CS 213 – Software Methodology

Spring 2023

Sesh Venugopal

Feb 8

ListView, Dialogs

Listening to List Item Selection Event

view.ListController

```java
package listview;
...
import javafx.stage.Stage;

public class ListController {
    ...

    public void start(Stage mainStage) {
        ...
        // select the first item
        listView.getSelectionModel().select(0);

        // set listener for the items
        listView
           .getSelectionModel()
           .selectedIndexProperty()
           .addListener(
               (obs, oldVal, newVal) ->
                    showItem(mainStage));
    }
}
```

lambda expression for the
changed method of the
functional interface
javafx.beans.value.ChangeListener

Handling List Item Selection Event

```java
package listview;
...
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;

public class ListController {
    ...

    private void showItem(Stage mainStage) {
        Alert alert =
            new Alert(AlertType.INFORMATION);
        alert.initOwner(mainStage);
        alert.setTitle("List Item");
        alert.setHeaderText(
            "Selected list item properties");

        String content = "Index: " +
            listView.getSelectionModel()
                    .getSelectedIndex() +
            "\nValue: " +
            listView.getSelectionModel()
                    .getSelectedItem();

        alert.setContentText(content);
        alert.showAndWait();
    }
}
```
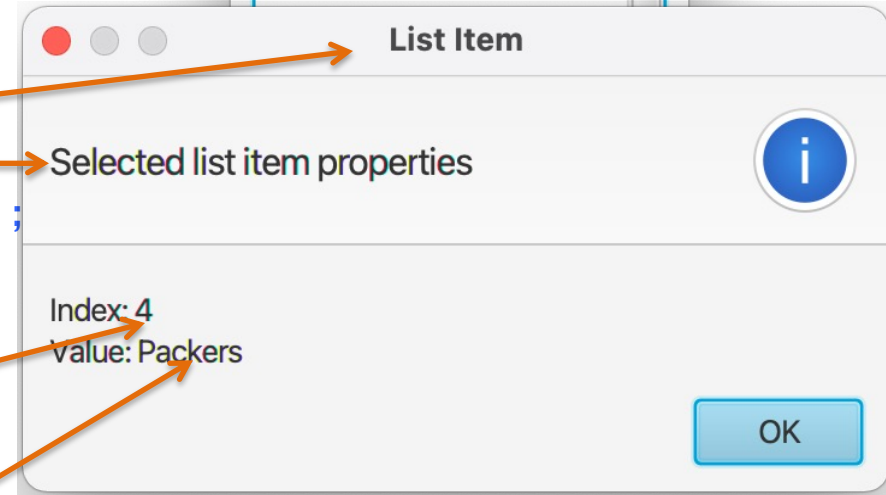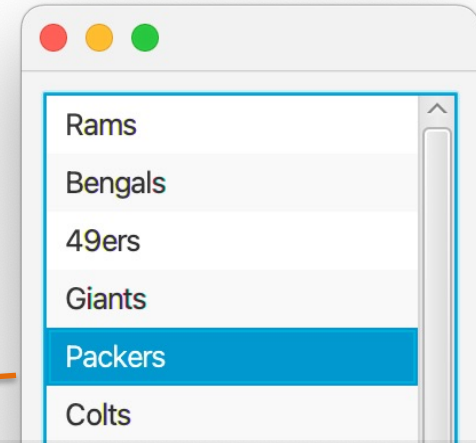
Rams
Bengals
49ers
Giants
Packers
Colts

List Item

Selected list item properties

Index: 4
Value: Packers

OK

The dialog will block execution until user responds (AndWait()). Also, it will not allow interaction with owner window: this makes the dialog "modal" (default behavior)

3

Enhancement: Change Item

```java
package listview;
...
import java.util.Optional; import javafx.scene.control.TextInputDialog;

public class ListController {
    ...
    public void start(Stage mainStage) {
        ...
        listView.
            ...
            .addListener((obs, oldVal, newVal) ->
                    showItemInputDialog(mainStage));
    }

    private void showItemInputDialog(Stage mainStage) {
        String item = listView.getSelectionModel().getSelectedItem();
        int index = listView.getSelectionModel().getSelectedIndex();

        TextInputDialog dialog = new TextInputDialog(item);
        dialog.initOwner(mainStage); dialog.setTitle("List Item");
        dialog.setHeaderText("Selected Item (Index: " + index + ")");
        dialog.setContentText("Enter name: ");

        Optional<String> result = dialog.showAndWait();
        if (result.isPresent()) { obsList.set(index, result.get()); }
    }
}
```

ObservableList => ListView Auto Update

`showItemInputDialog`

`TextInputDialog`
`showAndWait()`

change name

`Optional<String>`
`result`

`obsList.set(index,`
`result.get())`

Updating `ObservableList` automatically
refreshes the `ListView`