Application Program Development

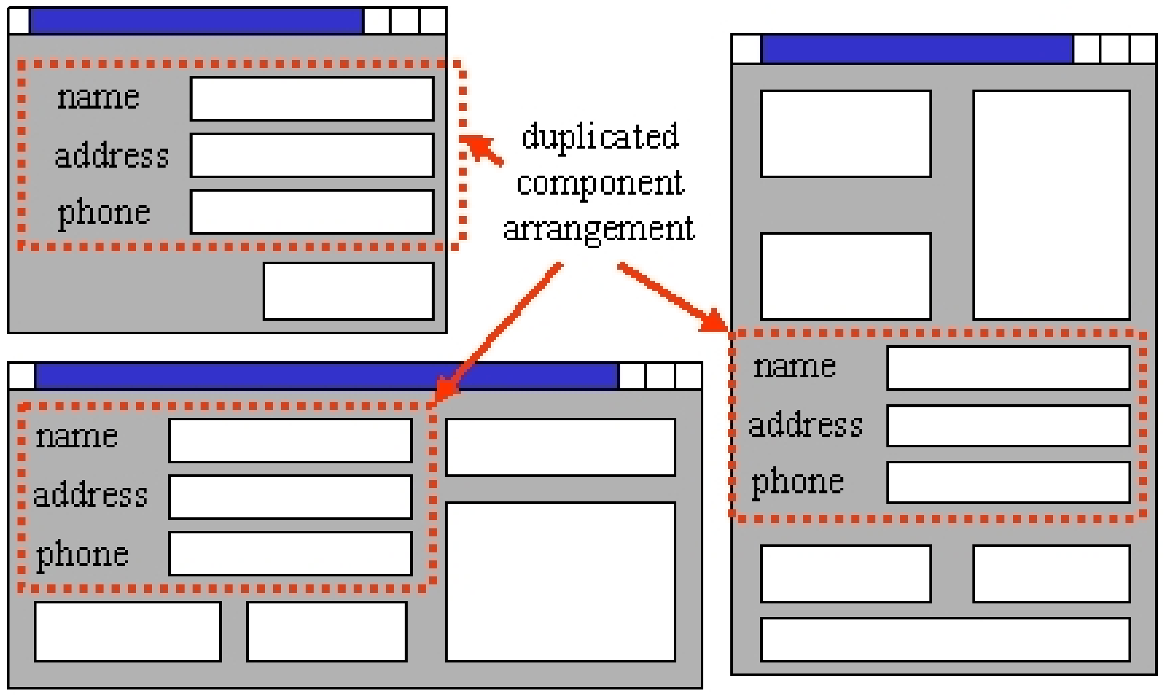
Segment : Grouping Components Together

Outcomes

- Understanding of Grouping components
- Designing simple applications with different groupings

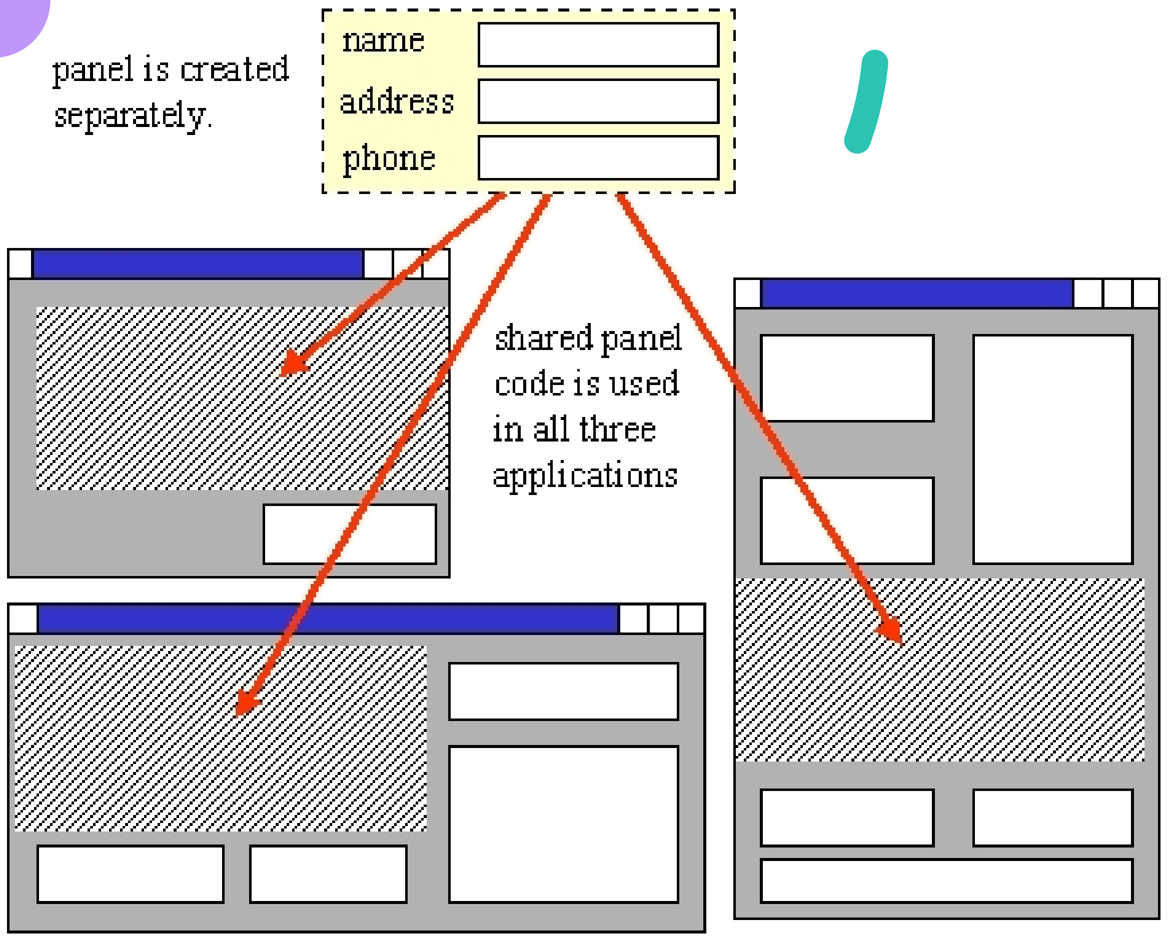
Organizing Components

- It is a very good idea to keep your window components organized. It is often the case that an arrangement of components may be similar (or duplicated) within different windows.
- For example, an application may require a name, address and phone number to be entered at different times in different windows →
- It is a good idea to share component layouts among the similar windows within an application so that the amount of code you write is reduced.
- To do this, we often lay out components onto a **Pane** and then place the **Pane** on our window. We can place the pane on many different windows with one line of code ... this can greatly reduce the amount of GUI code that you have to write.



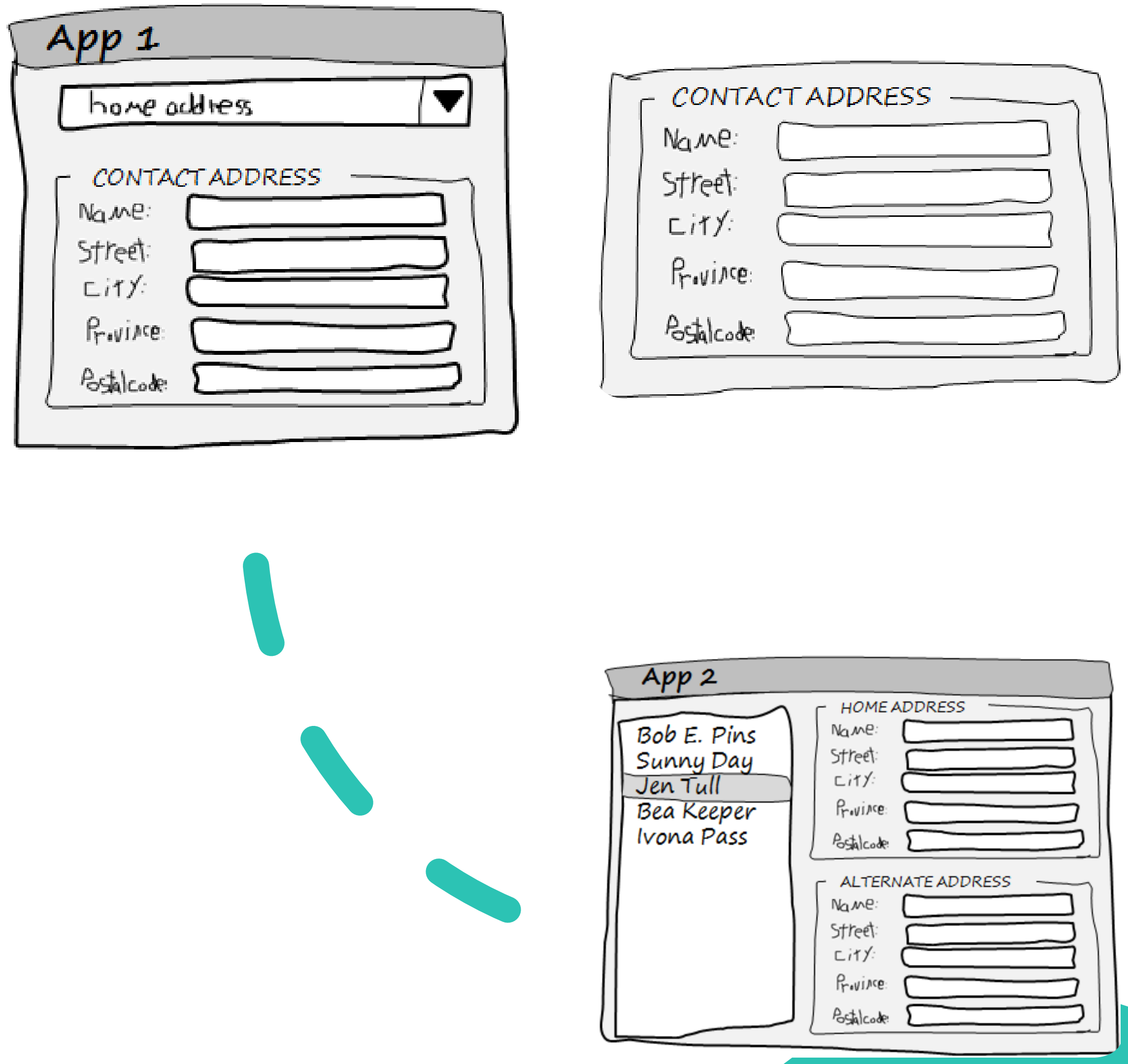
Organizing Components

- You will often want to create separate **Pane** objects to contain groups of components so that you can move them around (as a group) to different parts of a window or even be shared between different windows. The code to do this simply involves creating our **Pane** with its appropriate component arrangement and then adding the **Pane** to the window.
- **Pane** objects are added to a window just like any other objects. So, we can have a **Pane** within another **Pane**.
- The following example shows how this can be done. The code will show you how to make a **Pane** with a nice titled border.


panel is created separately.
name
address
phone
shared panel code is used in all three applications

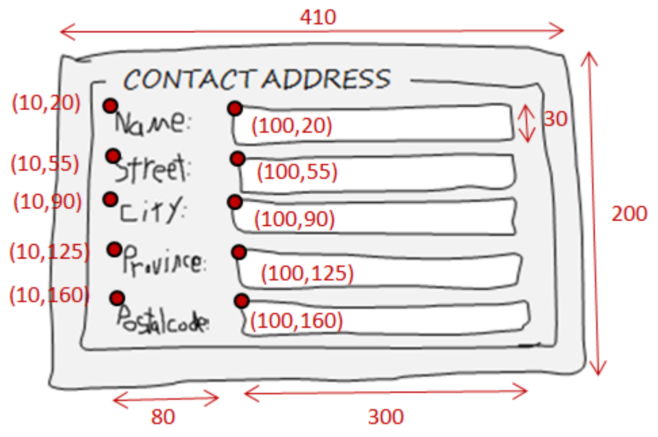
Example (Contact Address)

- Consider this example in which a **Pane** is used in more than one window. We will create a simple pane called **AddressPane** that contains 5 labels and 5 text fields for allowing the user to enter a name and address as shown here →
- The pane contains 5 **TextField** objects that allow the user to fill-in an address. It also has 5 **Label** objects (which are simply pieces of text) to indicate the kind of data expected for each text field. Lastly, there is a nice border around the **Pane** which has the title CONTACT ADDRESS. This pane will not be its own window. Instead, it will be a pane inside of two other windows that look as shown below. Notice that each application has the same kind of **AddressPane**, except that the border's title varies.



- Here are the dimensions →
- To begin the code, we will want to define an **AddressPane** class. It will be a special kind of **Pane**, and so it should be a subclass of **Pane**.

```
import javafx.geometry.Pos;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.Pane;
public class AddressPane extends Pane {
          public AddressPane(String title) {
          Pane innerPane = new Pane();
          innerPane.setStyle("-fx-background-color: white; " +
                            "-fx-border-color: gray; "
                            + "-fx-padding: 4 4;");// margin spacing at bottom right
          // Create the labels and textfields
          Label label1 = new Label("Name:");
          label1.relocate(10, 20);
          label1.setPrefSize(80, 30);
          Label label2 = new Label("Street:");
          label2.relocate(10, 55);
          label2.setPrefSize(80, 30);
          Label label3 = new Label("City:");
          label3.relocate(10, 90);
          label3.setPrefSize(80, 30);
          Label label4 = new Label("Province:");
          label4.relocate(10, 125);
          label4.setPrefSize(80, 30);
          Label label5 = new Label("Postal Code:");
          label5.relocate(10, 160);
          label5.setPrefSize(80, 30);
```



```
    TextField nameField = new TextField();
    nameField.relocate(100, 20);
    nameField.setPrefSize(300, 30);
    TextField streetField = new TextField();
    streetField.relocate(100, 55);
    streetField.setPrefSize(300, 30);
    TextField cityField = new TextField();
    cityField.relocate(100, 90);
    cityField.setPrefSize(300, 30);
    TextField provinceField = new TextField();
    provinceField.relocate(100, 125);
    provinceField.setPrefSize(300, 30);
    TextField postalField = new TextField();
    postalField.relocate(100, 160);
    postalField.setPrefSize(300, 30); // Add all labels and textfields to the pane
    innerPane.getChildren().addAll(label1, label2, label3, label4, label5,
                   nameField, streetField, cityField, provinceField, postalField);
    // Make a title for border and add it as well as inner pane to main pane
    Label titleLabel = new Label(); // Title to be placed onto border
    titleLabel.setText(title); // Incoming constructor parameter
    titleLabel.setStyle("-fx-background-color: white; \n" +
                      "-fx-translate-y: -8; \n" +
                      "-fx-translate-x: 10;");
    getChildren().addAll(innerPane, titleLabel);
    }
}
```

Example (Contact Address)

- The code starts by creating an **innerPane** that will hold the 5 labels and 5 text fields. It will have a white background and gray border. Each of the 5 labels and text fields are created and given a location and size. They are all added to the Pane in one method call after their creation.
- At the bottom of the method, another Label is created to be the label on the **Pane**'s border. This *border title* will be whatever was passed in to the **AddressPane** constructor. Finally, the **innerPane** and **titleLabel** are added to the **AddressPane**. The order of adding is important because we want the **titleLabel** to be drawn on top of the **innerPane**.
- You may have noticed that we did not set the size of the Pane. This Pane's size is automatically set according to the width and height of its components.
- The **AddressPane** class itself is not a runable application (i.e., there is no **main** method).
- The **AddressPane** will now be "treated" as a single component and will be placed on the main window by specifying its location. We do not need to specify the size of the **AddressPane**, since this is fixed.
- So we will now make our **App1** application to test it out. Next slide to check the dimensions for the application →

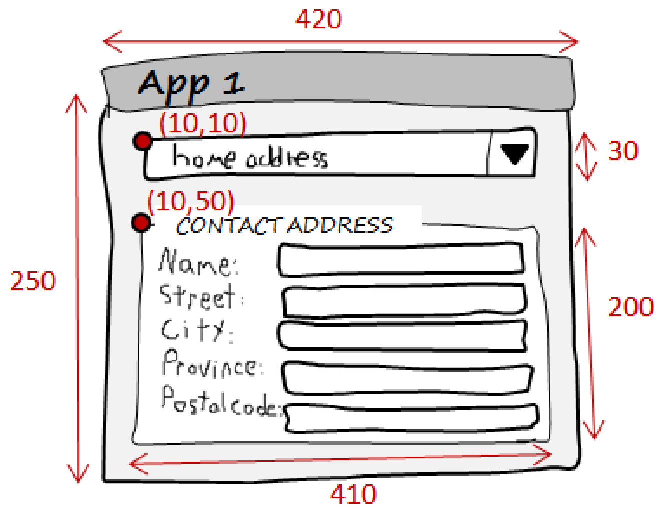
- The topmost component here is called a **ComboBox** and it represents what is known as a drop-down list. It is similar to a **ListView**, except that only the selected item is shown, and the remaining items can be shown by pressing the black arrow.
- A **ComboBox** can be created by specifying an array of objects that are to appear in the list and passing this in as a parameter to the constructor as follows:

```
ObservableList<String> options =
        FXCollections.observableArrayList( "Home Address",
                                "Work Address", "Alternate Address");

        ComboBox addressBox = new ComboBox(options);

        addressBox.setPromptText("Choose address type");
```

- Notice as well that you can set the *prompt text*. This is the text that will be displayed in the **ComboBox** when the window first opens. It is usually used to provide instructions to the user of the program to make a selection. Alternatively, you could set the initial value to anything you want as follows:

```
        addressBox.setValue("Work Address");
```

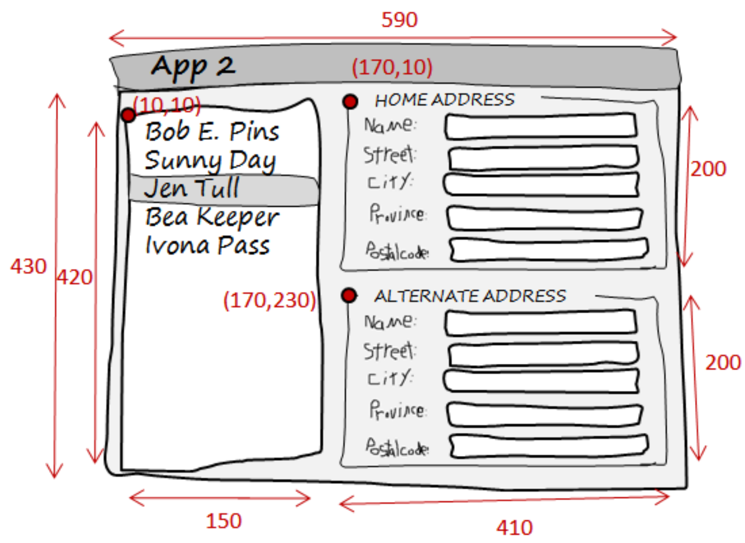
- In this case, the panel will show "Work Address" as its value upon startup.
- Here is the code to create the application.
- Take note of how the **AddressPane** is now used as if it were a single, simple component



```
import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.scene.Scene;
import javafx.scene.control.ComboBox;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class OneApp extends Application {
        public void start(Stage primaryStage) {
                Pane aPane = new Pane(); // Add the drop-down list
                ObservableList<String> options = FXCollections.observableArrayList(
                                "Home Address", "Work Address", "Alternate Address");
                ComboBox addressBox = new ComboBox(options);
        //addressBox.setPromptText("Choose address type");
                addressBox.setValue("Work Address");
                addressBox.relocate(10,10);
                addressBox.setPrefSize(410,30);
                aPane.getChildren().add(addressBox);
                // Now add an AddressPane
                AddressPane myPanel = new AddressPane("CONTACT ADDRESS");
                myPanel.relocate(10,50);
                aPane.getChildren().add(myPanel);
                primaryStage.setTitle("App 1"); // Set title of window
                primaryStage.setResizable(false); // Make it non-resizable
                primaryStage.setScene(new Scene(aPane, 420,250));
                // Set size of window
                primaryStage.show();
                }
        public static void main(String[] args) { launch(args); }
}
```

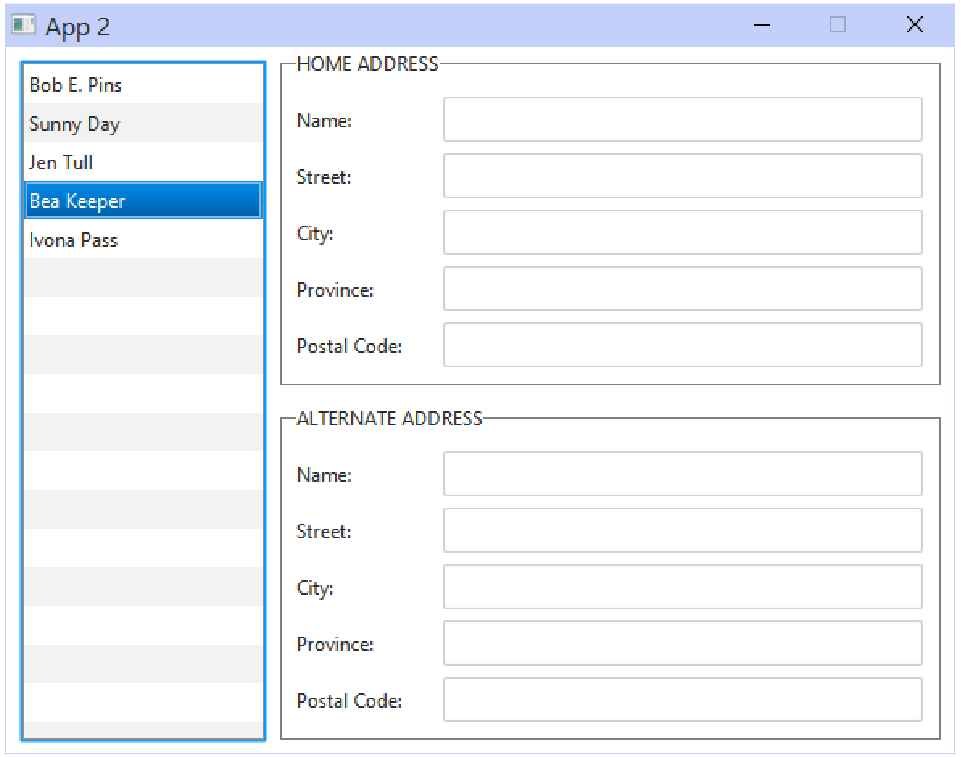
- Now let us consider a second application that makes use of the same **AddressPane** without altering the code in that class. Here are the dimensions for the 2nd application:
- Here is the code. Again, notice how the **AddressPane** is used twice in the same window with a different title:

```
import javafx.application.Application;
import javafx.collections.FXCollections;
import javafx.scene.Scene;
import javafx.scene.control.ListView;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class TwoApp extends Application {
        public void start(Stage primaryStage) {
                Pane aPane = new Pane(); // Add the list
        ListView<String> namesList = new ListView<String>();
        String[] fruits = {"Bob E. Pins", "Sunny Day", "Jen Tull",
                                          "Bea Keeper", "Ivona Pass"};
        namesList.setItems(FXCollections.observableArrayList(fruits));
        namesList.relocate(10, 10); namesList.setPrefSize(150, 420);
        aPane.getChildren().add(namesList); // Now add an AddressPane
        AddressPane myPanel1 = new AddressPane("HOME ADDRESS");
        myPanel1.relocate(170,10);
        aPane.getChildren().add(myPanel1); // Now add another AddressPane
        AddressPane myPanel2 = new AddressPane("ALTERNATE ADDRESS");
        myPanel2.relocate(170,230);
        aPane.getChildren().add(myPanel2);
```



```
        primaryStage.setTitle("App 2"); // Set title of window
        primaryStage.setResizable(false); // Make it non-resizable
        primaryStage.setScene(new Scene(aPane, 578,428)); // Set size of window
        primaryStage.show();
        }
        public static void main(String[] args) { launch(args); }
}
```

- Notice that the size of the scene will need to be adjusted a bit since we are using a non-resizable window. In this case, the width was reduced by 12 and the height by 2.



App 2

Bob E. Pins
Sunny Day
Jen Tull
Bea Keeper
Ivona Pass

HOME ADDRESS
Name:
Street:
City:
Province:
Postal Code:

ALTERNATE ADDRESS
Name:
Street:
City:
Province:
Postal Code: