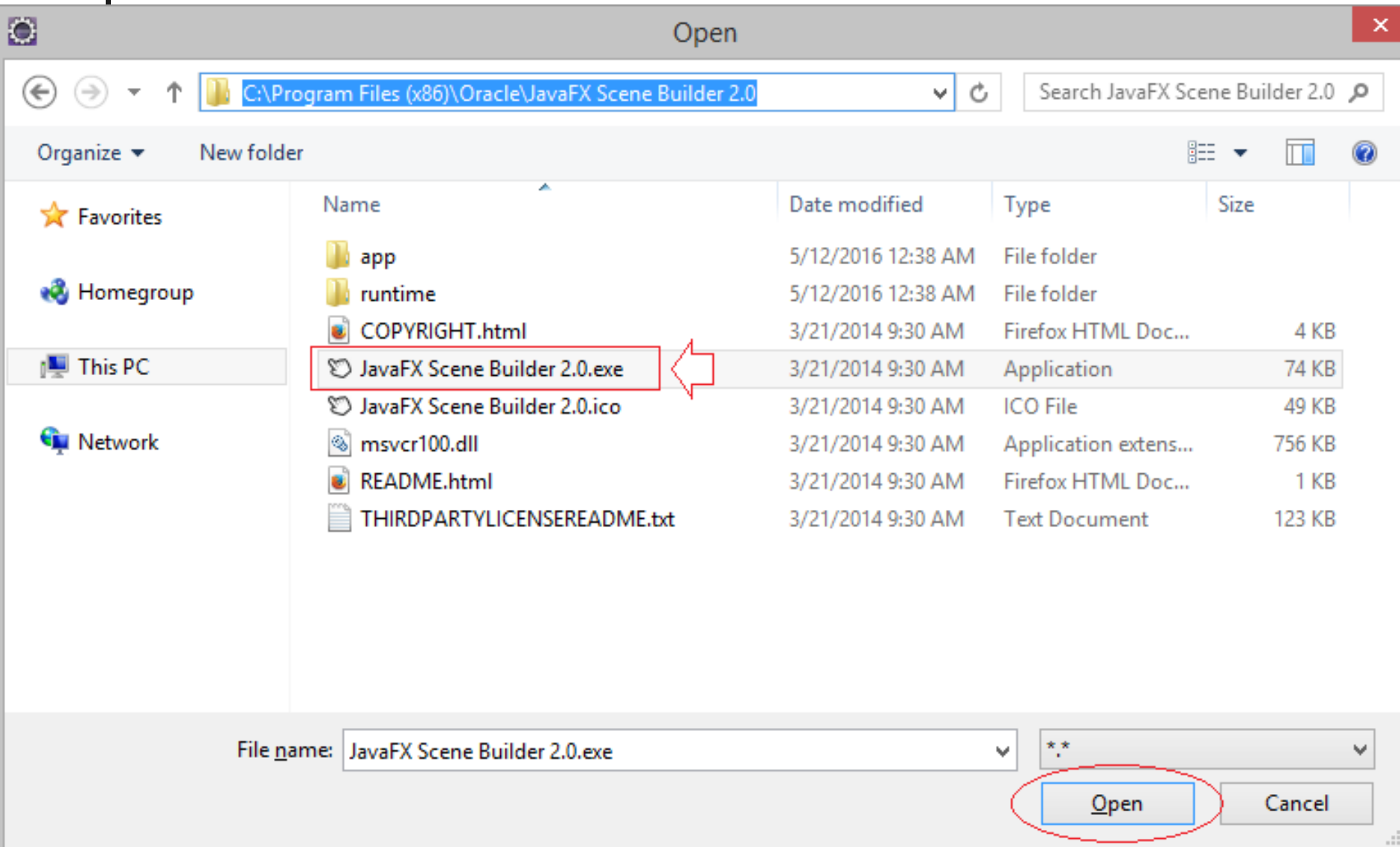
- Download, cài đặt JavaFX Scene Builder
- Trên eclipse, vào Window/Preferences



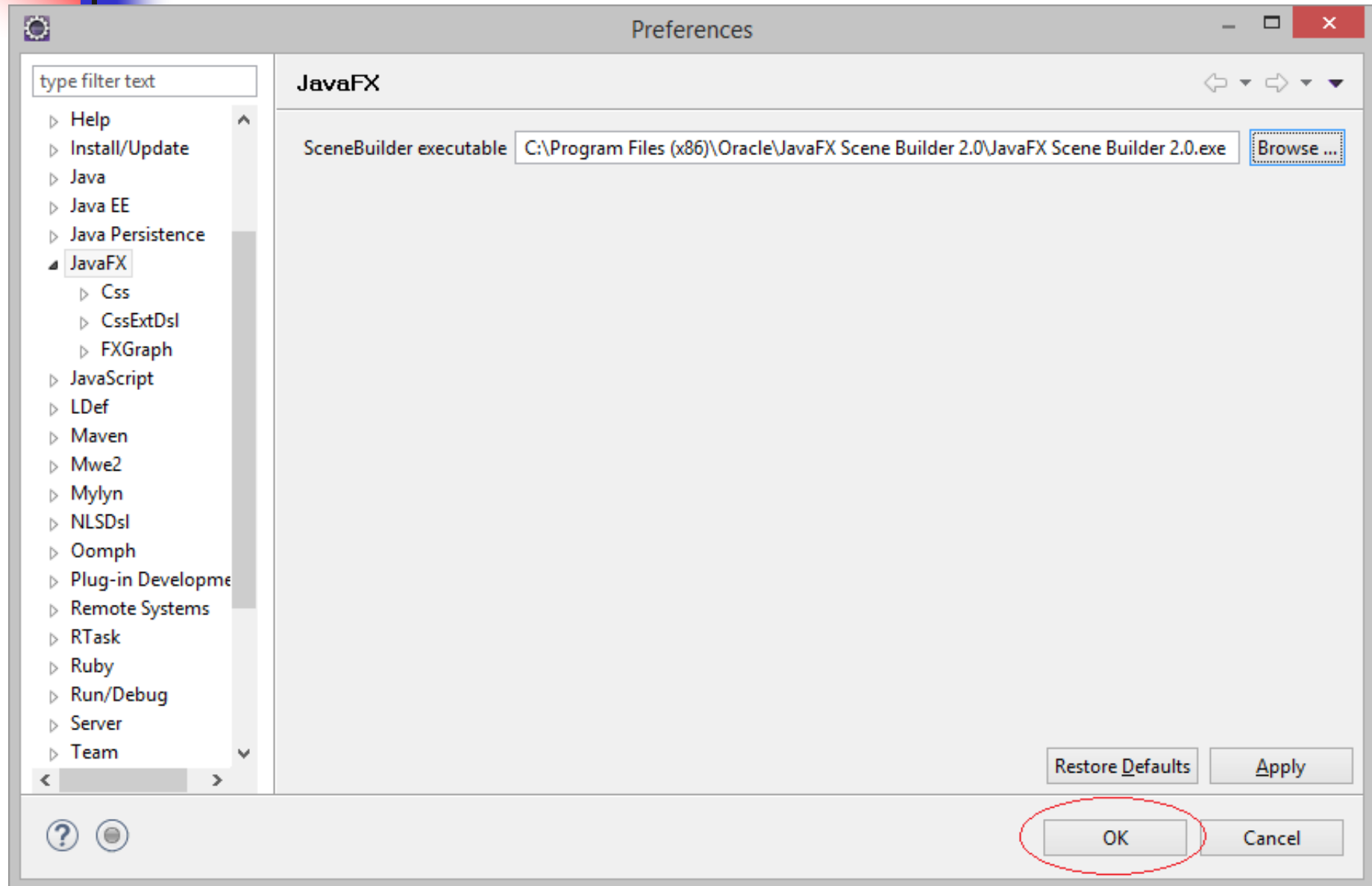
Tích hợp JavaFX Scene Builder vào Eclipse



Tích hợp JavaFX Scene Builder vào Eclipse

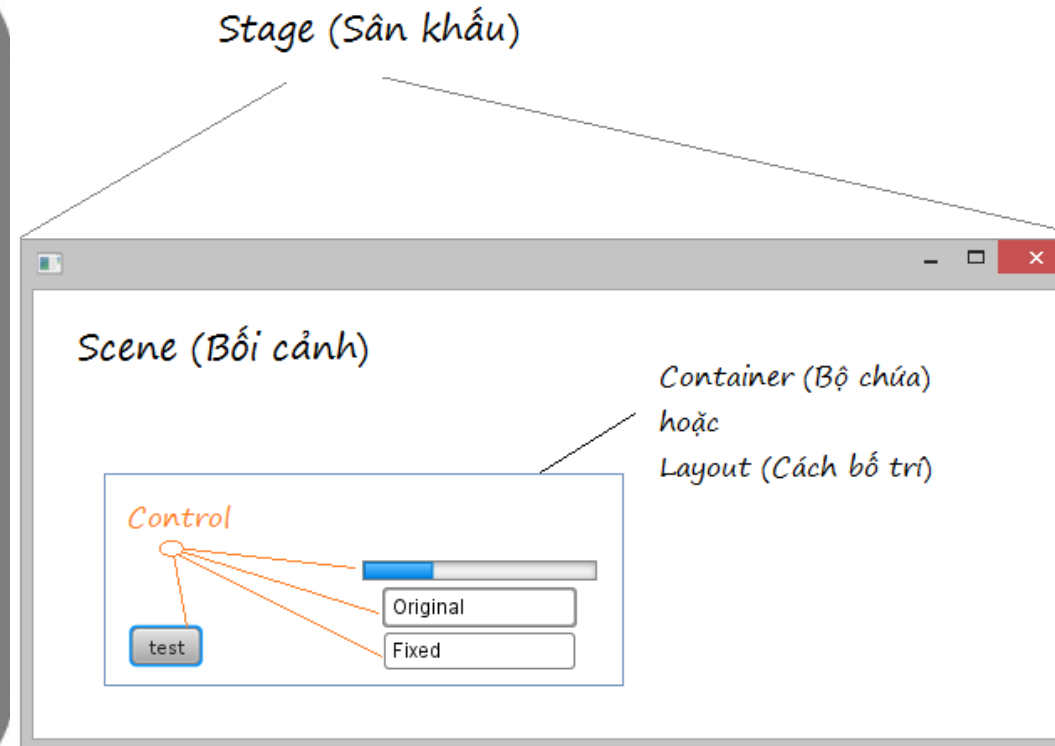
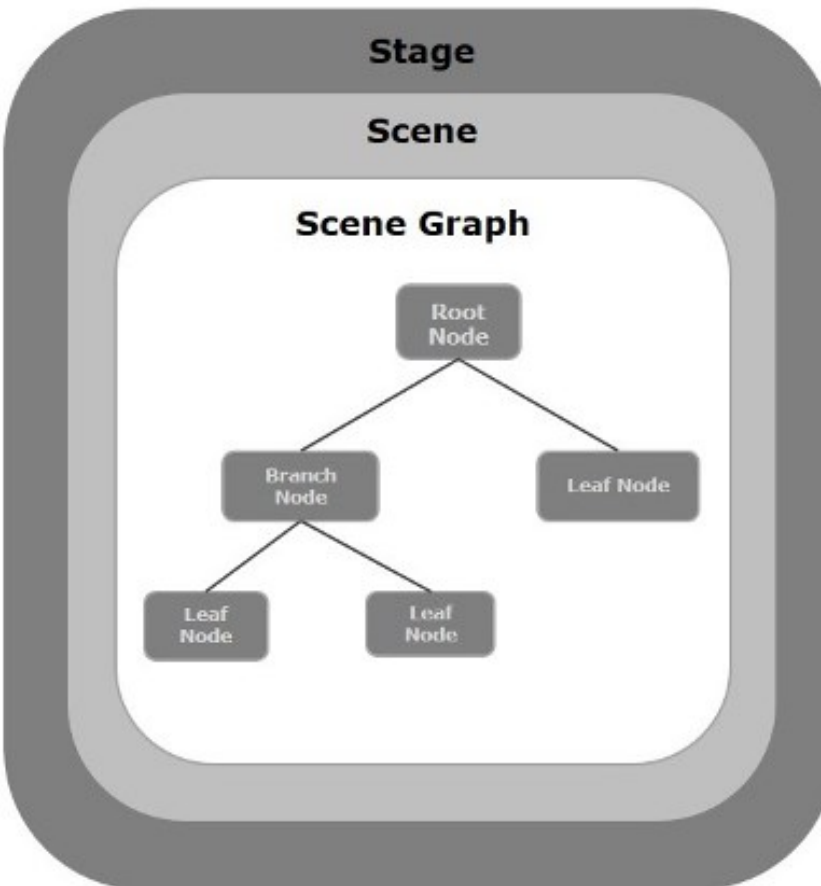


Tích hợp JavaFX Scene Builder vào Eclipse



Các thành phần giao diện JavaFX

- Cấu trúc ứng dụng JavaFX gồm 3 thành phần chính: Stage, Scene và Nodes



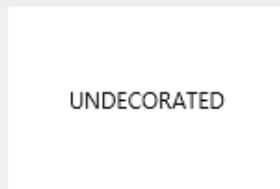
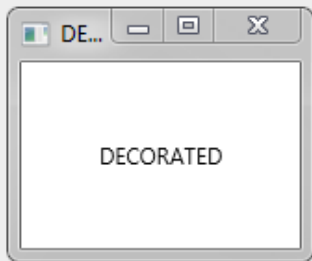


Stage

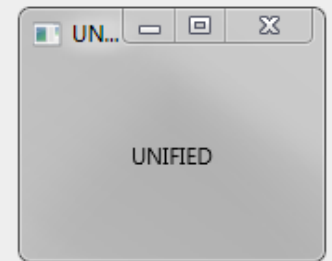
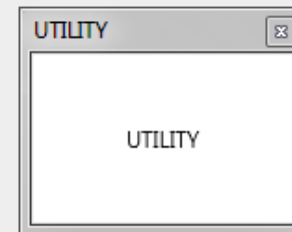
- Đối tượng Stage (Window) chứa tất cả các đối tượng khác trong ứng dụng JavaFX
- Là đối tượng của lớp `javafx.stage.Stage`
- Đối tượng Stage sẽ truyền làm tham số cho phương thức `start()` của lớp `Application` (Xem lại ví dụ `HelloWorld JavaFX`)
- Có 2 tham số `width` và `height`
- Được chia làm 2 phần: `Content Area` và `Decorations` (`Title bar` và `Borders`)
- Để hiển thị Stage, gọi phương thức `show()`
- Có 5 style cho Stage: `Decorated`, `Undecorated`, `Transparent`, `Unified`, `Utility`

Stage – thiết lập style

```
stage.initStyle(StageStyle.DECORATED);  
//stage.initStyle(StageStyle.UNDECORATED);  
//stage.initStyle(StageStyle.TRANSPARENT);  
//stage.initStyle(StageStyle.UNIFIED);  
//stage.initStyle(StageStyle.UTILITY);
```



TRANSPARENT





Scene

- Scene chứa tất cả các nội dung trình bày của một **scene graph**
- Là đối tượng của lớp `javafx.scene.Scene`
- Một Scene được thêm vào duy nhất một Stage
- Một số phương thức khởi dựng:
 - `Scene(Parent root)`
 - `Scene(Parent root, double width, double height)`
 - ...



Scene Graph và Nodes

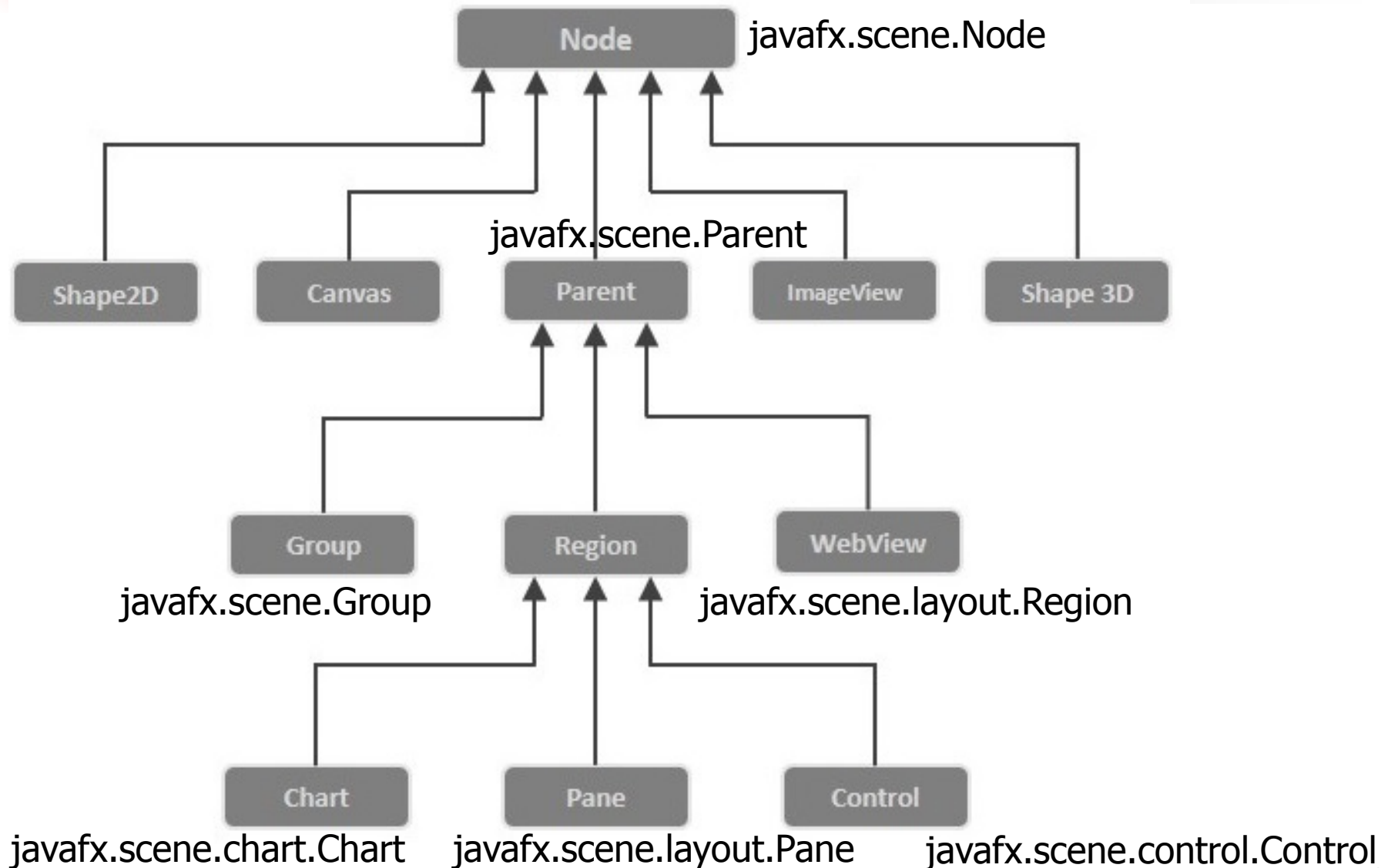
- **Scene graph**: là cấu trúc dữ liệu phân cấp dạng tree biểu diễn nội dung một **Scene**, bao gồm tất cả các controls, layout
- **Node**: là một đối tượng đồ họa của một Scene graph, bao gồm
 - Đối tượng hình học (2D và 3D) như: Circle, Rectangle, Polygon, ...
 - Đối tượng điều khiển UI như: Button, Checkbox, TextArea, ...
 - Phần tử đa phương tiện Media như: Audio, Video, Image
- Lớp cơ sở cho tất cả các loại Node: `javafx.scene.Node`



Scene Graph và Nodes

- Có 2 loại Node:
 - Branch Node/Parent Node: là các node có các node con, lớp cơ sở là `javafx.scene.Parent` (lớp trừu tượng). Có 3 loại:
 - **Group**: là một node tổng hợp, chứa một list các node con. Khi render node Group, tất cả các node con sẽ lần lượt được render. Các chuyển đổi hiệu ứng áp dụng cho một Group được áp dụng cho tất cả node con
 - **Region**: là lớp cơ sở cho các UI Controls, bao gồm **Chart** (AreaChart, BarChart, BubbleChart, ...), **Pane** (AnchorPane, BorderPane, DialogPane, FlowPane, HBox, VBox ...), **Control** (Accordion, ButtonBar, ChoiceBox, ComboBoxBase, HTML editor, ...)
 - **WebView**: tương tự như Browser
 - Leaf Node: là node không có node con. Ví dụ: Rectangle, Ellipse, Box, ImageView, MediaView
- Lưu ý: Root node là một branch/parent node, nhưng root node không có node cha.

Cây phân cấp kế thừa Node





Cách tạo ứng dụng JavaFX

- Viết lớp kế thừa lớp `javafx.application.Application`, thực thi phương thức trừu tượng `start`
- Trong phương thức `main`, gọi phương thức static `launch()`. Phương thức `launch` đã tự động gọi phương thức `start()`

```
public class JavafxSample extends Application {
    @Override
    public void start(Stage primaryStage) throws Exception {
        /*
         * Code for JavaFX application.
         * (Stage, scene, scene graph)
         */
    }
    public static void main(String args[]){
        launch(args);
    }
}
```




Vòng đời ứng dụng JavaFX

- Có 3 phương thức trong vòng đời ứng dụng JavaFX: `start()`, `stop()`, `init()`
- Cài đặt mặc định là phương thức rỗng, có thể override khi muốn làm gì đó
- Thứ tự hành động
 - Tạo thể hiện của lớp application
 - Gọi phương thức `init` (không tạo stage hoặc scene trong phương thức này)
 - Gọi phương thức `start`
 - Khi ứng dụng kết thúc, gọi phương thức `stop`
- Khi cửa sổ (window) cuối cùng của ứng dụng JavaFX được đóng, ứng dụng tự động kết thúc. Có thể gọi tường minh với phương thức `Platform.exit()` hoặc `System.exit(int)`



Cài đặt phương thức start

- 3 bước:
 - Tạo một Scene graph với các Node
 - Tạo một Scene với kích thước mong muốn và thêm vào root node của scene graph
 - Tạo một Stage, thêm Scene vào Stage, và hiển thị nội dung của Stage



Tạo scene graph

- Cần tạo node gốc, có thể là Group, Region hoặc WebView

- VD:

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

- Thêm các node vào root node theo 2 cách

- Cách 1:

```
//Retrieving the observable list object  
ObservableList list = root.getChildren();
```

```
//Setting a node object as a node  
list.add(NodeObject);
```

- Cách 2:

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



Tạo Scene

- Khởi tạo đối tượng Scene, bắt buộc phải truyền tham số là root object

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

- Có thể vừa khởi tạo vừa thiết lập kích thước của Scene

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



Tạo Stage

- Đối tượng Stage được truyền làm tham số cho phương thức start() của lớp Application
→ không cần khởi tạo
- Thao tác cơ bản

```
//Setting the title to Stage.  
primaryStage.setTitle("Sample application");
```

```
//Setting the scene to Stage  
primaryStage.setScene(scene);
```

```
//Displaying the stage  
primaryStage.show();
```

JavaFX Hello World

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class HelloWorld extends Application {
    @Override
    public void start(Stage primaryStage) {
        Button btn = new Button();
        btn.setText("Say 'Hello World'");
        btn.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });

        StackPane root = new StackPane();
        root.getChildren().add(btn);

        Scene scene = new Scene(root, 300, 250);

        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```



Ví dụ: tạo ứng dụng với cửa sổ JavaFX rỗng

```
public class JavafxSample extends Application {
    @Override
    public void start(Stage primaryStage) throws Exception {
        //creating a Group object
        Group group = new Group();

        //Creating a Scene
        Scene scene = new Scene(group , 600, 300);

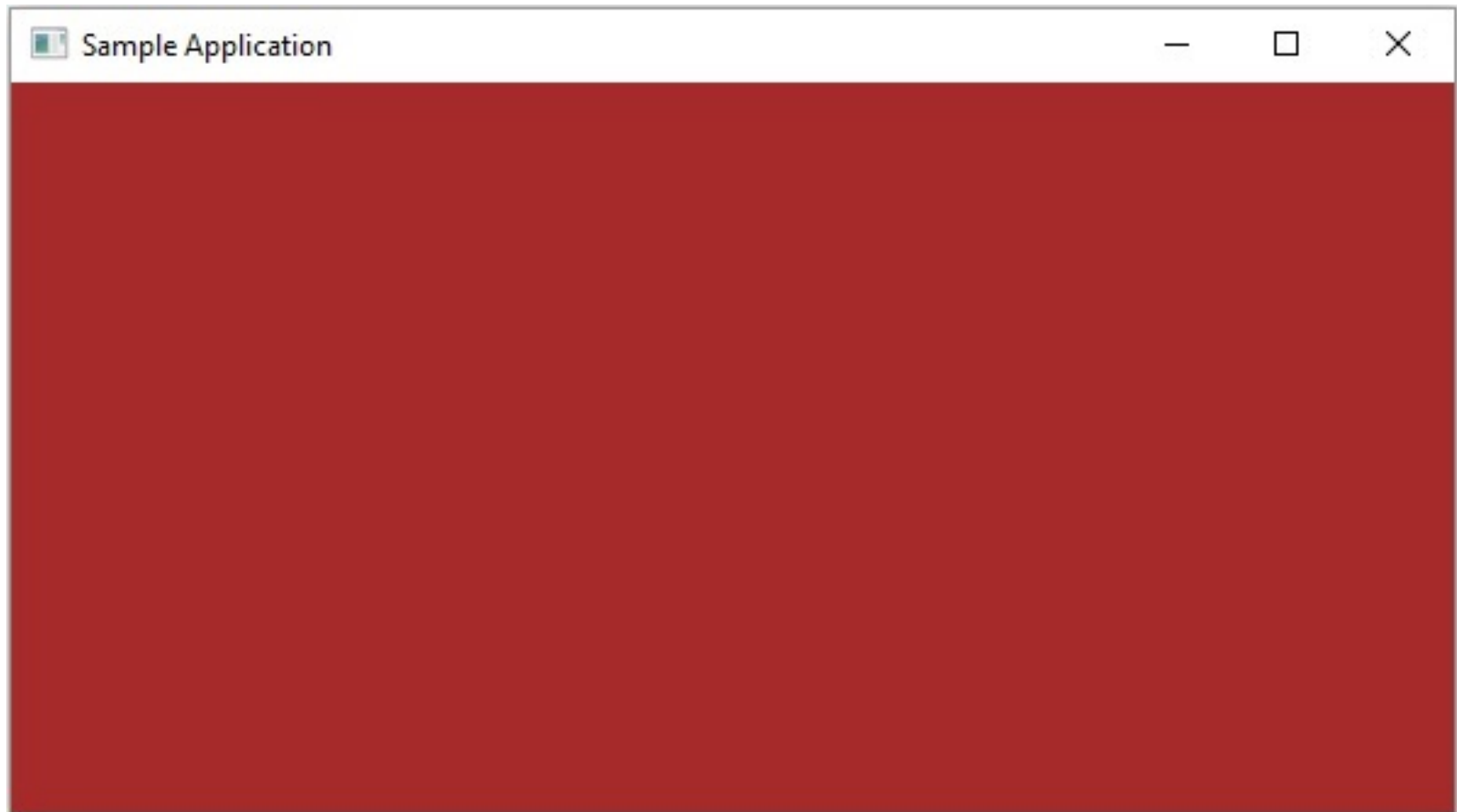
        //setting color to the scene
        scene.setFill(Color.BROWN);

        //Setting the title to Stage.
        primaryStage.setTitle("Sample Application");

        //Adding the scene to Stage
        primaryStage.setScene(scene);

        //Displaying the contents of the stage
        primaryStage.show();
    }
    public static void main(String args[]){
        launch(args);
    }
}
```

Ví dụ: tạo ứng dụng với cửa sổ JavaFX rỗng





VD: vẽ đường thẳng

```
public class DrawingLine extends Application{
    @Override
    public void start(Stage stage) {
        //Creating a line object
        Line line = new Line();

        //Setting the properties to a line
        line.setStartX(100.0);
        line.setStartY(150.0);
        line.setEndX(500.0);
        line.setEndY(150.0);

        //Creating a Group
        Group root = new Group(line);

        //Creating a Scene
        Scene scene = new Scene(root, 600, 300);

        //Setting title to the scene
        stage.setTitle("Sample application");

        //Adding the scene to the stage
        stage.setScene(scene);

        //Displaying the contents of a scene
        stage.show();
    }
    public static void main(String args[]){
        launch(args);
    }
}
```

VD: vẽ đường thẳng



VD: hiển thị dòng chữ

```
public class DisplayingText extends Application {
    @Override
    public void start(Stage stage) {
        //Creating a Text object
        Text text = new Text();

        //Setting font to the text
        text.setFont(new Font(45));

        //setting the position of the text
        text.setX(50);
        text.setY(150);

        //Setting the text to be added.
        text.setText("Welcome to Tutorialspoint");

        //Creating a Group object
        Group root = new Group();

        //Retrieving the observable list object
        ObservableList list = root.getChildren();

        //Setting the text object as a node to the group object
        list.add(text);

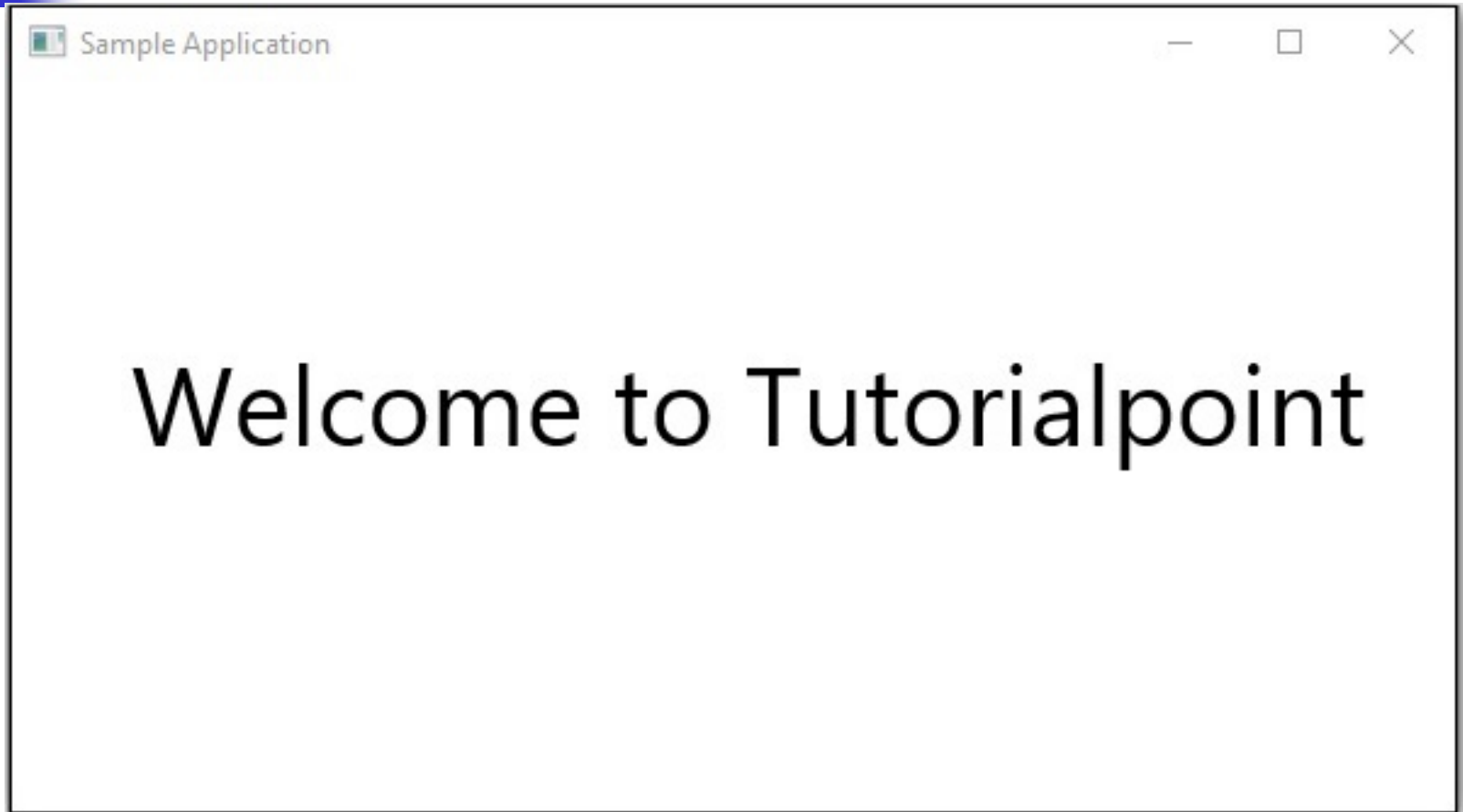
        //Creating a scene object
        Scene scene = new Scene(root, 600, 300);

        //Setting title to the Stage
        stage.setTitle("Sample Application");

        //Adding scene to the stage
        stage.setScene(scene);

        //Displaying the contents of the stage
        stage.show();
    }
    public static void main(String args[]){
        launch(args);
    }
}
```

VD: hiển thị dòng chữ





Ví dụ: hiển thị 2 dòng text

```
public class DecorationsExample extends Application {
    @Override
    public void start(Stage stage) {
        //Creating a Text Example object
        Text text1 = new Text("Hi how are you");

        //Setting font to the text
        text1.setFont(
            Font.font("verdana", FontWeight.BOLD, FontPosture.REGULAR, 20)
        );

        //setting the position of the text
        text1.setX(50);
        text1.setY(75);

        //Striking through the text
        text1.setStrikethrough(true);

        //Creating a Text Example object
        Text text2 = new Text("Welcome to Tutorialspoint");

        //Setting font to the text
        text2.setFont(
            Font.font("verdana", FontWeight.BOLD, FontPosture.REGULAR, 20)
        );
    }
}
```



Ví dụ: hiển thị 2 dòng text

```
//setting the position of the text
text2.setX(50);
text2.setY(150);

//underlining the text
text2.setUnderline(true);

//Creating a Group object
Group root = new Group(text1, text2);

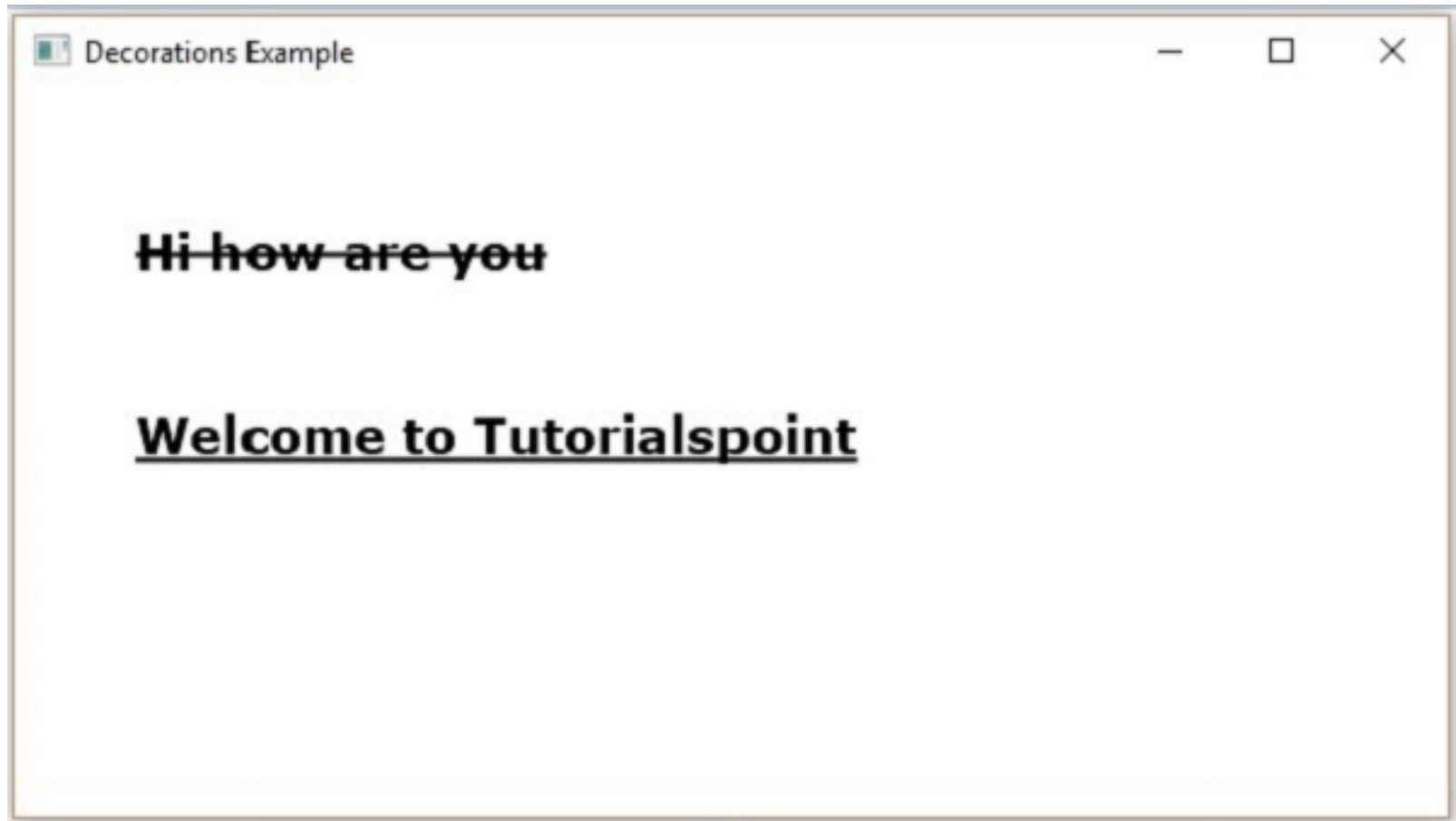
//Creating a scene object
Scene scene = new Scene(root, 600, 300);

//Setting title to the Stage
stage.setTitle("Decorations Example");

//Adding scene to the stage
stage.setScene(scene);

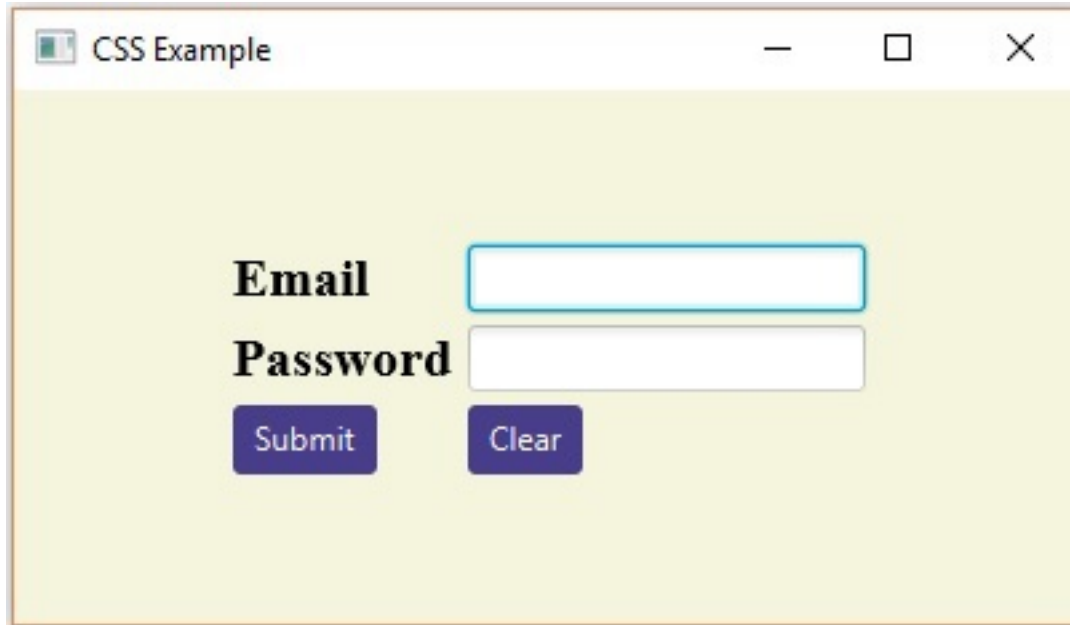
//Displaying the contents of the stage
stage.show();
}
public static void main(String args[]){
    launch(args);
}
}
```

Ví dụ: hiển thị 2 dòng text



Bài tập

- Tìm hiểu Java FX - UI Controls và viết ứng dụng với giao diện như sau



The image shows a Java FX application window titled "CSS Example". The window has a light yellow background and a thin brown border. Inside the window, there is a login form. The form consists of two labels, "Email" and "Password", each followed by a text input field. Below the input fields, there are two buttons: "Submit" and "Clear". The labels and buttons are in a dark blue font, while the input fields have a light blue border. The window also features standard Java FX window controls (minimize, maximize, close) in the top right corner.