

## Một số thành phần UI cơ bản trong JavaFX

Trong phần này, chúng tôi tập trung giới thiệu một số thành phần giao diện tương tác (UI Control) thừa kế từ lớp `javafx.scene.control.Control`.

Các mô tả về một thành phần giao diện bao gồm các lớp cha, các thuộc tính, phương thức của chính lớp đó hay thừa kế từ các lớp cha người lập trình cần tra cứu trong JavaFX API. Thông thường để hiểu và sử dụng một lớp, ta trước hết cần biết các thuộc tính của chính lớp đó, các thuộc tính kế thừa từ các lớp cha, căn cứ vào các thuộc tính đó ta sẽ dễ dàng tra ra các phương thức tác động vào thông qua các tham chiếu phương thức đính kèm của thuộc tính.

### Ví dụ:

Các thành phần giao diện tương tác thừa kế từ lớp `javafx.scene.control.Control` thuộc cây thừa kế:

`javafx.scene.Node`

`javafx.scene.Parent`

`javafx.scene.layout.Region`

`javafx.scene.control.Control`

Từ đó ta có thể liệt kê được một số thuộc tính, phương thức chung sau:

**Bảng một số thuộc tính, phương thức chung của các thành phần giao diện tương tác**

|             | Thuộc tính  | Phương thức   |
|-------------|---|---|
| Lớp Control | tooltip   | getTooltip, setTooltip  |
| Lớp Region  | background, border, width, height, insets, maxWidth, maxHeight, minWidth, minHeight, padding  | getBackground, setBackground, getBorder, setBorder, getHeight, setHeight, getPadding, setPadding...   |
| Lớp Parent  | needsLayout   | isNeedsLayout, setNeedsLayout   |
| Lớp Node    | id, disable, cursor, effect, focused, opacity, rotate, scene, style, visible, scaleX, scaleY, scaleZ, onDragEntered, onKeyPressed, onMouseClicked, onScroll, onZoom | getId, setId, isDisable, setDisable, getEffect, setEffect, getStyle, getScene, setStyle, isVisible, setVisible, setOnDragEntered, setOnKeyPressed, setOnMouseClicked... |

### ▪ **JavaFX Label**

Label (Nhãn) trong JavaFX được sử dụng để hiển thị nội dung văn bản, hình ảnh, ký tự, nó không cho phép người dùng chỉnh sửa trực tiếp nội dung.

Các thuộc tính, constructor, phương thức của lớp `javafx.scene.control.Label`:

**Bảng các thuộc tính, constructor, phương thức của lớp `javafx.scene.control.Label`**

| Thuộc tính | Constructor/Phương thức         | Mô tả                                     |
|------------|---------------------------------|---|
|            | <code>Label()</code>            | Tạo nhãn không có nội dung                |
|            | <code>Label(String text)</code> | Tạo nhãn với nội dung là chuỗi truyền vào |

|          |                                  |  |
|----------|----------------------------------|--|
|          | Label(String text, Node graphic) | Tạo nhãn với nội dung là chuỗi truyền vào và một Node đồ họa                 |
| labelFor | getLabelFor()                    | Lấy giá trị thuộc tính labelFor  |
|          | setLabelFor(Node value)          | Gắn nhãn với một Node. Thường dùng để có thể tương tác với Node qua bàn phím |

Một số thuộc tính, phương thức của lớp Label thừa kế từ lớp javafx.scene.control.Labeled:

**Bảng các thuộc tính, constructor, phương thức của lớp Label thừa kế từ lớp javafx.scene.control.Labeled**

| Thuộc tính | Phương thức                 | Mô tả  |
|------------|-----------------------------|--|
| text       | setText(String value)       | Thiết lập nội dung cho nhãn                  |
|            | getText()                   | Lấy giá trị nội dung văn bản của nhãn        |
| graphic    | setGraphic(Node value)      | Thiết lập hình ảnh hiển thị                  |
| alignment  | setAlignment(Pos value)     | Thiết lập vị trí hiển thị.                   |
| font       | setFont(Font value)         | Thiết lập font chữ cho nhãn                  |
| underLine  | setUnderline(boolean value) | Thiết lập kiểu gạch chân                     |
| wrapText   | setWrapText(boolean value)  | Thiết lập chế độ tự xuống dòng nội dung nhãn |
| textFill   | setTextFill(Paint value)    | Thiết lập màu chữ                            |

**Ví dụ:**

```
import java.io.File;
import java.io.FileInputStream;
import javafx.application.Application;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Stage;

public class JavaFXLabel extends Application {

    @Override
    public void start(Stage stage) throws Exception{
        Label label1 = new Label();
        label1.setGraphic(new ImageView(new Image(new FileInputStream(new
        File("E:\\Hình Ảnh\\fita-logo.png")))));
        label1.setMaxWidth(55);
        label1.setAlignment(Pos.CENTER);

        Label label2 = new Label();
```

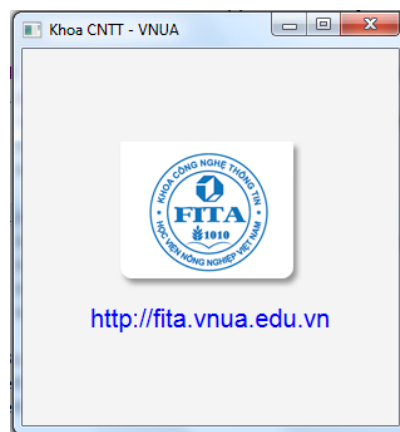
```
label2.setText("http://fita.vnua.edu.vn");
label2.setTextFill(Color.BLUE);
label2.setFont(new Font("Arial", 20));
label2.setAlignment(Pos.CENTER);

// VBox: layout sắp các thành phần chứa bên trong theo chiều dọc
VBox vbox = new VBox(label1, label2);
vbox.setSpacing(12);
vbox.setAlignment(Pos.CENTER);

Scene scene = new Scene(vbox, 300, 300);
stage.setTitle("Khoa CNTT - VNUA");
stage.setScene(scene);
stage.show();
}

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

Kết quả chạy chương trình:



## JavaFX Button

Trong JavaFX lớp Button (Nút nhấn) được sử dụng để xử lý một hành động khi người dùng nhấp vào một nút nào đó trên giao diện. Nhãn của Button có thể là văn bản, đồ họa hay cả hai.

Một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.Button:

**Bảng một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.Button**

| Thuộc tính | Constructor/Phương thức           | Mô tả  |
|------------|-----------------------------------|--|
|            | Button()                          | Tạo nhãn không có nội dung                                       |
|            | Button(String text)               | Tạo nút nhấn với nhãn là chuỗi truyền vào                        |
|            | Button(String text, Node graphic) | Tạo nút nhấn với nội dung là chuỗi truyền vào và một Node đồ họa |

|               |                                 |   |
|---------------|---------------------------------|---|
| cancelButton  | isCancelButton()                | Kiểm tra một nút nhấn có thuộc kiểu “cancel button” không   |
|               | setCancelButton(boolean value)  | Thiết lập cho một nút nhấn có khả năng nhận sự kiện phím ESC.<br>Ví dụ sau sẽ in ra Console thông điệp “Cancel clicked” khi bấm phím ESC:<br><br><pre>Button cancelButton = new Button("Cancel"); cancelButton.setCancelButton(true); cancelButton.setOnAction(e -&gt; {     System.out.println("Cancel clicked."); });</pre> |
| defaultButton | isDefaultButton()               | Kiểm tra một nút nhấn có thuộc kiểu “default button” không  |
|               | setDefaultButton(boolean value) | Thiết lập cho một nút nhấn có khả năng nhận sự kiện phím Enter.   |

Một số thuộc tính, phương thức của lớp Button thừa kế từ lớp javafx.scene.control.ButtonBase:

**Bảng một số thuộc tính, phương thức của lớp Button thừa kế từ lớp javafx.scene.control.ButtonBase**

| Thuộc tính | Constructor/Phương thức                      | Mô tả  |
|------------|--|--|
| onAction   | getOnAction()                                | Lấy giá trị thuộc tính onAction                  |
|            | setOnAction(EventHandler<ActionEvent> value) | Xử lý sự kiện khi nút nhấn được kích hoạt (fire) |

Một số thuộc tính, phương thức của lớp Button thừa kế từ lớp javafx.scene.control.Labeled tương tự như thành phần Label.

### **Ví dụ:**

(Xem ví dụ minh họa JavaFX ở đầu chương)

### **JavaFX CheckBox**

Trong JavaFX CheckBox thường được sử dụng trong trường hợp cho phép lựa chọn nhiều tùy chọn cùng một lúc, mỗi CheckBox trong JavaFX có thể phản ánh 2 trạng thái (có/không) hoặc 3 trạng thái (có/không/không xác định) tùy thuộc việc cấu hình.

Một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.CheckBox:

**Bảng một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.CheckBox**

| Thuộc tính    | Constructor/Phương thức         | Mô tả   |
|---------------|---------------------------------|---|
|               | CheckBox()                      | Tạo CheckBox không có nhãn  |
|               | CheckBox(String text)           | Tạo CheckBox với nhãn là chuỗi truyền vào                             |
| selected      | isSelected()                    | Kiểm tra một CheckBox có được check hay không                         |
|               | setSelected(boolean value)      | Thiết lập trạng thái check hay không check cho CheckBox               |
| indeterminate | isIndeterminate()               | Kiểm tra một CheckBox có đang ở trạng thái “không xác định” hay không |
|               | setIndeterminate(boolean value) | Thiết lập trạng thái “không xác định” cho CheckBox                    |

|                    |                                      |   |
|--------------------|--------------------------------------|---|
| allowIndeterminate | isAllowIndeterminate()               | Kiểm tra một CheckBox có cho phép trạng thái “không xác định” hay không   |
|                    | setAllowIndeterminate(boolean value) | Thiết lập CheckBox cho phép trạng thái “không xác định” xuất hiện. Ví dụ:<br>CheckBox cbox = new CheckBox("Indeterminate CheckBox");<br>cbox.setAllowIndeterminate(true); |

Một số thuộc tính, phương thức của lớp CheckBox thừa kế từ lớp javafx.scene.control.ButtonBase, javafx.scene.control.Labeled tương tự như lớp javafx.scene.control.Button

### Ví dụ:

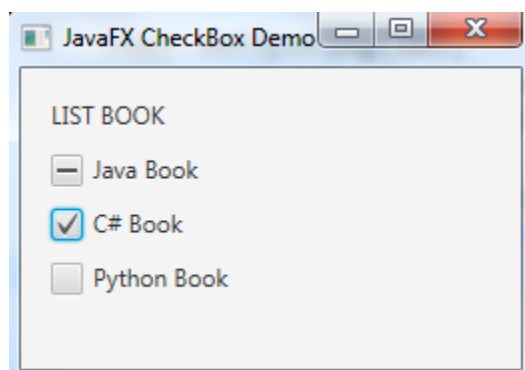
```
@Override
public void start(Stage stage) {
    VBox root = new VBox();

    Label label = new Label("LIST BOOK");
    CheckBox checkBox1 = new CheckBox("Java Book");
    checkBox1.setAllowIndeterminate(true);
    CheckBox checkBox2 = new CheckBox("C# Book");
    checkBox2.setAllowIndeterminate(true);
    CheckBox checkBox3 = new CheckBox("Python Book");
    checkBox3.setAllowIndeterminate(true);

    root.setSpacing(10);
    root.setPadding(new Insets(15));
    root.getChildren().addAll(label, checkBox1, checkBox2, checkBox3);

    Scene scene = new Scene(root, 200, 200);
    stage.setTitle("JavaFX CheckBox Demo");
    stage.setScene(scene);
    stage.show();
}
```

Bạn đọc tự hoàn thiện thêm khung chương trình căn bản như các ví dụ trước. Kết quả chạy chương trình:



Nội dung các CheckBox trong ví dụ cho biết, trong danh sách hiện đã có cuốn sách C#, cuốn Python chưa có và cuốn Java chưa xác định được đã có hay chưa.

### ▪ JavaFX ToggleButton

ToggleButton là một loại nút nhấn có 2 trạng thái, được lựa chọn hoặc không được lựa chọn. Các ToggleButton có thể được đặt trong một nhóm (Toggle Group), các ToggleButton trong cùng một nhóm tại một thời điểm chỉ có nhiều nhất một nút được chọn, nếu một nút nhấn được chọn, các nút nhấn khác trong nhóm sẽ bị mất lựa chọn.

Một số thuộc tính, constructor, phương thức của lớp `javafx.scene.control.ToggleButton`:

**Bảng một số thuộc tính, constructor, phương thức của lớp `javafx.scene.control.ToggleButton`**

| Thuộc tính  | Constructor/Phương thức                              | Mô tả  |
|-------------|--|--|
|             | <code>ToggleButton()</code>                          | Tạo ToggleButton không có nhãn   |
|             | <code>ToggleButton (String text)</code>              | Tạo ToggleButton với nhãn là chuỗi truyền vào                                    |
|             | <code>ToggleButton(String text, Node graphic)</code> | Tạo ToggleButton với nội dung là chuỗi truyền vào và một Node đồ họa             |
| selected    | <code>isSelected()</code>                            | Kiểm tra một ToggleButton có được chọn hay không                                 |
|             | <code>setSelected(boolean value)</code>              | Thiết lập trạng thái được chọn hay không cho ToggleButton                        |
| toggleGroup | <code>getToggleGroup()</code>                        | Lấy giá trị thuộc tính toggleGroup cho biết ToggleButton hiện tại thuộc nhóm nào |
|             | <code>setToggleGroup(ToggleGroup value)</code>       | Gán ToggleButton vào một nhóm  |

Lớp ToggleButton cũng thừa kế các thuộc tính, phương thức từ lớp `javafx.scene.control.ButtonBase`, `javafx.scene.control.Labeled`.

Ví dụ:

```
@Override
public void start(Stage stage) {
    HBox root = new HBox();
    root.setPadding(new Insets(10));
    root.setSpacing(5);

    // Tạo ToggleGroup
    ToggleGroup group = new ToggleGroup();

    ToggleButton maleBtn = new ToggleButton("Nam");
    ToggleButton femaleBtn = new ToggleButton("Nữ");

    // Đặt các ToggleButton vào nhóm.
    maleBtn.setToggleGroup(group);
    femaleBtn.setToggleGroup(group);

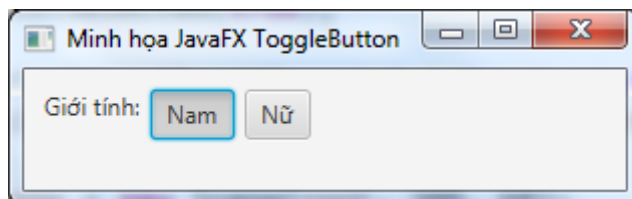
    // Đặt lựa chọn mặc định
    maleBtn.setSelected(true);

    root.getChildren().add(new Label("Giới tính:"));
    root.getChildren().addAll(maleBtn, femaleBtn);
}
```

```
Scene scene = new Scene(root, 300, 60);

stage.setTitle("Minh họa JavaFX ToggleButton");
stage.setScene(scene);
stage.show();
}
```

Kết quả chạy chương trình:



### ▪ JavaFX RadioButton

RadioButton trong JavaFX được sử dụng để cho phép người dùng thực hiện chỉ duy nhất một lựa chọn trong một nhóm danh sách lựa chọn. Muốn nhóm nhiều RadioButton lại với nhau chúng ta cần sử dụng ToggleGroup.

Các constructor của lớp `javafx.scene.control.ToggleButton`:

**Bảng 7.8. Các constructor của lớp `javafx.scene.control.ToggleButton`**

| Constructor                           | Mô tả  |
|---------------------------------------|--|
| <code>RadioButton()</code>            | Tạo RadioButton không có nhãn                |
| <code>RadioButton(String text)</code> | Tạo RadioButton với nhãn là chuỗi truyền vào |

Lớp `RadioButton` cũng thừa kế các thuộc tính, phương thức từ lớp `javafx.scene.control.ToggleButton`, `javafx.scene.control.ButtonBase`, `javafx.scene.control.Labeled`.

### Ví dụ:

```
@Override
public void start(Stage stage) {
    HBox root = new HBox();

    // Tạo các RadioButton và thêm vào nhóm
    ToggleGroup group = new ToggleGroup();
    RadioButton button1 = new RadioButton("Green");
    button1.setToggleGroup(group);
    RadioButton button2 = new RadioButton("Yellow");
    button2.setToggleGroup(group);
    RadioButton button3 = new RadioButton("Red");
    button3.setToggleGroup(group);

    // Thiết lập mặc định
    button1.setSelected(true);
    root.setStyle("-fx-background-color: #00FF00;");

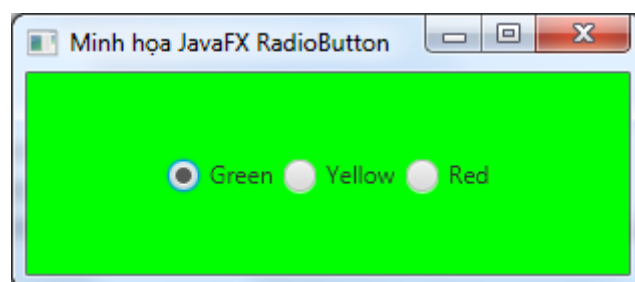
    // Xử lý sự kiện thay đổi lựa chọn RadioButton trong group
}
```

```
group.selectedToggleProperty().addListener(new
ChangeListener<Toggle>() {
    public void changed(ObservableValue<? extends Toggle> ov,
        Toggle old_toggle, Toggle new_toggle) {
        if (group.getSelectedToggle() != null) {
            RadioButton selectedRadio = (RadioButton)
group.getSelectedToggle();
            switch (selectedRadio.getText()) {
                case "Green":
                    root.setStyle("-fx-background-color:
#00FF00;");
                    break;
                case "Yellow":
                    root.setStyle("-fx-background-color:
#FFFF00;");
                    break;
                case "Red":
                    root.setStyle("-fx-background-color:
#FF0000;");
                    break;
                default:
                    root.setStyle("-fx-background-color:
#FAF8CC;");
                    break;
            }
        }
    }
});

root.setSpacing(5);
root.setAlignment(Pos.CENTER);
root.getChildren().addAll(button1, button2, button3);

Scene scene = new Scene(root, 300, 100);
stage.setTitle("Minh họa JavaFX RadioButton");
stage.setScene(scene);
stage.show();
}
```

Kết quả chạy chương trình:



#### ▪ JavaFX ChoiceBox



Lớp ChoiceBox trong JavaFX được sử dụng để trình bày cho người dùng một tập hợp các lựa chọn được xác định trước và người dùng chỉ được đưa ra một lựa chọn. Theo mặc định ChoiceBox không có mục nào được chọn trừ khi được cài đặt trước.

Một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.ChoiceBox:

**Bảng một số thuộc tính, constructor, phương thức của lớp javafx.scene.control.ChoiceBox**

| Thuộc tính     | Constructor/Phương thức                          | Mô tả  |
|----------------|--|--|
|                | ChoiceBox()                                      | Tạo ChoiceBox với danh sách lựa chọn rỗng  |
|                | ChoiceBox(ObservableList<T> items)               | Tạo ChoiceBox với danh sách lựa chọn truyền vào. Danh sách gắn kèm ChoiceBox là một danh sách các đối tượng nào đó, giá trị được hiển thị trên ChoiceBox tùy thuộc vào giá trị trả lại của phương thức toString trong cài đặt của đối tượng đó |
| items          | getItems()                                       | Lấy danh sách lựa chọn gắn với ChoiceBox   |
|                | setItems(ObservableList<T> value)                | Thiết lập danh sách lựa chọn cho ChoiceBox   |
| selectionModel | getSelectionModel()                              | Lấy giá trị thuộc tính selectionModel. Thuộc tính này là mô hình quản lý dữ liệu của ChoiceBox, nó chỉ cho phép chọn một lựa chọn, cho phép thiết lập lựa chọn mặc định hoặc theo dõi sự lựa chọn từ danh sách                                 |
|                | setSelectionModel(SingleSelectionModel<T> value) | Thiết lập giá trị cho thuộc tính selectionModel  |
| value          | getValue()                                       | Lấy giá trị thuộc tính value (thuộc tính ứng với giá trị đang được chọn trên ChoiceBox) là một thể hiện của đối tượng T trong ObservableList<T>  |
|                | setValue(T)                                      | Gán giá trị cho thuộc tính value   |

### Ví dụ:

```
import javafx.application.Application;
import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.ChoiceBox;
import javafx.scene.control.Label;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;

public class ChoiceBoxDemo extends Application {

    @Override
    public void start(Stage primaryStage) {
```

```
ProgrammingLanguage java = new ProgrammingLanguage("PL01", "Java");
ProgrammingLanguage cSharp = new ProgrammingLanguage("PL02", "C#");
ProgrammingLanguage ruby = new ProgrammingLanguage("PL03", "Ruby");
ProgrammingLanguage python = new ProgrammingLanguage("PL04",
"Python");

HBox root = new HBox();

Label label = new Label("Select Programming Language: ");
Label selectedPL = new Label();

ObservableList<ProgrammingLanguage> languages //
    = FXCollections.observableArrayList(cSharp, java, ruby,
python);

ChoiceBox<ProgrammingLanguage> choiceBox = new
ChoiceBox<ProgrammingLanguage>(languages);
// Đặt lựa chọn mặc định
choiceBox.setValue(java);
selectedPL.setText(java.getCode() + "-" + java.getName());

ChangeListener<ProgrammingLanguage> changeListener = new
ChangeListener<ProgrammingLanguage>() {
    @Override
    public void changed(ObservableValue<? extends
ProgrammingLanguage> observable, //
        ProgrammingLanguage oldValue, ProgrammingLanguage
newValue) {
        // Hiển thị mã và tên ngôn ngữ được chọn
        ProgrammingLanguage selected = choiceBox.getValue();
        selectedPL.setText(selected.getCode() + " - " +
selected.getName());
    }
};
// Sự kiện khi thay đổi Item trên ChoiceBox

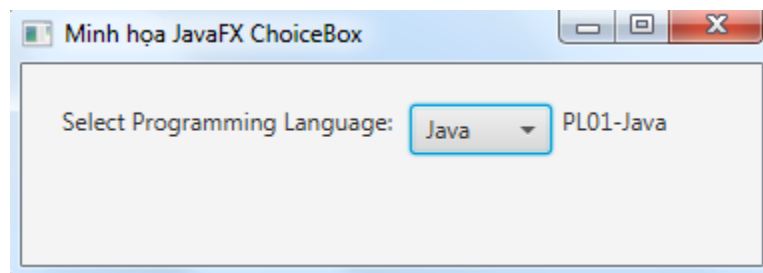
choiceBox.getSelectionModel().selectedItemProperty().addListener(changeList
ener);

root.getChildren().addAll(label, choiceBox, selectedPL);
root.setPadding(new Insets(20));
root.setSpacing(5);

primaryStage.setTitle("Minh họa JavaFX ChoiceBox");
Scene scene = new Scene(root, 370, 100);
primaryStage.setScene(scene);
primaryStage.show();
}
public static void main(String[] args) {
    launch(args);
}
```

```
    }  
}  
  
public class ProgrammingLanguage {  
  
    private String code;  
    private String name;  
  
    public ProgrammingLanguage() {  
    }  
  
    public ProgrammingLanguage(String code, String name) {  
        this.code = code;  
        this.name = name;  
    }  
  
    public String getCode() {  
        return code;  
    }  
  
    public void setCode(String code) {  
        this.code = code;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
  
    @Override  
    public String toString() {  
        return this.name;  
    }  
}  
}
```

Kết quả chạy chương trình:



#### ▪ JavaFX TextField

TextField trong JavaFX được sử dụng để cho phép người dùng nhập một dòng văn bản không sử dụng định dạng. JavaFX TextField thuộc gói `javafx.scene.control`, nó thừa kế từ lớp `javafx.scene.control.TextInputControl`.

Một số thuộc tính, constructor, phương thức của lớp `javafx.scene.control.TextField`:

**Bảng một số thuộc tính, constructor, phương thức của lớp `javafx.scene.control.TextField`**

| Thuộc tính | Constructor/Phương thức   | Mô tả  |
|------------|---|--|
|            | <code>TextField()</code>  | Tạo TextField với dòng text nội dung rỗng  |
|            | <code>TextField(String text)</code>                             | Tạo TextField với dòng text nội dung là chuỗi truyền vào                         |
| alignment  | <code>getAlignment()</code>                                     | Lấy giá trị thuộc tính alignment (thuộc tính căn chỉnh nội dung văn bản trong ô) |
|            | <code>setAlignment(Pos value)</code>                            | Thiết lập giá trị cho thuộc tính alignment                                       |
| onAction   | <code>getOnAction()</code>                                      | Lấy giá trị thuộc tính bắt sự kiện kết thúc nhập text trong ô và bấm Enter       |
|            | <code>setOnAction(EventHandler&lt;ActionEvent&gt; value)</code> | Thiết lập xử lý sự kiện kết thúc nhập text trong ô và bấm Enter                  |

Một số thuộc tính, phương thức thừa kế từ lớp `TextInputControl`:

**Bảng một số thuộc tính, phương thức thừa kế từ lớp `TextInputControl`**

| Thuộc tính    | Phương thức                              | Mô tả   |
|---------------|--|---|
| anchor        | <code>getAnchor()</code>                 | Lấy giá trị số ký tự tính từ đầu đến vị trí con trỏ soạn thảo trừ đi số ký tự đang được bôi đen |
| caretPosition | <code>getCaretPosition()</code>          | Lấy giá trị số ký tự tính từ đầu đến vị trí con trỏ soạn thảo                                   |
| editable      | <code>isEditable()</code>                | Kiểm tra giá trị thuộc tính editable (trạng thái có cho phép chỉnh sửa ô TextField hay không)   |
|               | <code>setEditable(boolean value)</code>  | Thiết lập giá trị cho thuộc tính editable   |
| font          | <code>getFont()</code>                   | Lấy giá trị thuộc tính font chữ cho text  |
|               | <code>setFont(Font value)</code>         | Thiết lập font chữ cho text   |
| length        | <code>getLength()</code>                 | Lấy giá trị độ dài text (số ký tự)  |
| promptText    | <code>getPromptText()</code>             | Lấy giá trị thuộc tính promptText (text chú thích cho ô TextField)                              |
|               | <code>setPromptText(String value)</code> | Thiết lập giá trị cho thuộc tính promptText   |
| selectedText  | <code>getSelectedText()</code>           | Lấy giá trị đoạn text đang được bôi đen trong ô   |
| selection     | <code>getSelection()</code>              | Lấy giá trị vị trí bắt đầu, vị trí kết thúc của đoạn text đang được bôi đen                     |
| text          | <code>getText()</code>                   | Lấy giá trị thuộc tính text (đoạn văn bản chứa trong ô)   |
|               | <code>setText(String value)</code>       | Thiết lập giá trị cho thuộc tính text   |

**Ví dụ:**

```
@Override
public void start(Stage stage) {
    Label label = new Label();
    label.setMaxWidth(300);
    label.setText("ĐĂNG NHẬP TRANG SINH VIÊN");
    label.setAlignment(Pos.CENTER);

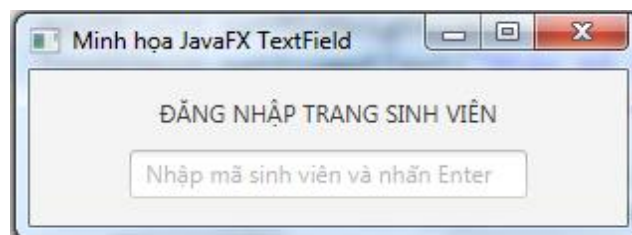
    TextField textField = new TextField();
    textField.setMaxWidth(200);
    textField.setMinHeight(25);
    textField.setPromptText("Nhập mã sinh viên và nhấn Enter");
    // Vô hiệu hóa trạng thái focus mặc định để hiển thị prompt text
    textField.setFocusTraversable(false);

    // Xử lý sự kiện kết thúc nhập và nhấn Enter
    textField.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            System.out.println("Kiểm tra mã sinh viên: " +
textField.getText());
        }
    });

    VBox vBox = new VBox(label, textField);
    vBox.setSpacing(10);
    vBox.setAlignment(Pos.CENTER);

    Scene scene = new Scene(vBox, 300, 80);
    stage.setTitle("Minh họa JavaFX TextField");
    stage.setScene(scene);
    stage.show();
}
```

Kết quả chạy chương trình:



### ▪ JavaFX PasswordField

Password Field trong JavaFX thường được dùng để nhập mật khẩu, các ký tự nhập vào sẽ được ẩn bằng cách hiển thị một chuỗi dấu sao. Lớp PasswordField thuộc gói `javafx.scene.control`, nó thừa kế từ lớp TextField.

Lớp PasswordField chỉ có duy nhất một constructor không tham số, không có thuộc tính riêng, nó ghi đè phương thức cut và copy của lớp TextInputControl để không cho phép cut, copy.

Ví dụ:

```
@Override
    public void start(Stage stage) {

        TextField textField = new TextField();
        textField.setMinWidth(200);
        textField.setPromptText("Nhập mật khẩu...");
        textField.setFocusTraversable(false);
        textField.setLayoutX(20);
        textField.setLayoutY(20);
        textField.setVisible(false);

        // Trường password trùng vị trí, kích thước với trường text
        PasswordField passwordField = new PasswordField();
        passwordField.setMinWidth(200);
        passwordField.setPromptText("Nhập mật khẩu...");
        passwordField.setFocusTraversable(false);
        passwordField.setLayoutX(20);
        passwordField.setLayoutY(20);

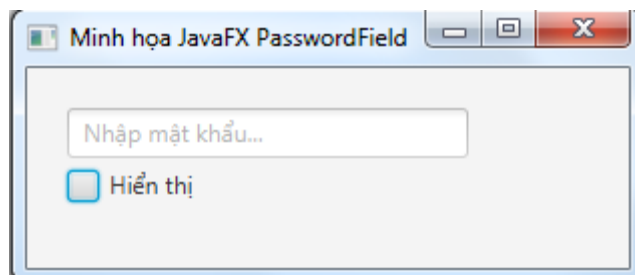
        CheckBox checkBox = new CheckBox();
        checkBox.setText("Hiển thị");
        checkBox.setLayoutX(20);
        checkBox.setLayoutY(50);

        // Xử lý sự kiện click vào checkbox
        checkBox.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                boolean selected = checkBox.isSelected();
                // Điều khiển việc ẩn hiện của trường password,
                textField.setVisible(selected);
                passwordField.setVisible(!selected);
                // Copy nội dung passwordfield -> textfield
                if(selected) {
                    textField.setText(passwordField.getText());
                } else { // Copy nội dung textfield -> passwordfield
                    passwordField.setText(textField.getText());
                }
            }
        });

        AnchorPane anchorPane = new AnchorPane(textField, passwordField,
        checkBox);
```

```
Scene scene = new Scene(anchorPane, 300, 100);
stage.setTitle("Minh họa JavaFX PasswordField");
stage.setScene(scene);
stage.show();
}
```

Kết quả chạy chương trình:



### ■ JavaFX TableView

JavaFX cung cấp lớp TableView, nó được sử dụng cùng với TableColumn và TableCell giúp bạn hiển thị dữ liệu dưới dạng bảng. TableColumn ứng với một cột trong bảng, TableCell ứng với một ô trong bảng.

#### + Khởi tạo TableView

```
// Tạo mới TableView hiển thị danh sách đối tượng sinh viên
TableView<Student> table = new TableView<Student>();
```

#### + Thêm cột vào TableView

```
// Tạo cột mã sinh viên, kiểu dữ liệu String
TableColumn<Student, String> codeCol
    = new TableColumn<Student, String>("Msv");
table.getColumns().add(codeCol);
```

Có thể dùng phương thức addAll(col1, col2, ...) để thêm cùng lúc nhiều cột vào bảng.

Có thể nhóm các cột lại với nhau trước khi thêm vào bảng:

```
// Tạo cột FullName (Kiểu dữ liệu String)
TableColumn<Student, String> fullNameCol
    = new TableColumn<Student, String>("Họ tên");

// Tạo 2 cột con cho cột FullName
TableColumn<Student, String> lastNameCol
    = new TableColumn<Student, String>("Họ đệm");
TableColumn<Student, String> firstNameCol
    = new TableColumn<Student, String>("Tên");

// Thêm 2 cột con vào cột FullName
fullNameCol.getColumns().addAll(lastNameCol, firstNameCol);
// Thêm cột FullName vào bảng
table.add(fullNameCol);
```

#### + Hiển thị danh sách dữ liệu lên TableView

- Tạo đối tượng Student với các trường mã sinh viên (code), email, họ đệm (firstName), tên (lastName), lớp (class\_) cùng các phương thức get, set

- Bắt cặp các cột của TableView với các thuộc tính của Student

Ví dụ:

```
codeCol.setCellValueFactory(new PropertyValueFactory<Student, String>("code"));
firstNameCol.setCellValueFactory(new PropertyValueFactory<Student, String>("firstName"));
```

- Chuyển kiểu danh sách đối tượng Student sang kiểu ObservableList<Student>

Ví dụ:

```
List<Student> studentList = getStudentList();
ObservableList<Student> obsStudentList =
FXCollections.observableArrayList(studentList);
```

- Thêm danh sách ObservableList<Student> vào TableView

```
table.setItems(obsStudentList);
```

- + Lấy vị trí dòng đang được chọn trên TableView

```
int selectedIndex = table.getSelectionModel().getSelectedIndex();
```

- + Lấy đối tượng ứng với dòng đang được chọn trên TableView

```
Student selectedItem = table.getSelectionModel().getSelectedItem();
```

Hoặc:

```
Student selectedItem = table.getItems().get(selectedIndex);
```

- + Thêm, cập nhật, xóa dữ liệu một dòng trong dữ liệu của TableView

- Thêm một hoặc một vài dòng:

```
table.getItems().add(student1);
table.getItems().addAll(student1, student2, ...);
```

- Cập nhật:

```
table.getItems().set(<chỉ số dòng cần cập nhật>, <tham chiếu đối tượng dữ liệu mới cập nhật>);
```

- Xóa:

```
table.getItems().remove(<chỉ số dòng cần xóa>);
```

- + Chỉnh sửa dữ liệu trực tiếp trên TableView

- Thiết lập chế độ cho phép chỉnh sửa trực tiếp dữ liệu:

```
table.setEditable(boolean value);
```

- Cài đặt môi trường chỉnh sửa cho dữ liệu tại các ô trong cột

Môi trường chỉnh sửa dữ liệu trong một ô khi người dùng click đúp vào có thể là TextFieldTableCell (giống TextField), ComboBoxTableCell, CheckBoxTableCell... Đây là các lớp thuộc gói javafx.scene.control.cell.

Vd: lastNameCol.setCellFactory(TextFieldTableCell.forTableColumn());

- Cài đặt xử lý sự kiện khi bắt đầu, kết thúc hay bỏ qua chỉnh sửa ô:

```
Vd: lastNameCol.setOnEditStart(EventHandler value);
      lastNameCol.setOnEditCommit(EventHandler value);
      lastNameCol.setOnEditCancel(EventHandler value);
```

**Ví dụ:**



```
import java.util.ArrayList;
import java.util.List;
import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableColumn.CellEditEvent;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.CheckBoxTableCell;
import javafx.scene.control.cell.ComboBoxTableCell;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.control.cell.TextFieldTableCell;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.Priority;
import javafx.stage.Stage;

public class TableViewDemo extends Application {

    @SuppressWarnings("unchecked")
    @Override
    public void start(Stage stage) {

        TableView<Student> table = new TableView<Student>();
        // Cho phép chỉnh sửa dữ liệu trong bảng với
        // các cột thực hiện setCellFactory
        table.setEditable(true);
        // Cho phép thay đổi kích thước bảng +
        // Kết hợp setHgrow, setVgrow trong layout
        table.setColumnResizePolicy(
            TableView.CONSTRAINED_RESIZE_POLICY);
        TableColumn<Student, String> codeCol =
            new TableColumn<Student, String>("Msv");
        TableColumn<Student, String> fullNameCol =
            new TableColumn<Student, String>("Họ tên");
        TableColumn<Student, String> lastNameCol =
            new TableColumn<Student, String>("Họ đệm");
        TableColumn<Student, String> firstNameCol =
            new TableColumn<Student, String>("Tên");
        TableColumn<Student, String> genderCol =
            new TableColumn<Student, String>("Giới tính");
        TableColumn<Student, Boolean> activeCol =
            new TableColumn<Student, Boolean>("Kích hoạt");

        // Thêm 2 cột Họ đệm, Tên vào cột Họ tên
        fullNameCol.getColumns().addAll(lastNameCol, firstNameCol);
        table.getColumns().addAll(codeCol, fullNameCol,
                                   genderCol, activeCol);
```

```
// Bắt cặp 3 cột với 3 thuộc tính đã có
// dữ liệu trong đối tượng sinh viên
codeCol.setCellValueFactory(new PropertyValueFactory<Student,
                                String>("code"));

lastNameCol.setCellValueFactory(
    new PropertyValueFactory<Student, String>("lastName"));
firstNameCol.setCellValueFactory(
    new PropertyValueFactory<Student, String>("firstName"));

// Cho phép chỉnh sửa dữ liệu trường Họ đệm qua ô TextField
lastNameCol.setCellFactory(
    TextFieldTableCell.forTableColumn());

// Xử lý sự kiện kết thúc việc chỉnh sửa ô trong cột Họ đệm
lastNameCol.setOnEditCommit(
    new EventHandler<TableColumn.CellEditEvent<Student, String>>()
    {
        @Override
        public void handle(CellEditEvent<Student, String> event)
        {
            System.out.println("Thay đổi họ đệm: " +
                event.getOldValue() + " -> " +
                event.getNewValue());
            // Cập nhật vào đối tượng sinh viên
            // tương ứng trong danh sách
            String newLastName = event.getNewValue();
            int row = event.getTablePosition().getRow();
            Student std =
                event.getTableView().getItems().get(row);
            std.setLastName(newLastName);
        }
    });

// Cho phép chỉnh sửa dữ liệu trường "Kích hoạt" qua ô checkbox
activeCol.setCellFactory(
    CheckBoxTableCell.forTableColumn(activeCol));

// Cho phép chỉnh sửa dữ liệu trường "Giới tính" qua ô ComboBox
genderCol.setCellFactory(
    ComboBoxTableCell.forTableColumn("Nam", "Nữ"));

List<Student> studentList = getStudentList();
ObservableList<Student> obsList =
    FXCollections.observableArrayList(studentList);
table.setItems(obsList);

GridPane root = new GridPane();
root.getChildren().add(table);
```

```
GridPane.setHgrow(table, Priority.ALWAYS);
GridPane.setVgrow(table, Priority.ALWAYS);

stage.setTitle("Minh họa JavaFX TableView");

Scene scene = new Scene(root, 450, 300);
stage.setScene(scene);
stage.show();
}

private List<Student> getStudentList() {
    List<Student> studentList = new ArrayList<Student>();
    studentList.add(new Student("639313", "Hoàng", "Nguyễn Văn"));
    studentList.add(new Student("638811", "Ngọc", "Vũ Văn"));
    studentList.add(new Student("620835", "Hoa", "Nguyễn Mai"));
    studentList.add(new Student("622649", "Tuân", "Đỗ Đức"));
    return studentList;
}

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

public class Student {

    private String code;
    private String firstName;
    private String lastName;
    private String gender;
    private boolean active;

    public Student() {}

    public Student(String code, String firstName, String lastName) {
        this.code = code;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
```

```
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

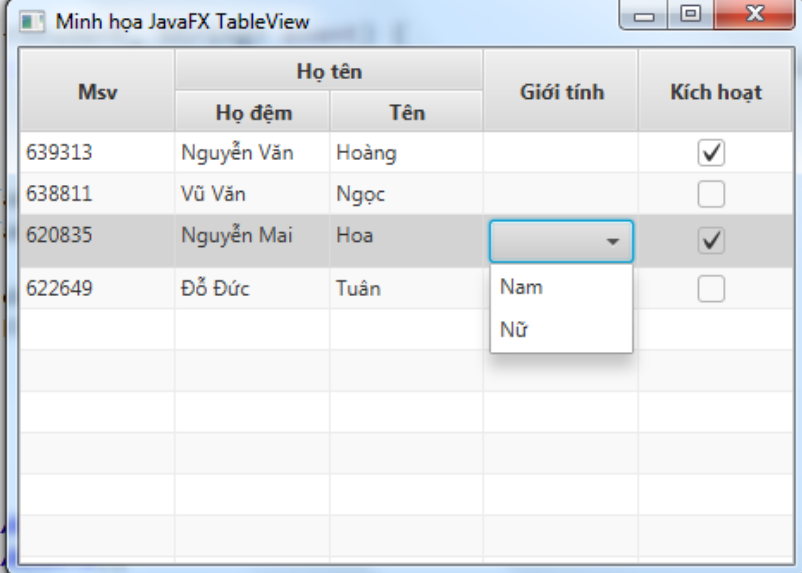
    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public boolean isActive() {
        return active;
    }

    public void setActive(boolean active) {
        this.active = active;
    }
}
```

Kết quả chạy chương trình:



| Msv    | Họ tên     |       | Giới tính   | Kích hoạt                           |
|--------|------------|-------|---|-------------------------------------|
|        | Họ đệm     | Tên   |   |                                     |
| 639313 | Nguyễn Văn | Hoàng |   | <input checked="" type="checkbox"/> |
| 638811 | Vũ Văn     | Ngọc  |   | <input type="checkbox"/>            |
| 620835 | Nguyễn Mai | Hoa   | <div><div></div><div>Nam</div><div>Nữ</div></div> | <input checked="" type="checkbox"/> |
| 622649 | Đỗ Đức     | Tuân  |   | <input type="checkbox"/>            |
|        |            |       |   |                                     |
|        |            |       |   |                                     |
|        |            |       |   |                                     |
|        |            |       |   |                                     |