ABSTRACT MODEL WIDGETS

CMPT 381

Overview

- Review of interaction model for widgets
- Example abstract model widgets
  - List View
  - Tree View
  - Table View

Basic widget interaction model

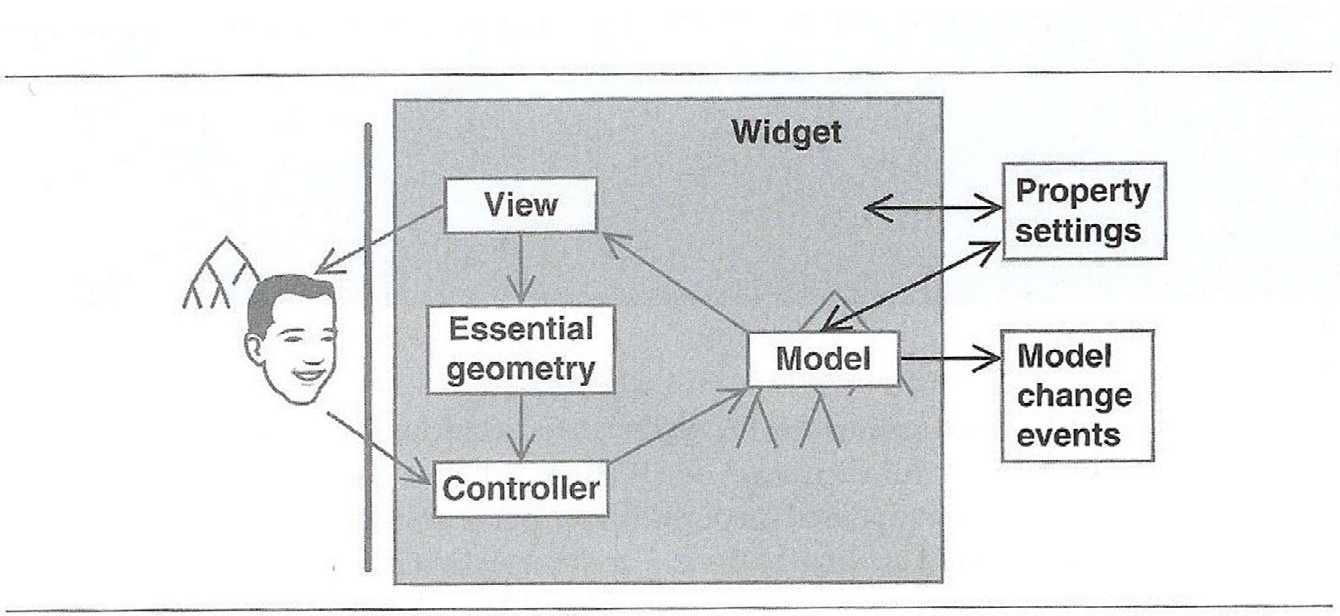


Figure 4.1 – Widget architecture

Basic widget interaction model

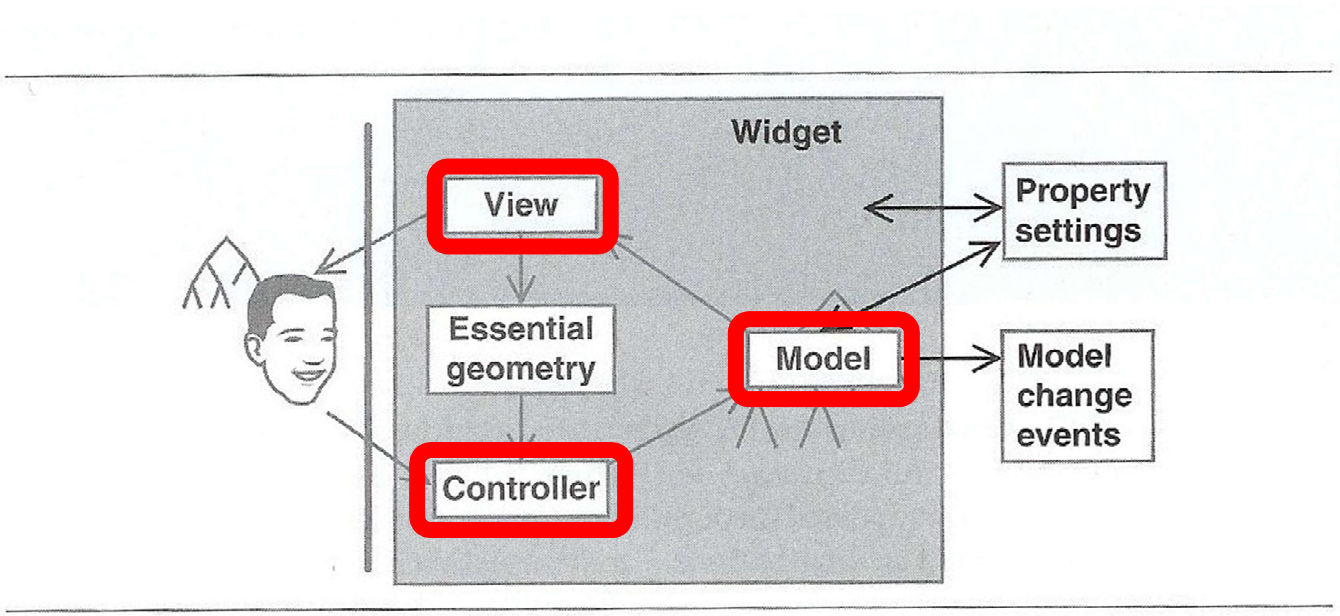
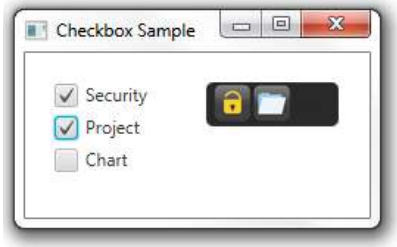
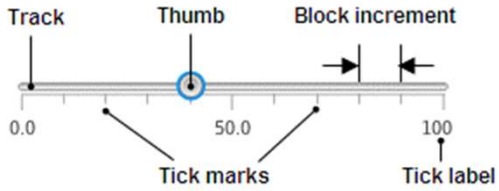


Figure 4.1 – Widget architecture

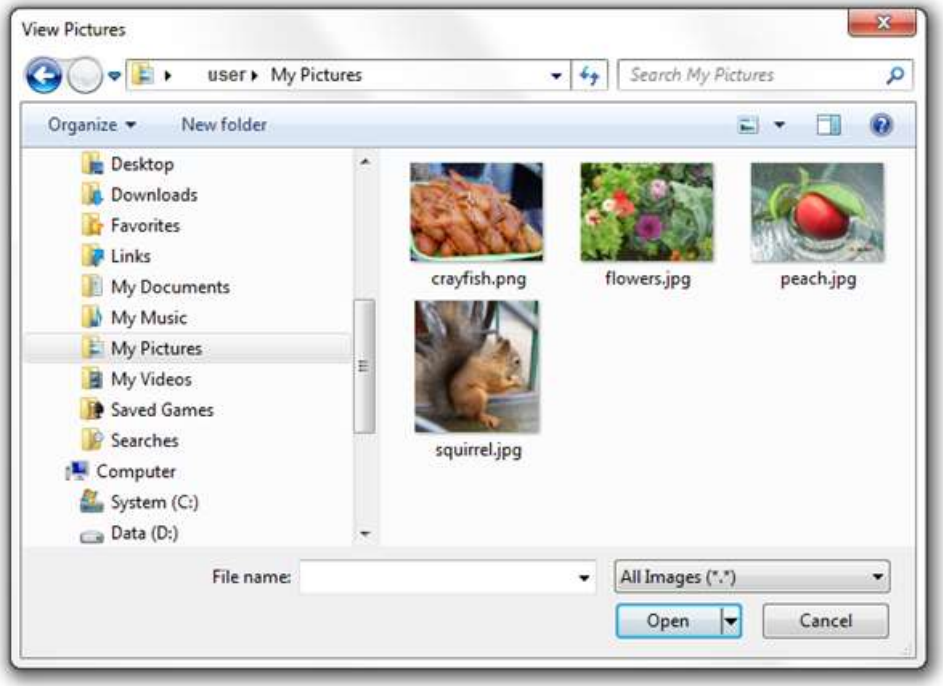
Simple Widgets

- Virtual input devices
  - Buttons
- Widgets with minimal model
  - Model integrated with widget
  - e.g., text string (label)
- State models
  - e.g., check boxes / radio buttons
- Number widgets
  - e.g., sliders or scrollbars

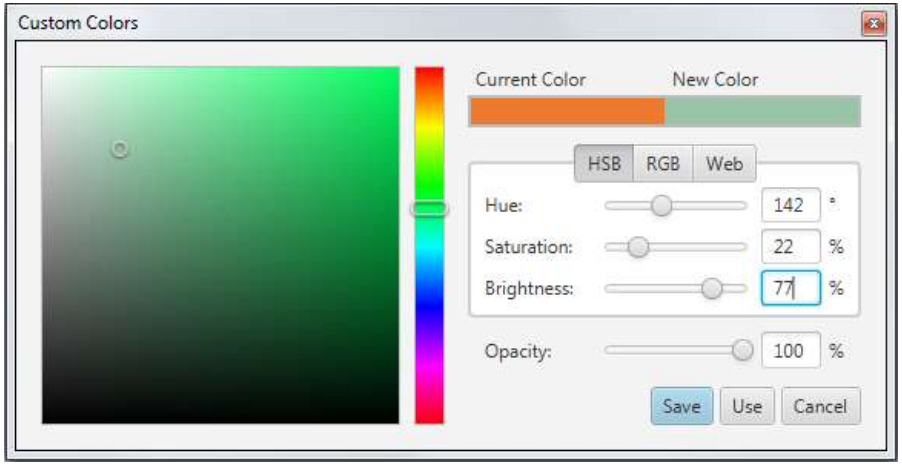




Large (but still simple) widgets



File dialog



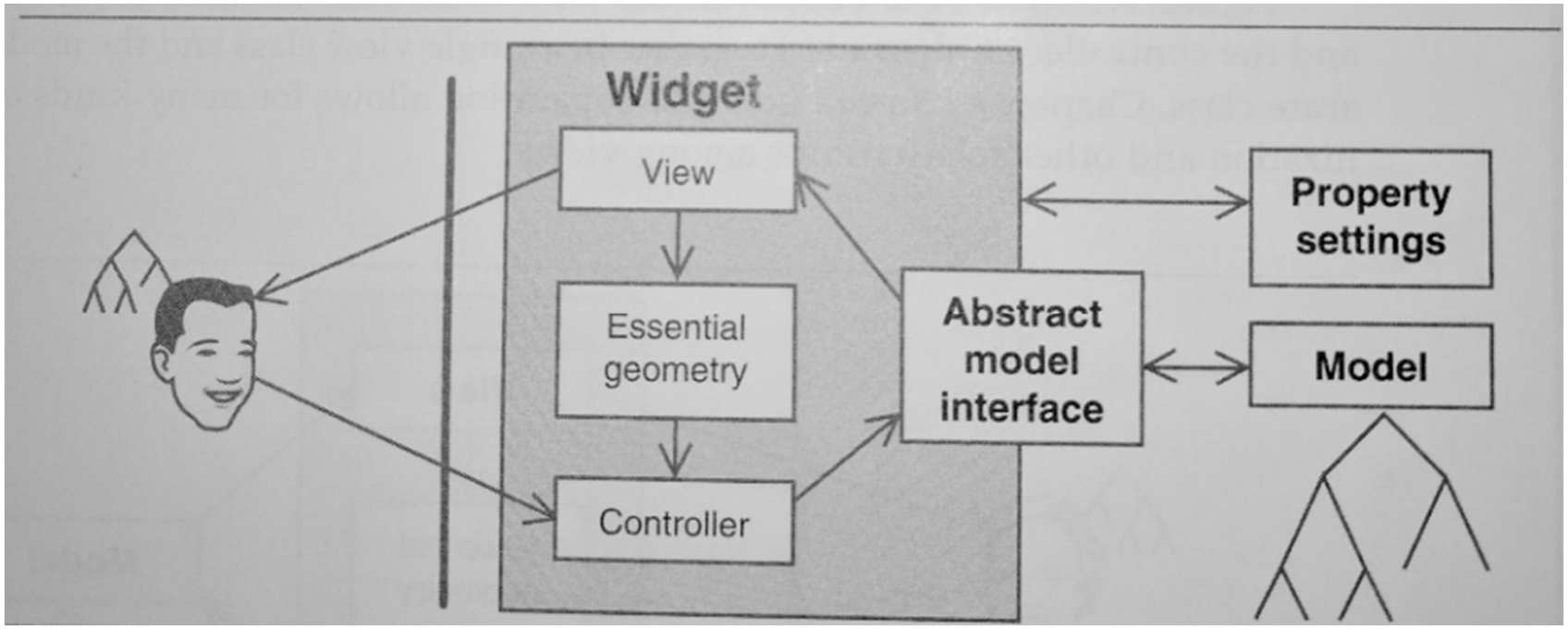
Colour picker

Abstract Model Widgets

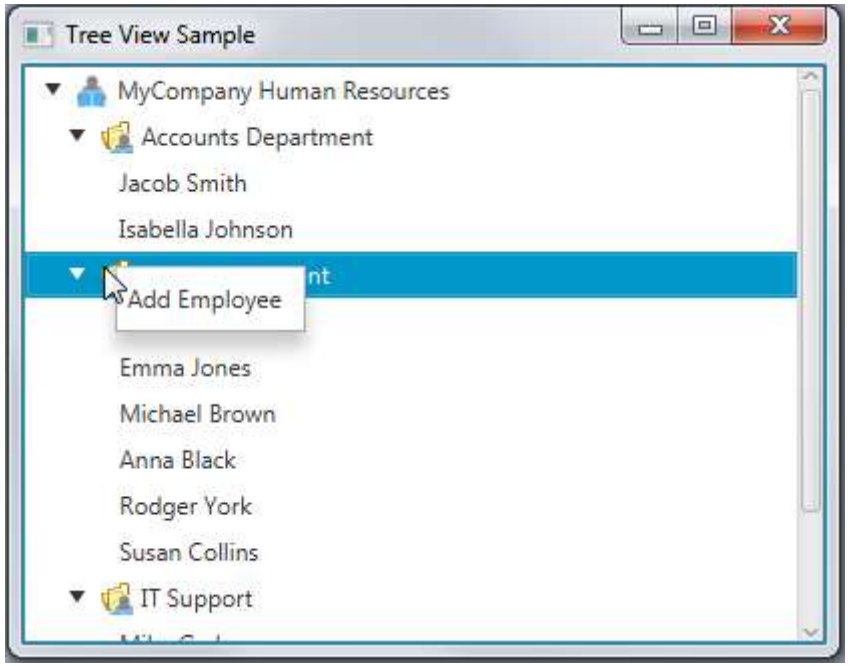
Abstract Model Widgets

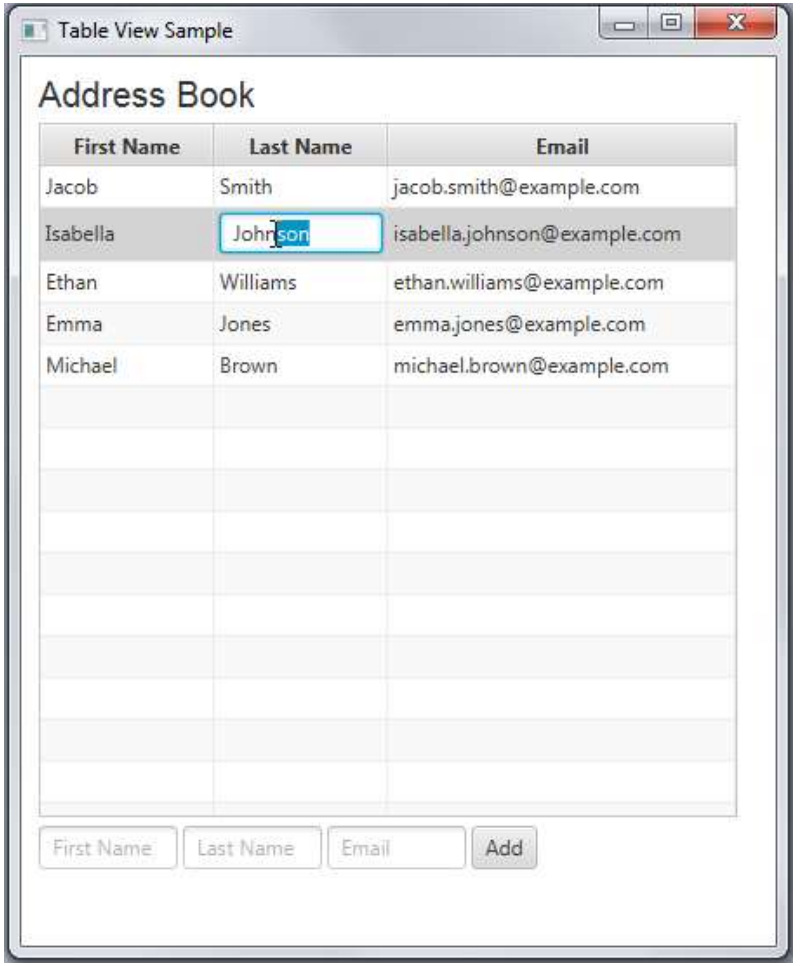
- More complex widgets require a detailed model of what is to be displayed
  - E.g., tree view / table view
  - Interaction is still defined in the widget, but model can be manipulated and populated separately

Interaction model for AMWs

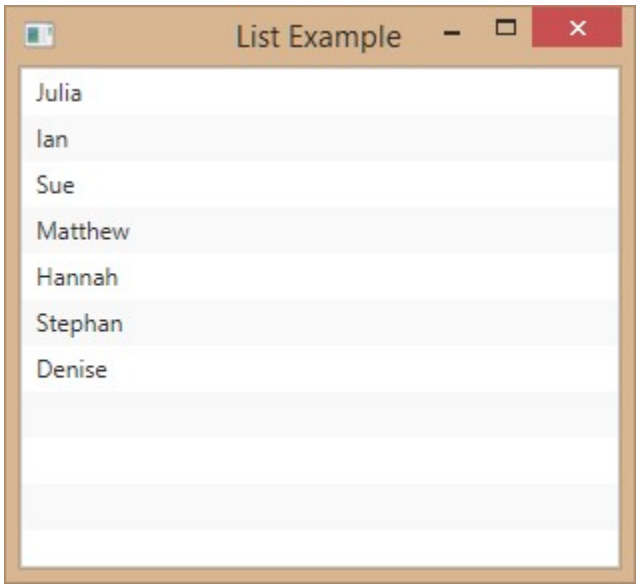


Tree Views and Table Views



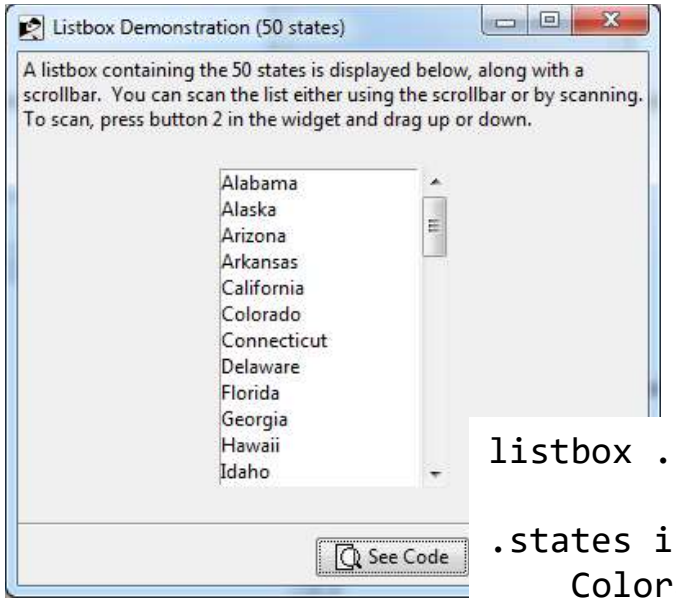


ListView widget in JavaFX




```
ObservableList<String> names = FXCollections.observableArrayList(
    "Julia", "Ian", "Sue", "Matthew", "Hannah", "Stephan", "Denise");
ListView<String> listView = new ListView<String>(names);
```

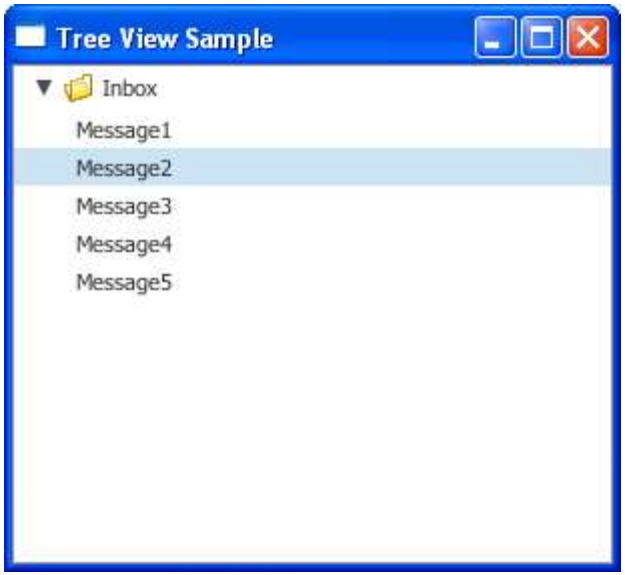
…compare with Tcl/Tk:




```
listbox .states -height 12

.states insert 0 Alabama Alaska Arizona Arkansas California \
    Colorado Connecticut Delaware Florida Georgia Hawaii Idaho \
    Indiana Iowa Kansas Kentucky Louisiana Maine Maryland \
    Massachusetts Michigan Minnesota Mississippi Missouri \
    Montana Nebraska Nevada "New Hampshire" "New Jersey" \
    "New York" "North Carolina" "North Dakota" \
    Ohio Oklahoma Oregon Pennsylvania "Rhode Island" \
    "South Carolina" "South Dakota" \
    Tennessee Texas Utah Vermont Virginia Washington \
    "West Virginia" Wisconsin Wyoming
```

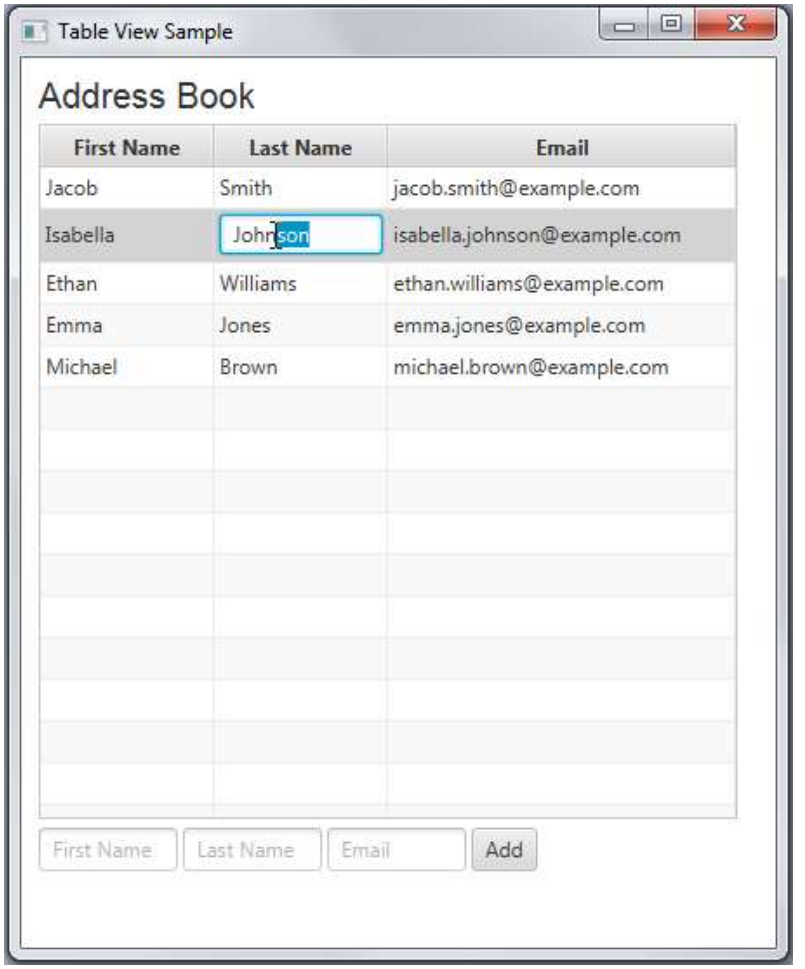
TreeView in JavaFX




```
TreeItem<String> rootItem = new TreeItem<String>(“Inbox”, rootIcon);
rootItem.setExpanded(true);
for (int i = 1; i < 6; i++) {
    TreeItem<String> item = new TreeItem<String> ("Message" + i);
    rootItem.getChildren().add(item);
}
TreeView<String> tree = new TreeView<String> (rootItem);
```

TableView in JavaFX

- Similar to ListView, but:
  - Rows instead of items
  - Multiple columns/row
  - Class to hold info
  - Class uses properties
  - Need to specify which info goes in which column

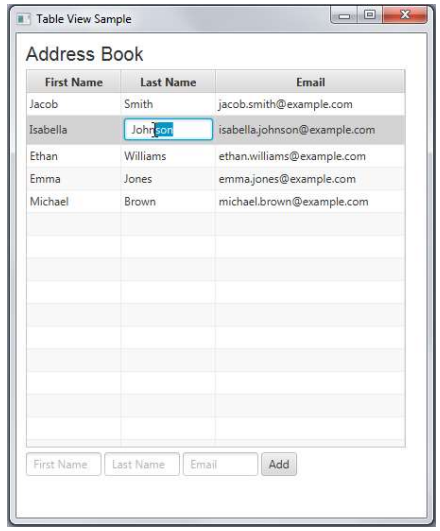


TableView in JavaFX

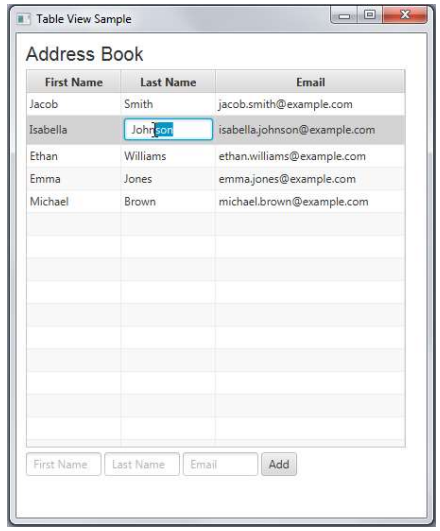
```
 public static class Person {
    private final SimpleStringProperty firstName;
    private final SimpleStringProperty lastName;
    private final SimpleStringProperty email;

    private Person(String fName, String lName, String email) {
        this.firstName = new SimpleStringProperty(fName);
        this.lastName = new SimpleStringProperty(lName);
        this.email = new SimpleStringProperty(email);
    }
}

final ObservableList<Person> data = FXCollections.observableArrayList(
    new Person("Jacob", "Smith", "jacob.smith@example.com"),
    new Person("Isabella", "Johnson", "isabella.johnson@example.com"),
    new Person("Ethan", "Williams", "ethan.williams@example.com"),
    new Person("Emma", "Jones", "emma.jones@example.com"),
    new Person("Michael", "Brown", "michael.brown@example.com")
);
```



TableView in JavaFX




```
private TableView table = new TableView();
TableColumn firstNameCol = new TableColumn("First Name");
TableColumn lastNameCol = new TableColumn("Last Name");
TableColumn emailCol = new TableColumn("Email");
table.getColumns().addAll(firstNameCol, lastNameCol, emailCol);

firstNameCol.setCellValueFactory(new
        PropertyValueFactory<Person,String>("firstName"));
lastNameCol.setCellValueFactory(new
        PropertyValueFactory<Person,String>("lastName"));
emailCol.setCellValueFactory(new
        PropertyValueFactory<Person,String>("email"));
```