

Lập trình JAVA cơ bản



Tuần 8

Giảng viên: Trần Đức Minh

Nội dung bài giảng



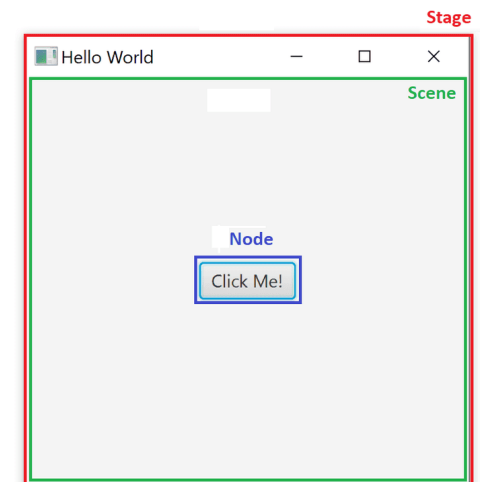
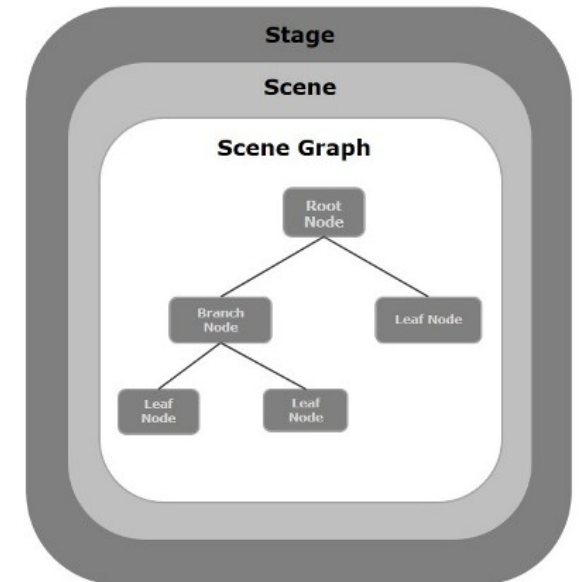
- Cấu trúc của ứng dụng JavaFX
- Giao diện người dùng
 - Các thành phần giao diện người dùng
 - Các lớp bố cục
 - Xử lý hành vi tương tác sự kiện
- Giới thiệu Scene Builder



Cấu trúc của ứng dụng JavaFX



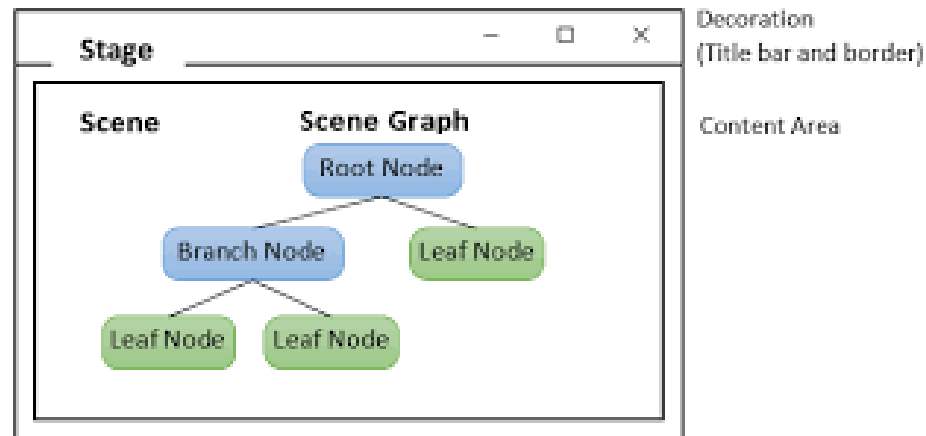
- Ứng dụng JavaFX có 3 thành phần chính
 - **Stage:** Là một cửa sổ làm việc chứa tất cả các đối tượng của ứng dụng JavaFX
 - **Scene:** Nơi trực tiếp chứa các đối tượng của ứng dụng JavaFX
 - **Scene Graph và Nodes**
 - Scene graph là một cấu trúc dữ liệu phân cấp được dùng để mô tả cấu trúc của scene.
 - Node là đối tượng trực quan hay đồ họa của scene.



Stage



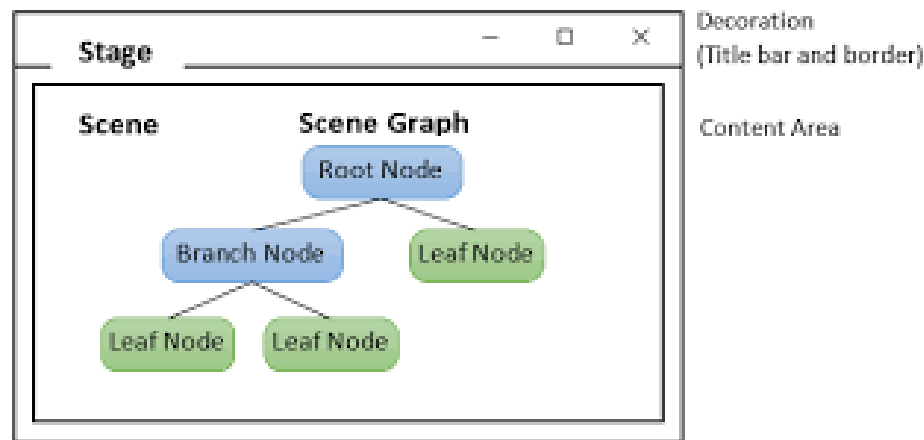
- Stage được mô tả bởi lớp **`javafx.stage.Stage`**
- **Stage chính tự động được** tạo lúc ban đầu và sử dụng làm tham số đầu vào của **phương thức `start()`**
- Stage được chia thành 2 thành phần cơ bản
 - Decorations (Title Bar và đường viền)
 - Vùng nội dung



Scene



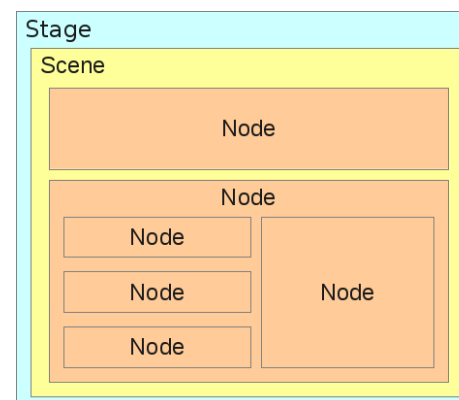
- Scene được mô tả bởi lớp **`javafx.scene.Scene`**
- Scene chứa toàn bộ nội dung của Scene graph
- Một đối tượng scene chỉ được thêm vào duy nhất một stage.



Scene graph



- Trong JavaFX, toàn bộ giao diện người dùng được **mã hóa** bằng cách sử dụng Scene graph.
- Scene graph là **điểm bắt đầu** của việc tạo ra giao diện người dùng.
- Scene graph **chứa các thành phần liên quan đến giao diện người dùng**. Thành phần này được gọi là các Nodes.

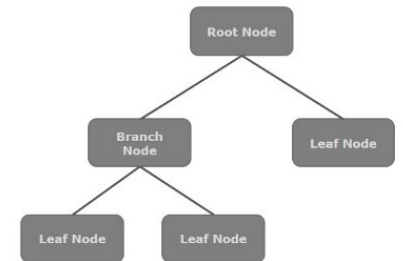


Nodes



- Mỗi Node là một đối tượng trực quan hay đồ họa thuộc các dạng sau:
 - Các đối tượng hình học (2D hay 3D): Hình tròn, hình chữ nhật, đa giác, ...
 - Các điều khiển giao diện người dùng: Button, Checkbox, TextArea, ...
 - Các Containers: Liên quan đến bố cục trình bày như Border Pane, Grid Pane, Flow Pane, ...
 - Các thành phần Media như âm thanh, video và các đối tượng ảnh.
- Node được mô tả bởi lớp **`javafx.scene.Node`** và là lớp cha (lớp cơ sở) của tất cả các nodes.

Nodes



- Tập hợp các nodes tạo nên một scene graph
- Có 3 loại node
 - **Node gốc** (root): Là node không có node cha và bắt buộc phải được đưa vào Scene graph.
 - **Node cha** (parent): là node có chứa node con
 - Parent được mô tả bởi lớp **`javafx.scene.Parent`** và là lớp cơ sở của tất cả các node cha. Có 3 loại node cha:
 - **Group**: Chứa một danh sách các node con có thứ tự. Bất kỳ những thay đổi nào áp dụng lên Group thì cũng sẽ được áp dụng lên các node con trong đó.
 - **Region**: Đây là lớp cơ sở của tất cả các node điều khiển giao diện người dùng.
 - **Webview**: Node này quản lý web engine và hiển thị nội dung của nó.
 - **Node lá** (Leaf): là node không chứa node con nào. Như các node phục vụ cho vẽ hình, hiển thị media, ...

Cấu trúc mã nguồn của ứng dụng JavaFX



- Lớp đầu vào của ứng dụng JavaFX cần đảm bảo quy tắc sau
 - Kế thừa từ lớp **javafx.application.Application**
 - Thực hiện **nạp chồng phương thức start()** của lớp **Application**.
- Phương thức **main()** gọi đến phương thức **launch()** để bắt đầu chạy ứng dụng JavaFX.
 - Phương thức **launch()** sẽ tự động gọi đến phương thức **start()** sau khi tự động thiết lập một số thông số cho chương trình.

Cấu trúc mã nguồn của ứng dụng JavaFX



```
public class MainClass extends Application {  
  
    public void start(Stage primaryStage) throws Exception {  
        ...  
    }  
  
    public static void main(String args[]) {  
        launch(args);  
    }  
}
```

Cấu trúc mã nguồn của ứng dụng JavaFX



- Các bước cần thực hiện bên trong phương thức start()
 - **Thiết lập Scene Graph** bằng cách đẩy các node vào cấu trúc.
 - **Thiết lập Scene** bằng cách đặt kích thước cửa sổ làm việc và đưa Scene Graph vào trong **thông qua Node gốc**.
 - **Thiết lập Stage** rồi đưa Scene vào trong, sau đó hiển thị nội dung của Stage.

Ví dụ

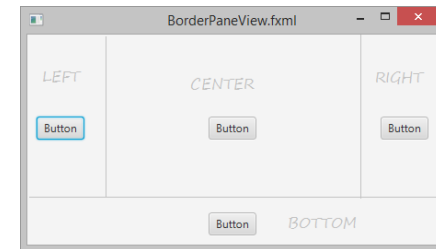


```
public class MainClass extends Application {  
    public static void main(String[] args) {  
        launch(args);  
    }  
    public void start(Stage primaryStage) {  
        Button button = new Button();  
        button.setText("Chọn vào đây");  
        StackPane layout = new StackPane();  
        layout.getChildren().add(button);  
  
        Scene scene = new Scene(layout, 250, 250);  
  
        primaryStage.setTitle("Hello World");  
        primaryStage.setScene(scene);  
        primaryStage.show();  
    }  
}
```

Giao diện người dùng



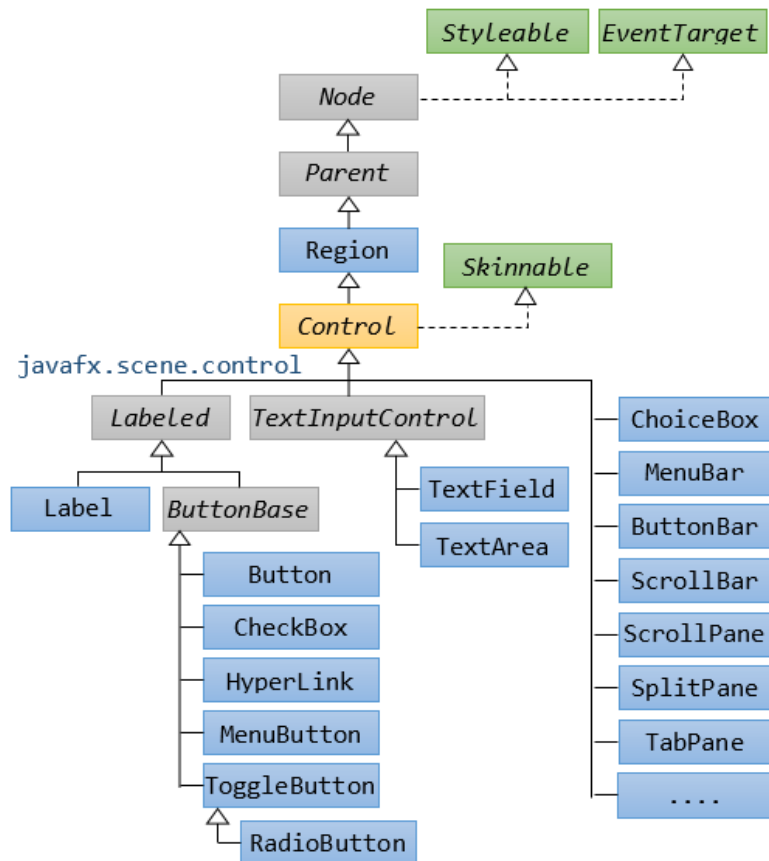
- Giao diện người dùng đều được xây dựng dựa trên 3 khía cạnh
 - **Các thành phần giao diện người dùng (UI elements):** Đó là các điều khiển (controls) mà người dùng có thể thấy và tương tác với nó (Button, TextField, Checkbox, ...)
 - **Các bố cục (layouts):** Cho chúng ta biết các điều khiển sẽ được sắp xếp như thế nào trên màn hình.
 - **Hành vi (behavior):** Đó là các sự kiện xuất hiện khi người sử dụng tương tác với các điều khiển (click chuột vào một nút, chọn vào danh sách list, ...)



Các thành phần giao diện người dùng

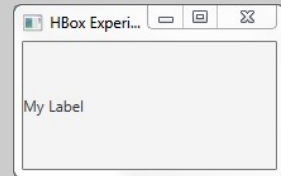

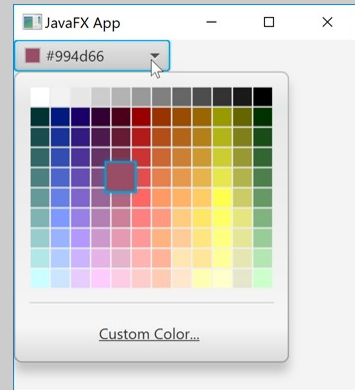
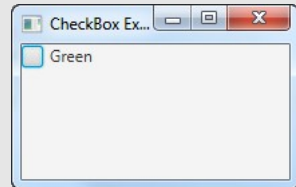


- Các lớp dùng để tạo các điều khiển đều nằm trong gói **javafx.scene.control**



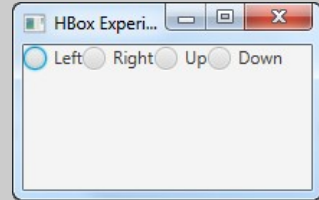
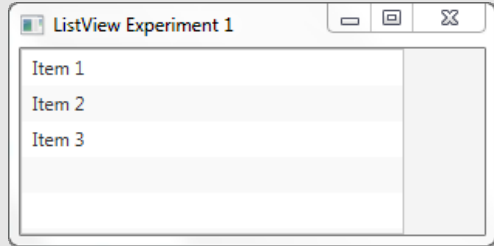
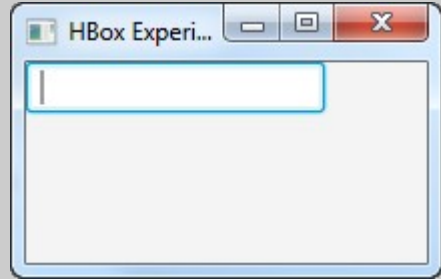
Các thành phần giao diện người dùng



Tên lớp	Mô tả	Hình ảnh
Label	Dùng để đặt các dòng text vào bên trong	
Button	Tạo ra nút bấm	
ColorPicker	Tạo ra một bảng điều khiển để người sử dụng thao tác chọn màu.	
CheckBox	Tạo ra một điều khiển lựa chọn đúng/sai hoặc bật/tắt	

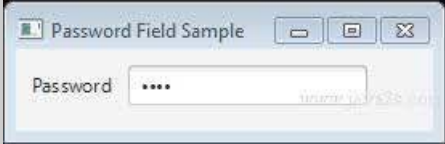
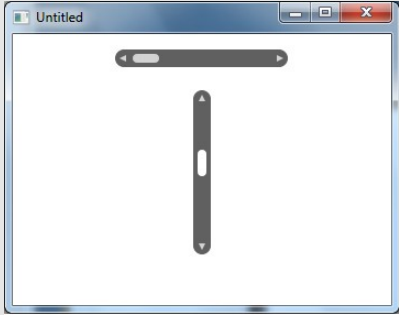
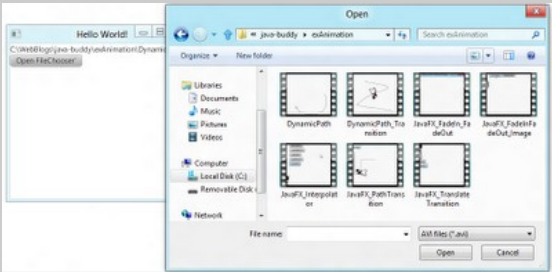
Các thành phần giao diện người dùng



Tên lớp	Mô tả	Hình ảnh
RadioButton	Tạo ra một điều khiển lựa chọn đúng/sai hoặc bật/tắt bên trong một nhóm trạng thái	
ListView	Tạo ra một danh sách các dòng text để người sử dụng lựa chọn	
TextField	Tạo ra một ô cho phép người dùng nhập vào một dòng text	

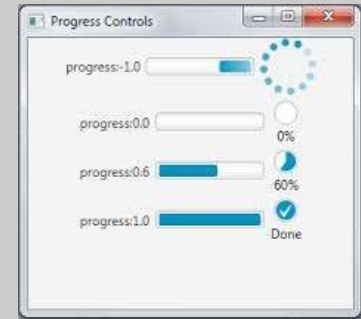
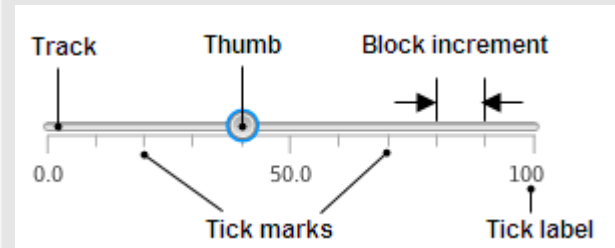
Các thành phần giao diện người dùng



Tên lớp	Mô tả	Hình ảnh
PasswordField	Tạo ra một ô cho phép người dùng nhập vào mật khẩu	
Scrollbar	Cho phép người dùng lựa chọn một khoảng giá trị	
FileChooser	Tạo ra một cửa sổ hội thoại để người dùng lựa chọn file	

Các thành phần giao diện người dùng



Tên lớp	Mô tả	Hình ảnh
ProgressBar	Tạo ra một thanh hiển thị phần trăm hoàn thành công việc	
Slider	Cho phép người dùng chọn giá trị bằng một núm trên thanh trượt	

Ví dụ



```
public void start(Stage primaryStage) {  
    Text text1 = new Text("Email");  
    Text text2 = new Text("Password");  
    TextField textField1 = new TextField();  
    PasswordField textField2 = new PasswordField();  
    Button button1 = new Button("Submit");  
    Button button2 = new Button("Clear");  
  
    button1.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white;");  
    button2.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white;");  
    text1.setStyle("-fx-font: normal bold 20px 'serif' ");  
    text2.setStyle("-fx-font: normal bold 20px 'serif' ");  
  
    GridPane gridPane = new GridPane();  
    gridPane.setMinSize(400, 200);  
    gridPane.setPadding(new Insets(10, 10, 10, 10));  
    gridPane.setVgap(5);  
    gridPane.setHgap(5);  
    gridPane.setAlignment(Pos.CENTER);
```

Ví dụ



```
gridPane.add(text1, 0, 0);
gridPane.add(textField1, 1, 0);
gridPane.add(text2, 0, 1);
gridPane.add(textField2, 1, 1);
gridPane.add(button1, 0, 2);
gridPane.add(button2, 1, 2);

gridPane.setStyle("-fx-background-color: BEIGE;");

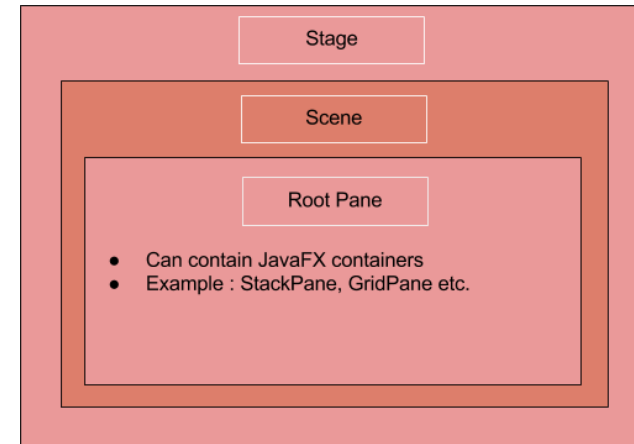
Scene scene = new Scene(gridPane);

primaryStage.setTitle("CSS Example");
primaryStage.setScene(scene);
primaryStage.show();
}
```

Bố cục

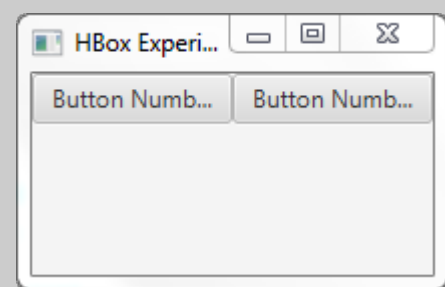
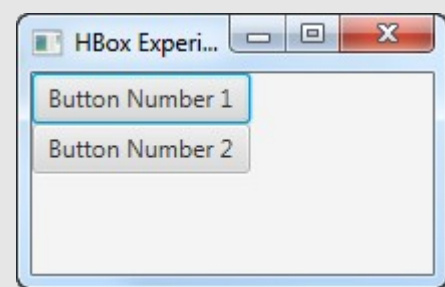
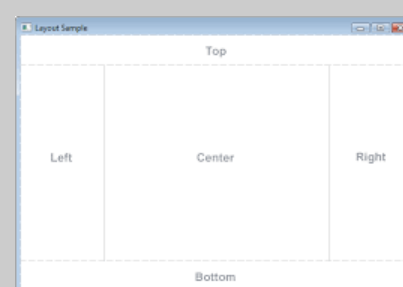


- Việc **sắp xếp các điều khiển** bên trong một **container** (khung chứa) được gọi là **bố cục** (layout) của khung chứa.
- Tất cả các lớp bố cục đều thuộc về gói **`javafx.scene.layout`**
- Lớp có tên là **Pane** là **lớp cơ sở** của tất cả các lớp bố cục.
- Để tạo một bố cục ta cần thực hiện 4 bước sau:
 - Tạo node
 - Khởi tạo đối tượng cho một bố cục đã lựa chọn
 - Thiết lập các thuộc tính cho bố cục
 - Thêm các node đã được tạo đến bố cục



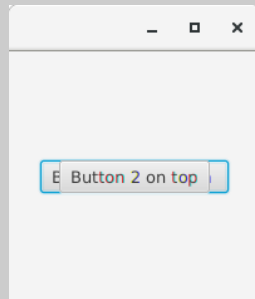
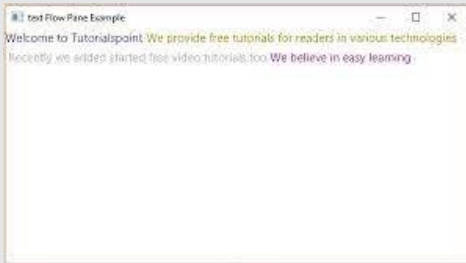
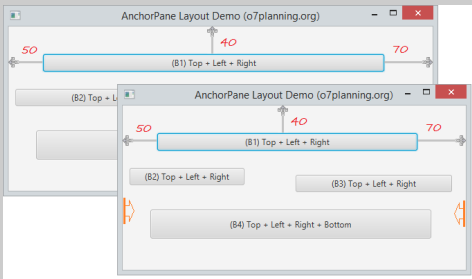
Các lớp bố cục



Tên lớp	Mô tả	Hình ảnh
HBox	Tạo ra một bố cục trên một hàng ngang	
VBox	Tạo ra một bố cục trên một hàng dọc	
BorderPane	Sắp xếp các node ở các vị trí trên , trái , phải , dưới và trung tâm	

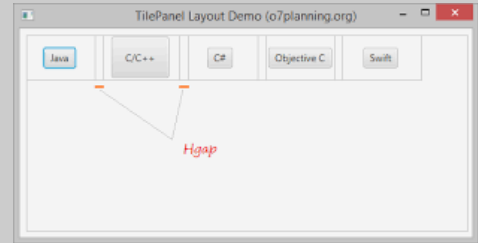

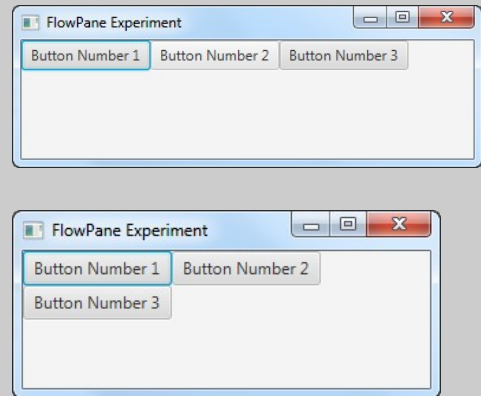
Các lớp bố cục



Tên lớp	Mô tả	Hình ảnh
StackPane	Sắp xếp node nọ chồng lên node kia theo dạng ngăn xếp, tức là node nào được đưa vào sau sẽ được đặt ở bên trên.	
TextFlow	Sắp xếp các node Text liên tiếp nhau theo luồng thông tin	
AnchorPane	Đặt các node ở một khoảng cách cụ thể tính từ khung	

Các lớp bố cục



Tên lớp	Mô tả	Hình ảnh
TilePane	Đưa node vào các ô có kích thước đồng nhất.	
GridPane	Sắp xếp các node dưới dạng lưới hàng và cột.	
FlowPane	Cho phép các node tự động căn chỉnh nếu không đủ khoảng trống. Nếu một dòng hết chỗ trống, node sẽ được đưa xuống dòng tiếp theo.	

Ví dụ



```
Button button1 = new Button("Button Number 1");  
Button button2 = new Button("Button Number 2");  
Button button3 = new Button("Button Number 3");
```

```
FlowPane flowpane = new FlowPane();  
flowpane.getChildren().add(button1);  
flowpane.getChildren().add(button2);  
flowpane.getChildren().add(button3);
```

```
Scene scene = new Scene(flowpane, 200, 100);
```

```
primaryStage.setTitle("Chạy FlowPane");  
primaryStage.setScene(scene);  
primaryStage.show();
```

Xử lý hành vi tương tác sự kiện



- Khi ta thực hiện các thao tác như **click lên một button**, **di chuyển chuột**, **nhập ký tự** từ bàn phím, **chọn một mục** từ điều khiển list, **cuộn trang**, ... thì lúc một sự kiện (event) sẽ xảy ra.
- Lớp có tên là **Event** của gói **javafx.event** là **lớp cơ sở** của các lớp sự kiện.
- Các loại sự kiện trong JavaFX
 - **Sự kiện chuột**: Đại diện bởi một lớp có tên là **MouseEvent**, nó chứa các hành động liên quan đến chuột như click, di chuyển chuột, ...
 - **Sự kiện bàn phím**: Đại diện bởi một lớp có tên là **KeyEvent**, nó chứa các hành động liên quan đến việc nhấn, thả và gõ phím.
 - **Sự kiện kéo thả**: Đại diện bởi một lớp có tên là **DragEvent**, nó chứa các hành động liên quan đến việc bắt kéo chuột, thả chuột,
 - **Sự kiện cửa sổ**: Đại diện bởi một lớp có tên là **WindowEvent**, nó chứa các hành động liên quan đến hiện, ẩn cửa sổ,

Xử lý hành vi tương tác sự kiện



- Xử lý sự kiện (Event handling)
 - Là cơ chế kiểm soát sự kiện. Mỗi khi một sự kiện nào đó xảy ra, hệ thống sẽ tự động gọi đến một đoạn mã nguồn liên quan đến sự kiện đó (đã được chỉ định trước) để thực thi.
 - Mỗi sự kiện trong JavaFX có chứa 3 thành phần được dùng để xử lý và lọc sự kiện
 - **Target:** Là một node mà trong đó sự kiện xuất hiện.
 - Ví dụ: Cửa sổ, scene, button, ...
 - **Source:** Nơi mà từ đó sự kiện được sinh ra.
 - Ví dụ: Chuột, bàn phím, ...
 - **Type:** Loại sự kiện xuất hiện.
 - Ví dụ: nhấn chuột, nhấn phím, ...
 - Một node có thể đăng ký nhiều hơn 1 loại sự kiện.

Ví dụ: Xử lý sự kiện chuột



```
public void start(Stage primaryStage) {  
    Label label = new Label();  
    FlowPane flowpane = new FlowPane();  
    flowpane.getChildren().add(label);  
    Scene scene = new Scene(flowpane, 500, 500);  
    scene.setOnMouseClicked(new EventHandler<MouseEvent>() {  
        public void handle(MouseEvent mouseEvent) {  
            System.out.println("X = " + mouseEvent.getX() + "; Y = " + mouseEvent.getY());  
        }  
    });  
    scene.setOnMouseMoved(new EventHandler<MouseEvent>() {  
        public void handle(MouseEvent mouseEvent) {  
            label.setText("X = " + mouseEvent.getX() + "; Y = " + mouseEvent.getY());  
        }  
    });  
    primaryStage.setTitle("Vị trí chuột");  
    primaryStage.setScene(scene);  
    primaryStage.show();  
}
```

Ví dụ: Xử lý sự kiện Button (bổ sung code cho ví dụ trong Slide 19)

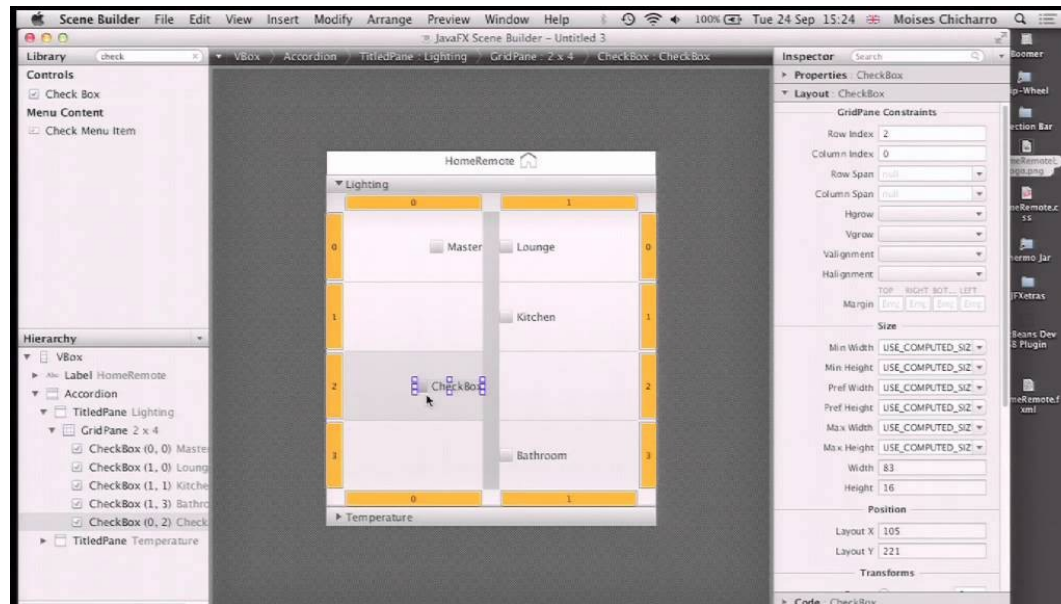


```
button1.setOnMouseClicked(new EventHandler<MouseEvent>() {  
    public void handle(MouseEvent mouseEvent) {  
        Alert alert = new Alert(Alert.AlertType.CONFIRMATION);  
        alert.setTitle("Xác nhận đăng nhập");  
        alert.setHeaderText("Thông tin cảnh báo");  
        alert.setContentText("Hãy lựa chọn");  
  
        ButtonType buttonNo = new ButtonType("No", ButtonBar.ButtonData.NO);  
        ButtonType buttonYes = new ButtonType("Yes", ButtonBar.ButtonData.YES);  
  
        alert.getButtonTypes().setAll(buttonNo, buttonYes);  
  
        Optional<ButtonType> result = alert.showAndWait();  
  
        if (result.get() == buttonYes) {  
            System.out.println("Yes được chọn");  
        } else if (result.get() == buttonNo) {  
            System.out.println("No được chọn");  
        }  
    }  
});
```

Giới thiệu Scene Builder



- Scene Builder là một sản phẩm được dùng để hỗ trợ JavaFX thiết kế giao diện cho thuận tiện.
- Đầu ra của Scene Builder là file có đuôi .fxml



Hết Tuần 8



Cảm ơn các bạn đã chú ý lắng nghe !!!