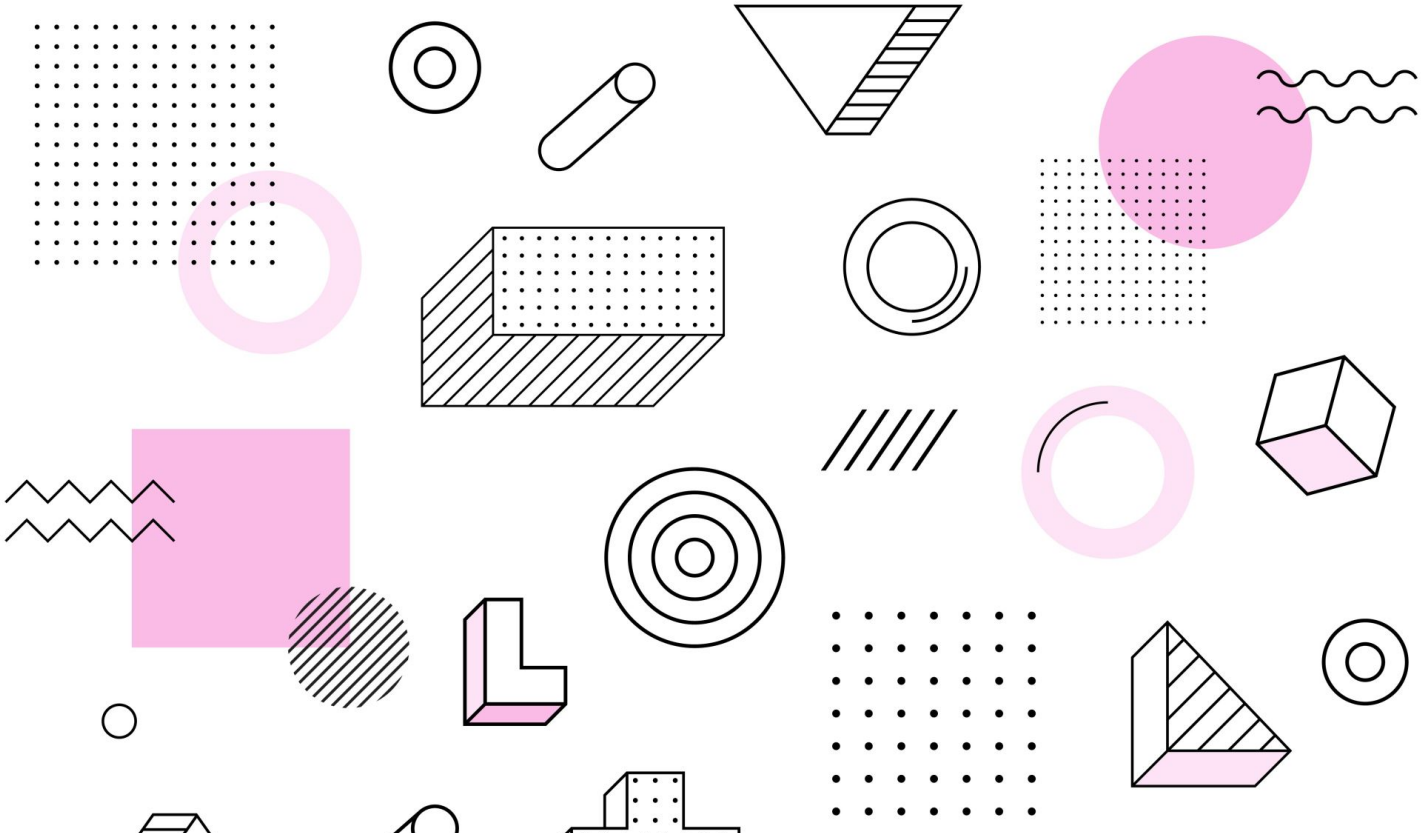


Graphicical User Interface V

Various Widgets

Week 6 Presentation 3

Widgets: Buttons

Common buttons and expected behavior…

Standards are important in making a GUI usable, and as a result your application, consistence also creates a professional appearance.

OK

- Changes applied, close window

Cancel

- No changes, close window

Close

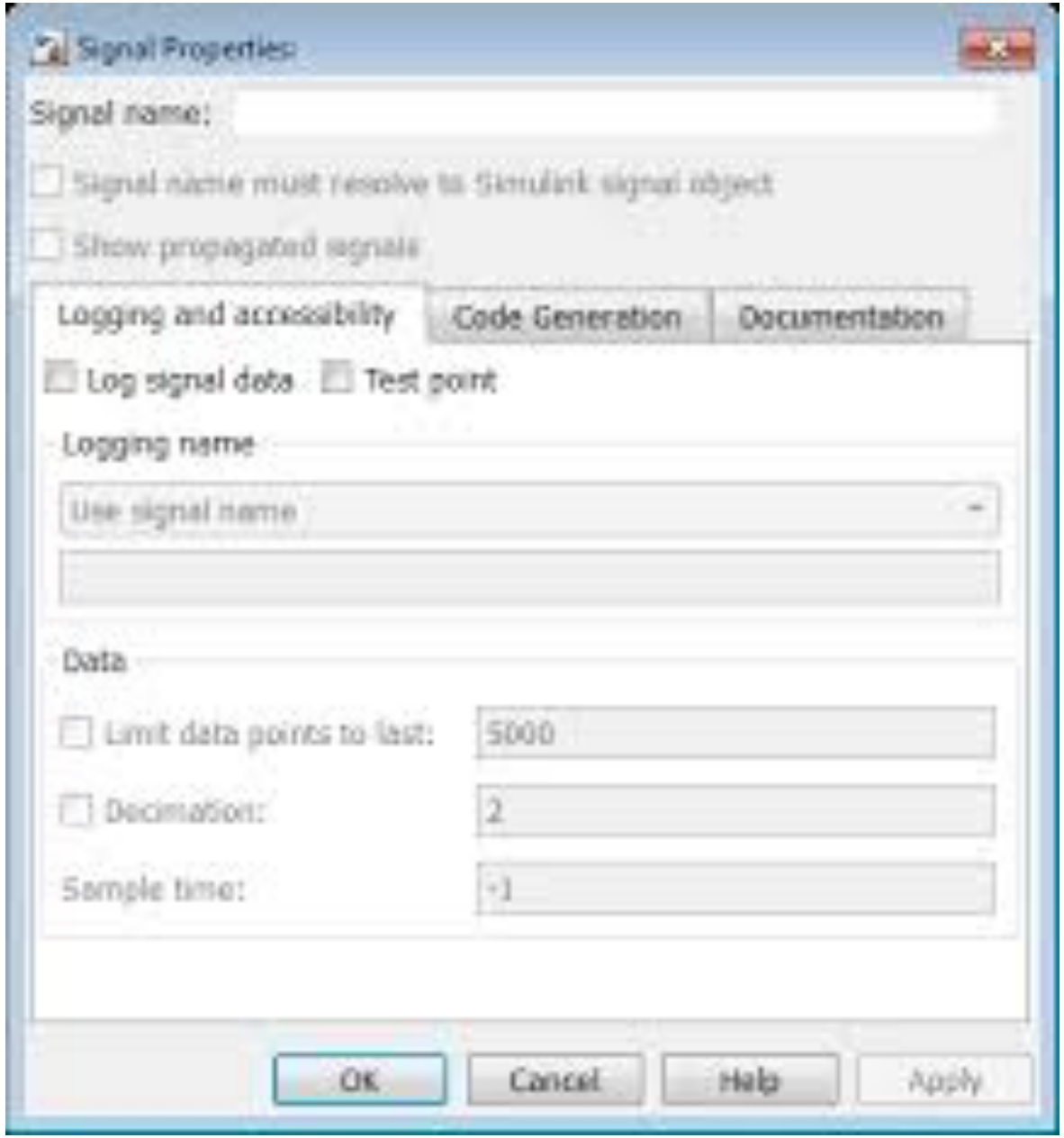
- Can’t cancel, close window

Reset

- Set default, keep window open

OK

- Sometimes changes applied, keep window open


Signal Properties
Signal name:
Signal name must resolve to Simulink signal object
Show propagated signals
Logging and accessibility
Code Generation
Documentation
Log signal data
Test point
Logging name
Use signal name
Data
Limit data points to last:
5000
Decimation:
2
Sample time:
-1
OK
Cancel
Help
Apply

Widgets: Buttons

Keep all buttons the same size

... or have a "short button" and a "long" button size

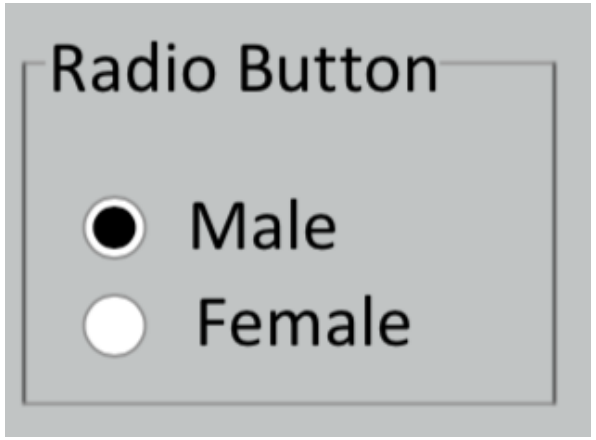
Group buttons

Isolate buttons from the rest (space)

Widgets: Radio Buttons

For several exclusive choices

Usually in a group

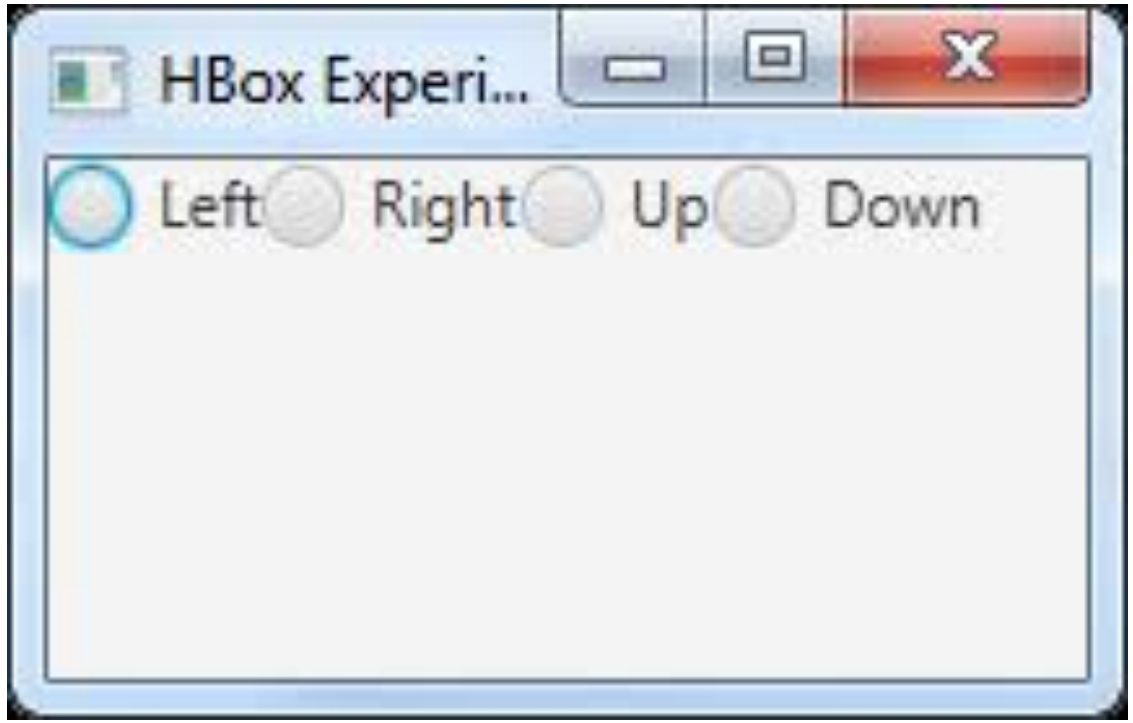


In a quiz a radio button will tell you only one answer is correct….

Toggle group object

Toggle groups can be used to make radio buttons represent a set of on off switches in which only one can be on

javafx.scene.control.ToggleGroup

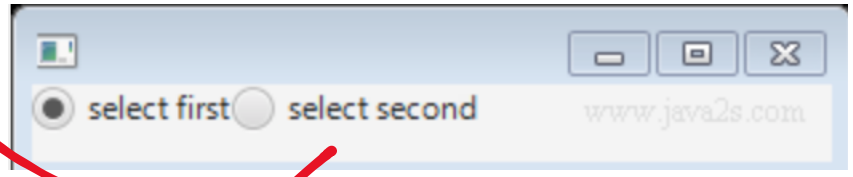


Toggle group object

Set of on off switches in which one can be on.

javafx.scene.control.ToggleGroup

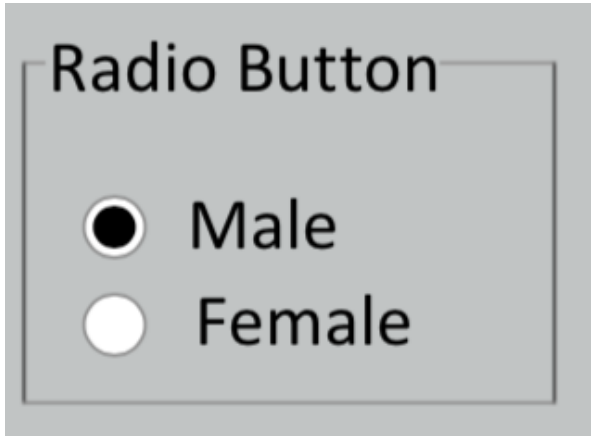
```
ToggleGroup radioGroup = new ToggleGroup();
radioButton1.setToggleGroup(radioGroup);
radioButton2.setToggleGroup(radioGroup);
radioButton3.setToggleGroup(radioGroup);
```

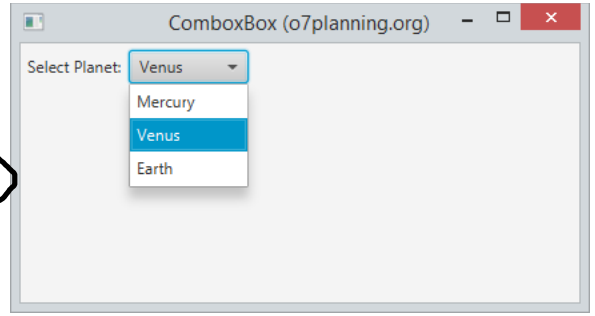



```java
public void start(Stage stage) {
  HBox root = new HBox();
  Scene scene = new Scene(root, 300, 150);
  stage.setScene(scene);
  stage.setTitle("");

  ToggleGroup group = new ToggleGroup();
  RadioButton button1 = new RadioButton("select first");
  button1.setToggleGroup(group);
  button1.setSelected(true);
  RadioButton button2 = new RadioButton("select second");
  button2.setToggleGroup(group);

  root.getChildren().add(button1);
  root.getChildren().add(button2);

  scene.setRoot(root);
  stage.show();
}
```

Radio Button Widgets

- One of several exclusive choices
- Usually in a group
- Use vertically
- Six options or less
- If more than six options use a ListBox
- Avoid Yes/No or On/Off

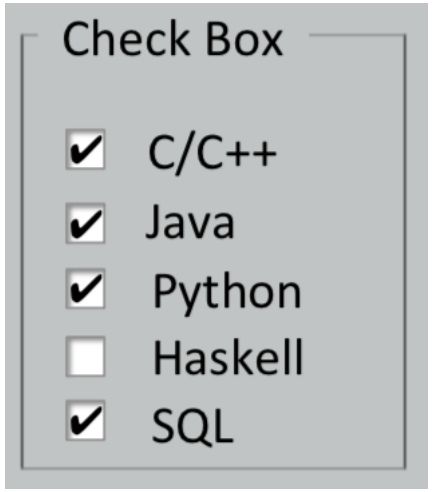




Widgets: CheckBox

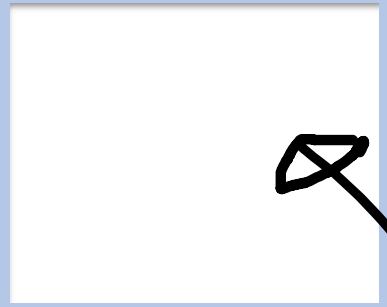
- More than several options allowed
- Toggling (Yes/No or On/Off)
- Use vertically
- Ten options or less
- Button for ”select all”

- Alternative is a multiple – select ListBox



Widgets for getting data from the user

One line – TextField
No echo – PasswordField
Prompt text (eg ”Your email”)

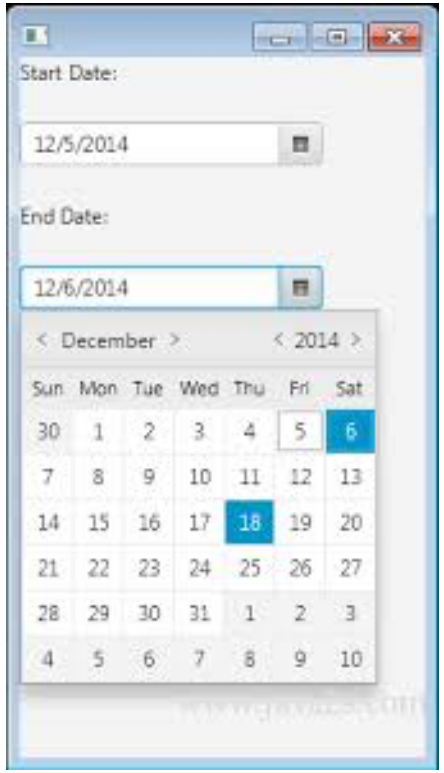


Multiple lines: TextArea

Special Widgets for Special Purposes

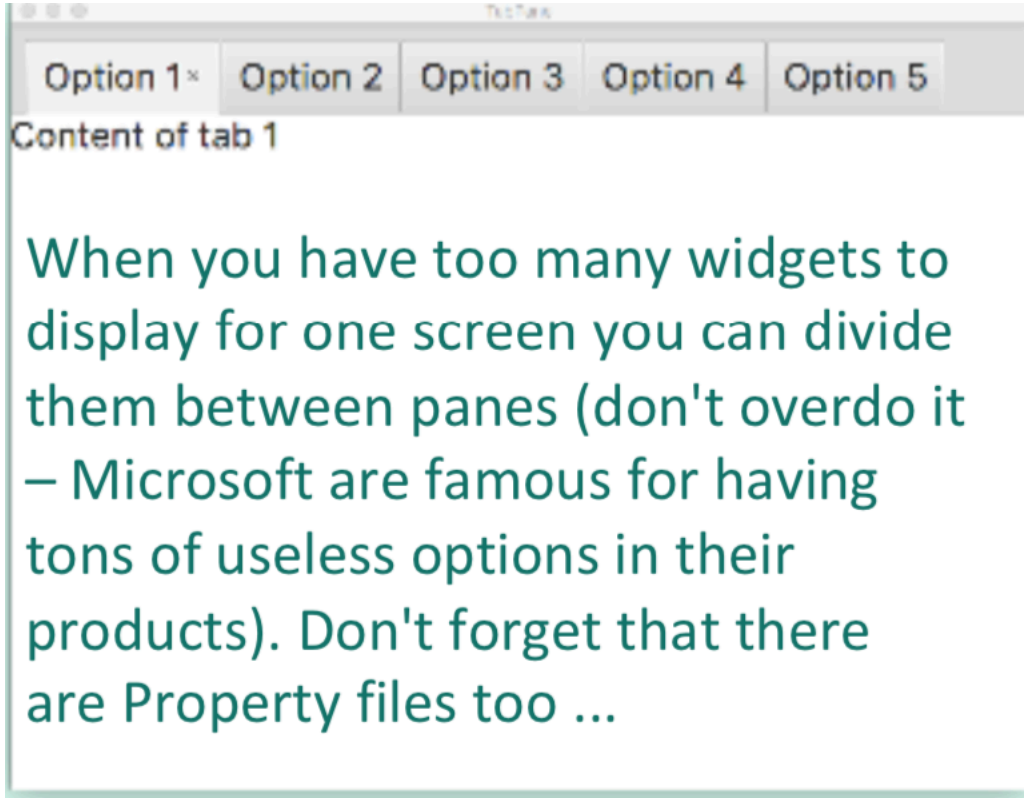
- ColorPicker

- DatePicker



- There are others too…

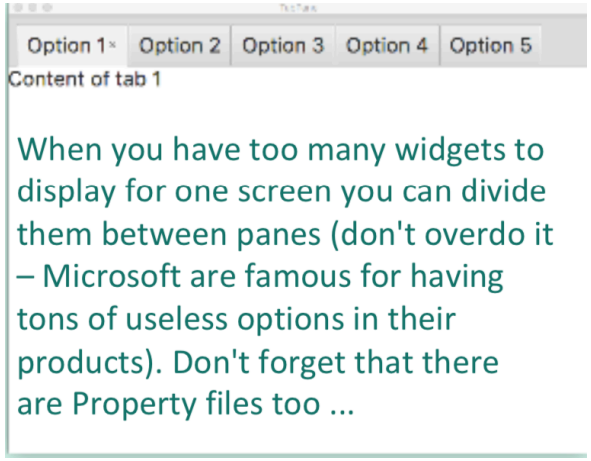
Widget TabPane



- TabPane: useful to avoid clutter

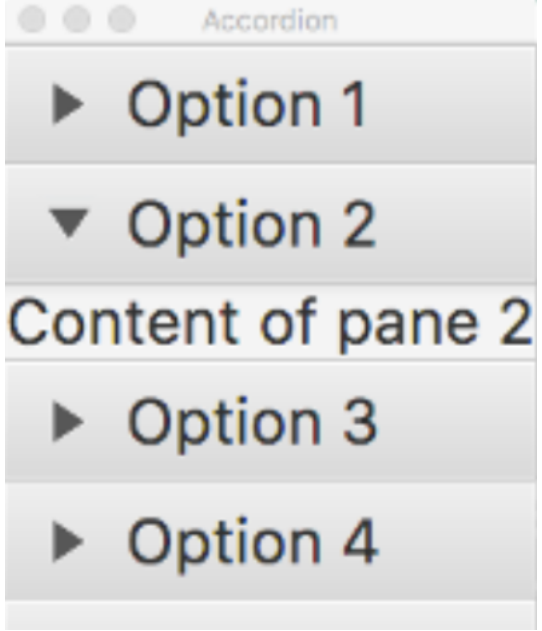
```
…
// Create a TabPane
TabPane pane = new TabPane();
pane.setPrefWidth(800);
pane.setPrefHeight(600);
root.getChildren().add(pane);
Tab tab;


// Create five tabs
for (int i = 1; i <= 5; i++) {
        tab = new Tab();
        tab.setText("Option " + i);
        tab.setContent(new Label("Content of tab " + i));
        pane.getTabs().add(tab);
        }
```

Container of tabs



Nodes can have any container or other widget

Widget Accordion and Titled Panes



- Titled panes are added to an accordion

Too Many Widgets? Can Use Accordion and Titled Panes

```
// Create an Accordion
Accordion accordion = new Accordion();
root.getChildren().add(accordion);
TitledPane pane;

// Create five titled panes
for (int i = 1; i <= 5; i++) {
        pane = new TitledPane();
        pane.setText("Option " + i);
        pane.setContent(new Label("Content of pane " + i));
        accordion.getPanes().add(pane);
        }
```

Padding and Spacing

Padding	Distance from the edge
Spacing	Distance between widgets

To make everything more readable, there should be “white” space. Two options, padding and spacing (which can change when you resize windows)

Padding and Spacing

.setPadding(Insets paddingValue)

```
import javafx.geometry.Insets;
Insets(double top, double right,
                double bottom, double left);

Insets(double sameValueEverywhere);
```

in pixels

Padding and Spacing

- Same distance between all the widgets in a container

.setSpacing(double spacingValue)

- Used to compute initial size
- When the window is first displayed, it may have a size you set, or the size may be computed.
- Of course, a lot of things will change in spacing if you broaden the window for instance.

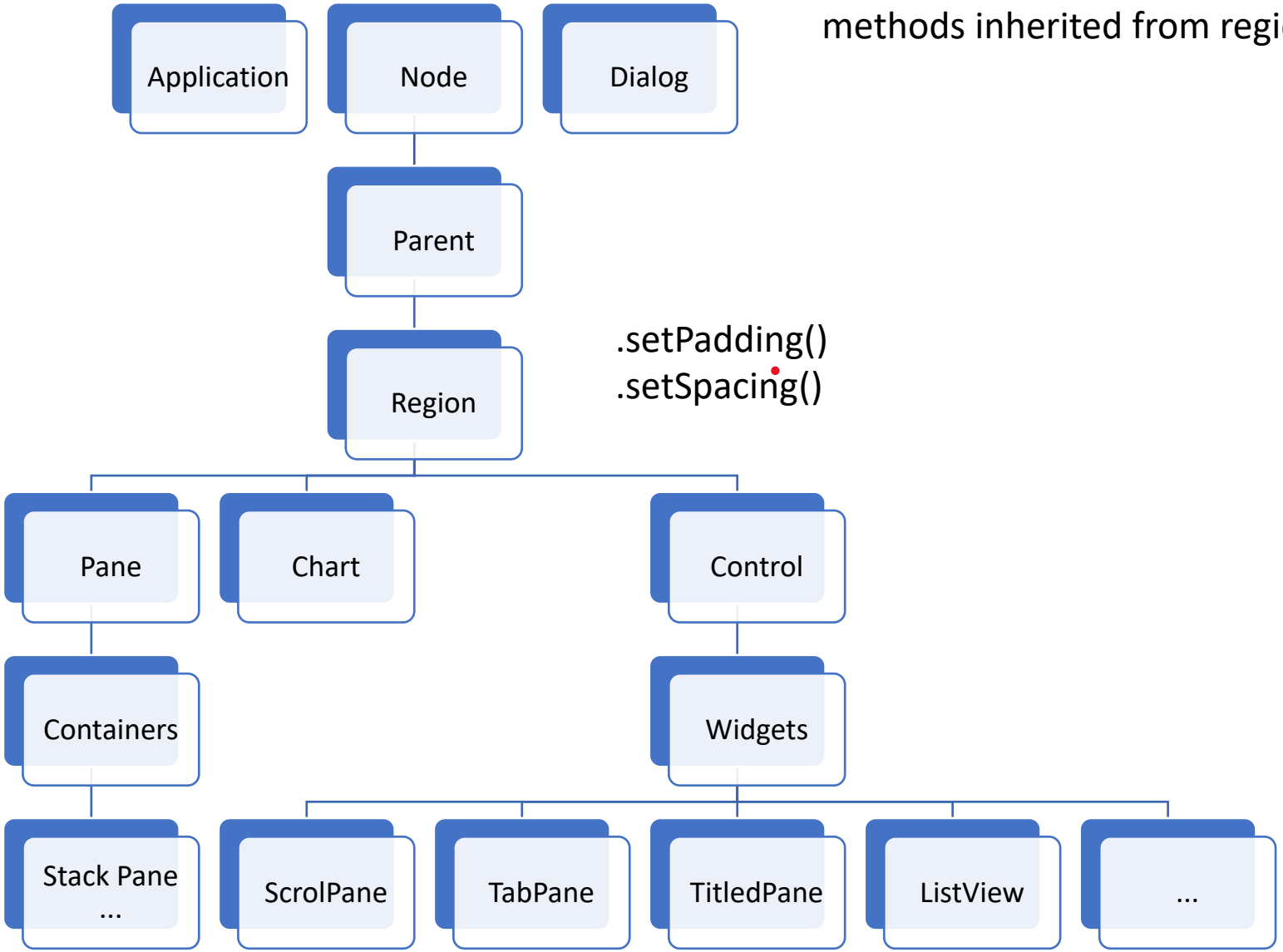
Padding and Spacing

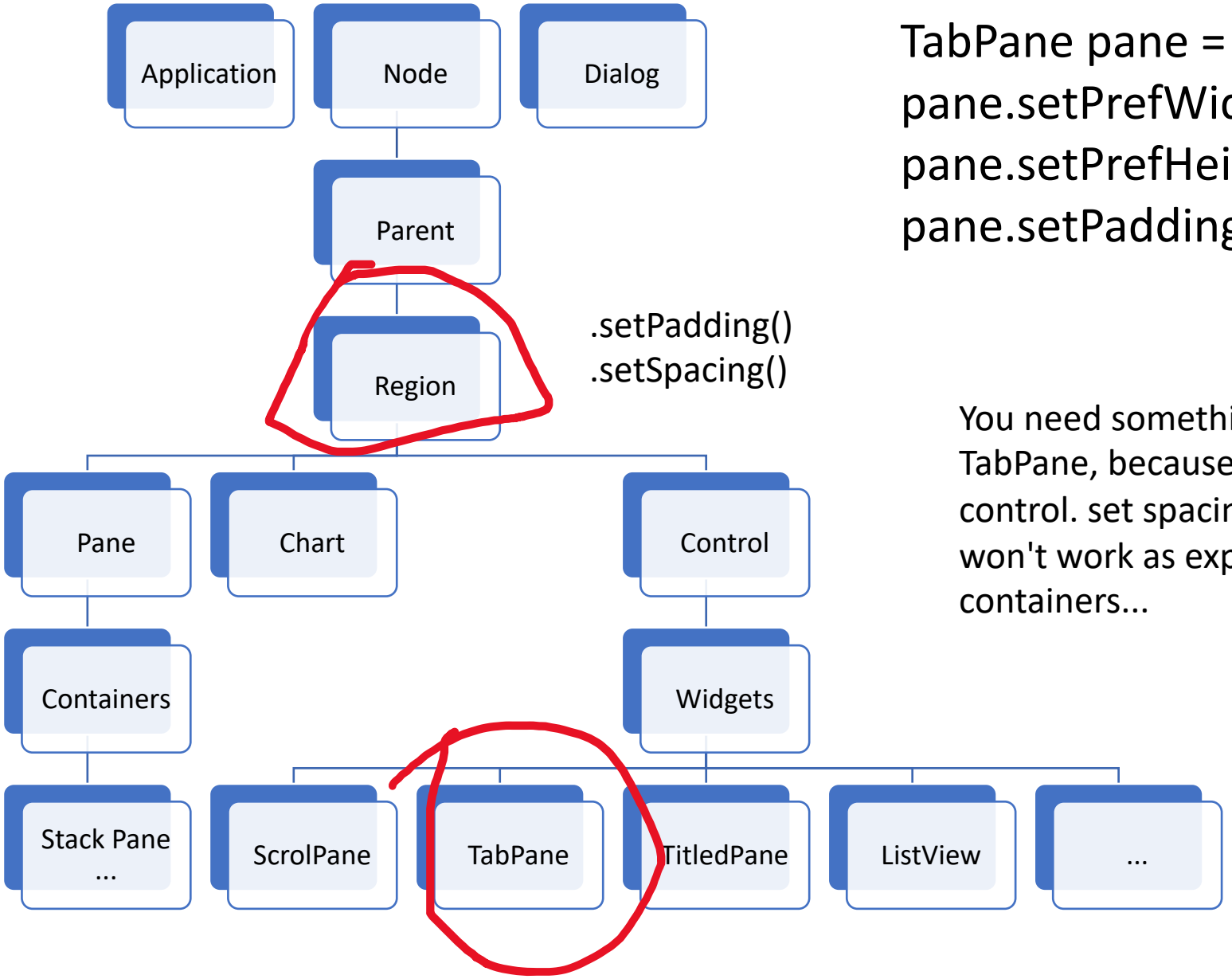
- Some containers (BorderPane, GridPane, HBox, VBox, StackPane, TilePane) implement a static method:

```
.setMargin(Node child, Insets marginValue)
```

(allows for setting spacing at the level of the individual widgets)

Be careful: padding and spacing are set with methods inherited from region






```
TabPane pane = new TabPane();
pane.setPrefWidth(800);
pane.setPrefHeight(600);
pane.setPadding(new Insets(20));
```

You need something special with a TabPane, because it's a composite control. set spacing or padding won't work as expected for containers...

Instead you should add a container (region) to the tab, eg Vbox:

```
Tab tab;
VBox tabBox;
// Create five tabs
for (int i = 1; i <= 5; i++) {
        tab = new Tab(); tab.setText("Option " + i);
        tabBox = new VBox();
        tabBox.setPadding(new Insets(20));
        tabBox.getChildren().add(new Label("Content of tab " + i));
        tab.setContent(tabBox);
        pane.getTabs().add(tab);
        }
```