Beispielcode

Modul Objektorientierte Programmierung 2 (oopI2)

Simon Wächter

2016

Inhalt

[1 Beispiel Addressviewer 2](#_Toc452897828)

[1.1 AddressViewerApplication.java 2](#_Toc452897829)

[1.2 AddressViewerWindow.java 3](#_Toc452897830)

[2 Beispiel Colorchooser 7](#_Toc452897831)

[2.1 ColorChooserApplication.java 7](#_Toc452897832)

[3 Beispiel Imageviewer 10](#_Toc452897833)

[3.1 ImageViewerApplication.java 10](#_Toc452897834)

[3.2 ImageViewerPanel.java 13](#_Toc452897835)

[4 Beispiel Calculator 15](#_Toc452897836)

[4.1 CalculatorApplication.java 15](#_Toc452897837)

[4.2 CalculatorWindow.java 15](#_Toc452897838)

[4.3 Style.css 17](#_Toc452897839)

# Beispiel Addressviewer

## AddressViewerApplication.java

|  |
| --- |
| **package** ch.swaechter.javafx.addressviewer;  **import** javafx.application.Application; **import** javafx.scene.Scene; **import** javafx.stage.Stage;  **import** java.io.\*;  **public class** AddressViewerApplication **extends** Application {   **private static** Stage *stage*;   **public static void** main(String[] args) {  *launch*(args);  *saveWindowDimensions*();  }   @Override  **public void** start(Stage primarystage) **throws** Exception {  Person persona = **new** Person();  persona.setFirstName(**"Hans"**);  persona.setName(**"Müller"**);   Person personb = **new** Person();  personb.setFirstName(**"Fritz"**);  personb.setName(**"Keller"**);   Person personc = **new** Person();  personc.setFirstName(**"Ueli"**);  personc.setName(**"Maurer"**);   File storagefile = **new** File(**"test.raf"**);  storagefile.delete();   RA\_Storage<Person> storage = **new** RA\_Storage<>(**"test.raf"**, Person.**class**);  storage.appendItem(persona);  storage.appendItem(personb);  storage.appendItem(personc);   *stage* = primarystage;   AddressViewerWindow addressviewerwindow = **new** AddressViewerWindow(storage);  primarystage.setMinWidth(400);  primarystage.setMinHeight(300);  primarystage.setWidth(400);  primarystage.setHeight(300);  primarystage.setScene(**new** Scene(addressviewerwindow, 300, 250));  primarystage.setTitle(**"Address Viewer"**);  *restoreWindowDimensions*();  primarystage.show();  }   **private static void** saveWindowDimensions() {  **try** {  BufferedWriter writer = **new** BufferedWriter(**new** FileWriter(**new** File(**"settings.ini"**)));  writer.write(**"posx = "** + *stage*.getX());  writer.newLine();  writer.write(**"posy = "** + *stage*.getY());  writer.newLine();  writer.write(**"width = "** + *stage*.getWidth());  writer.newLine();  writer.write(**"height = "** + *stage*.getHeight());  writer.close();  } **catch** (IOException exception) {  System.***out***.println(**"severe I/O error while writing ini-file"**);  }  }   **private static void** restoreWindowDimensions() {  **double** x = 100, y = 100, w = 300, h = 300;  File control = **new** File(**"settings.ini"**);  **if** (control.exists()) {  String line = **" "**, name, value;  **try** {  BufferedReader reader = **new** BufferedReader(**new** FileReader(control));  **int** separator;  **while** (line != **null**) {  line = reader.readLine();  **if** (line != **null**) {  separator = line.indexOf(**'='**);  **if** (separator < 0) {  reader.close();  **throw new** RuntimeException(**"ini-file contains a line of invalid format"**);  }  name = (line.substring(0, separator)).trim();  value = (line.substring(separator + 1, line.length())).trim();  **switch** (name) {  **case "posx"**:  x = Double.*parseDouble*(value);  **break**;  **case "posy"**:  y = Double.*parseDouble*(value);  **break**;  **case "width"**:  w = Double.*parseDouble*(value);  **break**;  **case "height"**:  h = Double.*parseDouble*(value);  **break**;  }  }  }  } **catch** (IOException e) {  System.***out***.println(**"severe I/O error while reading ini-file"**);  }  }  *stage*.setWidth(w);  *stage*.setHeight(h);  *stage*.setX(x);  *stage*.setY(y);  } } |

## AddressViewerWindow.java

|  |
| --- |
| **package** ch.swaechter.javafx.addressviewer;  **import** javafx.application.Platform; **import** javafx.event.ActionEvent; **import** javafx.event.EventHandler; **import** javafx.geometry.Insets; **import** javafx.geometry.Orientation; **import** javafx.geometry.Pos; **import** javafx.scene.control.\*; **import** javafx.scene.layout.BorderPane; **import** javafx.scene.layout.HBox; **import** javafx.scene.layout.Priority; **import** javafx.scene.layout.VBox; **import** javafx.scene.paint.Color;  **import** java.io.File;  **public class** AddressViewerWindow **extends** BorderPane **implements** EventHandler<ActionEvent> {   **private final** String[] **LABELS** = {**"Name"**, **"Adresse"**, **"Geburtsdatum"**, **"Kontostand"**, **"Geschlecht"**, **"Zivilstand"**, **"Brillenträger"**};   **private final** String **FORWARD** = **">"**;   **private final** String **BACKWARD** = **"<"**;   **private final** RA\_Storage<Person> **storage**;   **private** Person **buffer** = **new** Person();   **private final** TextField[] **inputfields** = **new** TextField[7];   **private final** Label **position**;   **private final** Label **status**;   **private final** Button **forwardbutton**;   **private final** Button **backwardbutton**;   **private final** Slider **slider**;   **private int index**;   **public** AddressViewerWindow(RA\_Storage<Person> storage) {  VBox labelbox = **new** VBox();  VBox inputbox = **new** VBox();  HBox controlbox = **new** HBox();  VBox buttonbox = **new** VBox();  VBox statusbox = **new** VBox();   setLeft(labelbox);  setCenter(inputbox);  setRight(controlbox);  setBottom(statusbox);  setPadding(**new** Insets(10, 10, 0, 10));   inputbox.setPadding(**new** Insets(0, 10, 0, 10));  inputbox.setSpacing(5);   buttonbox.setAlignment(Pos.***CENTER***);  buttonbox.setPrefHeight(150);   controlbox.setPadding(**new** Insets(20, 10, 20, 20));   statusbox.setPadding(**new** Insets(5, 0, 5, 0));   **for** (**int** i = 0; i < 7; ++i) {  Label label = **new** Label(**LABELS**[i]);  label.setMaxHeight(Double.***MAX\_VALUE***);  VBox.*setVgrow*(label, Priority.***ALWAYS***);  labelbox.getChildren().add(label);  }   **for** (**int** i = 0; i < 7; ++i) {  **inputfields**[i] = **new** TextField();  **inputfields**[i].setMaxHeight(Double.***MAX\_VALUE***);  **inputfields**[i].setEditable(**false**);  **inputfields**[i].setFocusTraversable(**false**);  **inputfields**[i].setOnMouseClicked(event -> requestFocus());  VBox.*setVgrow*(**inputfields**[i], Priority.***ALWAYS***);  inputbox.getChildren().add(**inputfields**[i]);  }   **forwardbutton** = **new** Button(**FORWARD**);  **forwardbutton**.setPrefWidth(50);  VBox.*setMargin*(**forwardbutton**, **new** Insets(2));  buttonbox.getChildren().add(**forwardbutton**);   **backwardbutton** = **new** Button(**BACKWARD**);  **backwardbutton**.setPrefWidth(50);  VBox.*setMargin*(**backwardbutton**, **new** Insets(2));  buttonbox.getChildren().add(**backwardbutton**);   **position** = **new** Label(**"position"**);  **position**.setPrefWidth(50);  **position**.setAlignment(Pos.***CENTER***);  VBox.*setMargin*(**position**, **new** Insets(2));  buttonbox.getChildren().add(**position**);   Button exitbutton = **new** Button(**"Exit"**);  exitbutton.setTooltip(**new** Tooltip(**"Close the application"**));  exitbutton.setPrefWidth(50);  VBox.*setMargin*(exitbutton, **new** Insets(2));  buttonbox.getChildren().add(exitbutton);   **slider** = **new** Slider();  **slider**.setOrientation(Orientation.***VERTICAL***);  **slider**.valueProperty().addListener((x, o, n) -> {  **index** = (**int**) (storage.itemCount()) - (**int**) ((Double) n + 0.5) - 1;  showRecord(**index**);  **position**.setText(**""** + (**index** + 1) + **"/"** + storage.itemCount());  });   controlbox.getChildren().add(**new** Separator(Orientation.***VERTICAL***));  controlbox.getChildren().add(buttonbox);  controlbox.getChildren().add(**slider**);   **status** = **new** Label(**"status"**);  VBox.*setMargin*(**status**, **new** Insets(5));   statusbox.getChildren().add(**new** Separator());  statusbox.getChildren().add(**status**);   **this**.**storage** = storage;  initData();   exitbutton.setOnAction(event -> {  storage.close();  Platform.*exit*();  });   **forwardbutton**.setOnAction(**this**);  **backwardbutton**.setOnAction(**this**);  }   @Override  **public void** handle(ActionEvent event) {  **if** (event.getSource() == **forwardbutton**) {  **if** (**index** < **storage**.itemCount() - 1) {  **index**++;  showRecord(**index**);  **position**.setText(**""** + (**index** + 1) + **"/"** + **storage**.itemCount());  }  } **else if** (event.getSource() == **backwardbutton**) {  **if** (**index** > 0) {  **index**--;  showRecord(**index**);  **position**.setText(**""** + (**index** + 1) + **"/"** + **storage**.itemCount());   }  }  }   **private void** initData() {  **if** (!**new** File(**"test.raf"**).exists()) {  **status**.setText(**"File not found!"**);  **status**.setTextFill(Color.***RED***);  } **else** {  **if** (**storage**.isConnected()) {  **status**.setText(**"Connected"**);  **status**.setTextFill(Color.***GREEN***);  **if** (**storage**.itemCount() > 0) {  **slider**.setMax(**storage**.itemCount() - 1);  **slider**.setMin(0);  **slider**.setMajorTickUnit(1);  **slider**.setMinorTickCount(0);  **slider**.setBlockIncrement(1);  **slider**.setShowTickLabels(**true**);  **slider**.setShowTickMarks(**true**);  **slider**.setValue(**slider**.getMax());  **slider**.setLabelFormatter(**new** LabelConverter(**storage**.itemCount()));  **slider**.setSnapToPixel(**true**);  **position**.setText(**"1"** + **"/"** + **storage**.itemCount());  showRecord(0);  } **else** {  **position**.setText(**"0 / 0"**);  }  } **else** {  **status**.setText(**"Invalid file format!"**);  }  }  }   **private void** showRecord(**int** record) {  **if** (record >= 0 && record <= **index** && **storage**.isConnected()) {  **storage**.readItem(record, **buffer**);  **int** b = **buffer**.getBirthdate();  **inputfields**[0].setText(**buffer**.getLastName() + **" "** + **buffer**.getFirstName());  **inputfields**[1].setText(**buffer**.getAddress());  **inputfields**[2].setText(String.*format*(**"%02d"**, b % 100) + **"."** + String.*format*(**"%02d"**, b / 100 % 100) + **"."** + String.*format*(**"%04d"**, b / 10000));  **inputfields**[3].setText(String.*format*(**"%.2f"**, **buffer**.getMoney()));  **inputfields**[4].setText(**buffer**.isFemale() ? **"Weiblich"** : **"Männlich"**);  **inputfields**[5].setText(**buffer**.isMarried() ? **"Verheiratet"** : **"Unverheiratet"**);  **inputfields**[6].setText(**buffer**.wearsGlasses() ? **"Ja"** : **"Nein"**);  **slider**.setValue(**storage**.itemCount() - **index** - 1);  }  } } |

# Beispiel Colorchooser

## ColorChooserApplication.java

|  |
| --- |
| **package** ch.swaechter.javafx.colorchooser;  **import** javafx.application.Application; **import** javafx.beans.InvalidationListener; **import** javafx.beans.Observable; **import** javafx.beans.binding.Bindings; **import** javafx.beans.binding.DoubleBinding; **import** javafx.beans.property.DoubleProperty; **import** javafx.geometry.Insets; **import** javafx.geometry.Orientation; **import** javafx.geometry.Pos; **import** javafx.scene.Scene; **import** javafx.scene.control.Label; **import** javafx.scene.control.Slider; **import** javafx.scene.layout.\*; **import** javafx.scene.paint.Color; **import** javafx.stage.Stage;  **public class** ColorChooserApplication **extends** Application **implements** InvalidationListener {   **private** StackPane **stackpane**;   **private** Slider **rgbsliders**[] = **new** Slider[3];   **private** Slider **hsbsliders**[] = **new** Slider[3];   **public static void** main(String[] args) {  *launch*(args);  }   @Override  **public void** start(Stage primarystage) **throws** Exception {  BorderPane borderpane = **new** BorderPane();  borderpane.setPadding(**new** Insets(10, 10, 10, 10));   **stackpane** = **new** StackPane();  **stackpane**.setBackground(**new** Background(**new** BackgroundFill(**new** Color(0.5, 0.5, 0.5, 1), **new** CornerRadii(20), **new** Insets(10))));   HBox rgbcontrol = **new** HBox();  HBox hsbcontrol = **new** HBox();   String[] rgblabels = {**"R"**, **"G"**, **"B"**};  **for** (**int** i = 0; i < **rgbsliders**.**length**; i++) {  **rgbsliders**[i] = **new** Slider();  **rgbsliders**[i].setOrientation(Orientation.***VERTICAL***);  **rgbsliders**[i].setPrefWidth(50);  **rgbsliders**[i].setMax(1);  **rgbsliders**[i].valueProperty().addListener(**this**);  **rgbsliders**[i].setOnMousePressed((e) -> {  bindRGBtoHSB();  });   VBox.*setVgrow*(**rgbsliders**[i], Priority.***ALWAYS***);   VBox vbox = **new** VBox();  vbox.setAlignment(Pos.***CENTER***);  vbox.getChildren().add(**rgbsliders**[i]);  vbox.getChildren().add(**new** Label(rgblabels[i]));  rgbcontrol.getChildren().add(vbox);  }   String[] hsblabels = {**"H"**, **"S"**, **"B"**};  **for** (**int** i = 0; i < **hsbsliders**.**length**; i++) {  **hsbsliders**[i] = **new** Slider();  **hsbsliders**[i].setOrientation(Orientation.***VERTICAL***);  **hsbsliders**[i].setPrefWidth(50);  **hsbsliders**[i].setMax(1);  **hsbsliders**[i].valueProperty().addListener(**this**);  **hsbsliders**[i].setOnMousePressed((e) -> {  bindHSBtoRGB();  });   VBox.*setVgrow*(**hsbsliders**[i], Priority.***ALWAYS***);   VBox vbox = **new** VBox();  vbox.setAlignment(Pos.***CENTER***);  vbox.getChildren().add(**hsbsliders**[i]);  vbox.getChildren().add(**new** Label(hsblabels[i]));  hsbcontrol.getChildren().add(vbox);  }   borderpane.setCenter(**stackpane**);  borderpane.setLeft(rgbcontrol);  borderpane.setRight(hsbcontrol);   primarystage.setScene(**new** Scene(borderpane, 400, 300));  primarystage.setMinWidth(600);  primarystage.setMinHeight(300);  primarystage.setTitle(**"Color Chooser"**);  primarystage.show();   invalidated(**null**);  }   @Override  **public void** invalidated(Observable observable) {  Color color = **new** Color(**rgbsliders**[0].getValue(), **rgbsliders**[1].getValue(), **rgbsliders**[2].getValue(), 1);  BackgroundFill fill = **new** BackgroundFill(color, **new** CornerRadii(20), **new** Insets(10, 10, 10, 10));  **stackpane**.setBackground(**new** Background(fill));  }   **public void** bindRGBtoHSB() {  **for** (Slider slider : **rgbsliders**) {  slider.valueProperty().unbind();  }   DoubleProperty red = **rgbsliders**[0].valueProperty();  DoubleProperty green = **rgbsliders**[1].valueProperty();  DoubleProperty blue = **rgbsliders**[2].valueProperty();   DoubleBinding hsbh = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(red.get(), green.get(), blue.get(), 1);  **return** color.getHue();  }, red, green, blue);   DoubleBinding hsbs = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(red.get(), green.get(), blue.get(), 1);  **return** color.getSaturation();  }, red, green, blue);   DoubleBinding hsbb = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(red.get(), green.get(), blue.get(), 1);  **return** color.getBrightness();  }, red, green, blue);   **hsbsliders**[0].valueProperty().bind(hsbh);  **hsbsliders**[1].valueProperty().bind(hsbs);  **hsbsliders**[2].valueProperty().bind(hsbb);  }   **public void** bindHSBtoRGB() {  **for** (Slider slider : **hsbsliders**) {  slider.valueProperty().unbind();  }   DoubleProperty hue = **hsbsliders**[0].valueProperty();  DoubleProperty saturation = **hsbsliders**[1].valueProperty();  DoubleProperty brightness = **hsbsliders**[2].valueProperty();   DoubleBinding rgbred = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(hue.get(), saturation.get(), brightness.get(), 1);  **return** color.getRed();  }, hue, saturation, brightness);   DoubleBinding rgbgreen = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(hue.get(), saturation.get(), brightness.get(), 1);  **return** color.getBlue();  }, hue, saturation, brightness);   DoubleBinding rgbblue = Bindings.*createDoubleBinding*(() -> {  Color color = **new** Color(hue.get(), saturation.get(), brightness.get(), 1);  **return** color.getGreen();  }, hue, saturation, brightness);   **rgbsliders**[0].valueProperty().bind(rgbred);  **rgbsliders**[1].valueProperty().bind(rgbgreen);  **rgbsliders**[2].valueProperty().bind(rgbblue);  } } |

# Beispiel Imageviewer

## ImageViewerApplication.java

|  |
| --- |
| **package** ch.swaechter.javafx.imageviewer;  **import** javafx.application.Application; **import** javafx.collections.FXCollections; **import** javafx.collections.ObservableList; **import** javafx.geometry.Insets; **import** javafx.geometry.Orientation; **import** javafx.scene.Scene; **import** javafx.scene.control.\*; **import** javafx.scene.layout.BorderPane; **import** javafx.scene.layout.HBox; **import** javafx.scene.layout.VBox; **import** javafx.stage.DirectoryChooser; **import** javafx.stage.Stage;  **import** java.io.\*; **import** java.util.Locale;  **public class** ImageViewerApplication **extends** Application {   **private static final** String[] ***BUTTON\_NAMES*** = {**"<<<"**, **"<"**, **">"**, **">>>"**, **"Open"**};   **private static final** String[] ***SIZE\_PREFIXES*** = {**" bytes"**, **"KB"**, **"MB"**, **" GB"**, **" TB"**};   **private final** ListView<File> **filelistview** = **new** ListView<>();   **private final** ImageViewerPanel **imagepanel** = **new** ImageViewerPanel();   **private final** TextField **infotextfield** = **new** TextField();   **private** File **currentpath**;   **private** File[] **currentfiles**;   **private int currentindex**;   **public static void** main(String[] args) {  *launch*(args);  }   @Override  **public void** start(Stage primarystage) **throws** IOException {  BorderPane borderpane = **new** BorderPane();  borderpane.setPadding(**new** Insets(10, 10, 10, 10));   primarystage.setScene(**new** Scene(borderpane));  primarystage.setTitle(**"Image Viewer"**);  primarystage.setMinWidth(700);  primarystage.setMinHeight(360);   VBox controlbox = **new** VBox();  HBox elementbox = **new** HBox();   elementbox.getChildren().addAll(**new** Separator(Orientation.***VERTICAL***), controlbox);   ObservableList<File> data = FXCollections.*observableArrayList*();  **filelistview**.setItems(data);  **filelistview**.setCellFactory(e -> **new** FileListCell());  SelectionModel<File> selection = **filelistview**.getSelectionModel();  selection.selectedIndexProperty().addListener(e -> {  **currentindex** = selection.getSelectedIndex();  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  updateStatus();  });   **imagepanel**.setMinSize(100, 100);   **infotextfield**.setEditable(**false**);   borderpane.setLeft(**filelistview**);  borderpane.setCenter(**imagepanel**);  borderpane.setRight(elementbox);  borderpane.setBottom(**infotextfield**);  BorderPane.*setMargin*(controlbox, **new** Insets(20, 10, 0, 20));   Button[] buttons = **new** Button[***BUTTON\_NAMES***.**length**];  **for** (**int** i = 0; i < ***BUTTON\_NAMES***.**length**; ++i) {  buttons[i] = **new** Button(***BUTTON\_NAMES***[i]);  buttons[i].setPrefWidth(50);  VBox.*setMargin*(buttons[i], **new** Insets(10, 0, 10, 0));  controlbox.getChildren().add(buttons[i]);  }   VBox.*setMargin*(buttons[4], **new** Insets(50, 0, 10, 0));   buttons[0].setOnAction(e -> {  **if** (**currentfiles** != **null**) {  **currentindex** = 0;  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  selection.select(**currentindex**);  **filelistview**.scrollTo(**currentindex**);  updateStatus();  }  });   buttons[1].setOnAction(e -> {  **if** (**currentfiles** != **null** && **currentindex** > 0) {  **currentindex**--;  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  selection.select(**currentindex**);  **filelistview**.scrollTo(**currentindex**);  updateStatus();  }  });   buttons[2].setOnAction(e -> {  **if** (**currentfiles** != **null** && **currentindex** < **currentfiles**.**length** - 1) {  **currentindex**++;  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  selection.select(**currentindex**);  **filelistview**.scrollTo(**currentindex**);  updateStatus();  }  });   buttons[3].setOnAction(e -> {  **if** (**currentfiles** != **null**) {  **currentindex** = **currentfiles**.**length** - 1;  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  selection.select(**currentindex**);  **filelistview**.scrollTo(**currentindex**);  updateStatus();  }  });   buttons[4].setOnAction(e -> {  DirectoryChooser directorychooser = **new** DirectoryChooser();  directorychooser.setInitialDirectory(**currentpath**);  File newcurrentpath = directorychooser.showDialog(primarystage);  **if** (newcurrentpath != **null**) {  **currentpath** = newcurrentpath;  **currentfiles** = **currentpath**.listFiles();  **if** (**currentfiles**.**length** == 0) {  **currentfiles** = **null**;  } **else** {  **currentindex** = 0;  data.clear();  data.setAll(**currentfiles**);  **imagepanel**.setPicture(**currentfiles**[**currentindex**]);  directorychooser.setInitialDirectory(**currentpath**);  }  }  updateStatus();  });   updateStatus();   primarystage.show();  }   @Override  **public void** init() **throws** IOException {  File ini = **new** File(**"picview.ini"**);  **if** (!ini.exists()) {  **currentpath** = **new** File(**"c:/"**);  } **else** {  BufferedReader textIn = **new** BufferedReader(**new** FileReader(ini));  **currentpath** = **new** File(textIn.readLine());  textIn.close();  }  }   @Override  **public void** stop() **throws** IOException {  **if** (**currentpath** != **null**) {  File ini = **new** File(**"picview.ini"**);  BufferedWriter textOut = **new** BufferedWriter(**new** FileWriter(ini));  textOut.write(**currentpath**.getAbsolutePath());  textOut.newLine();  textOut.close();  }  }   **public void** updateStatus() {  **if** (**currentfiles** == **null** || **currentfiles**.**length** == 0) {  **infotextfield**.setText(**"No folder open"**);  } **else** {  **int** scale = 0;  **double** size = **currentfiles**[**currentindex**].length();  **while** (size >= 1024) {  size /= 1024;  ++scale;  }  **if** (scale > 0) {  **infotextfield**.setText(**" "** + (**currentindex** + 1) + **" / "** + **currentfiles**.**length** + **" ("** + String.*format*(Locale.***ENGLISH***, **"%.1f"**, size) + ***SIZE\_PREFIXES***[scale] + **")"**);  } **else** {  **infotextfield**.setText(**" "** + (**currentindex** + 1) + **" / "** + **currentfiles**.**length** + **" ("** + String.*format*(**"%.0f"**, size) + ***SIZE\_PREFIXES***[scale] + **")"**);  }  }  }   **private static class** FileListCell **extends** ListCell<File> {  @Override  **protected void** updateItem(File item, **boolean** empty) {  **super**.updateItem(item, empty);  **if** (!empty) setText(item.getName());  }  } } |

## ImageViewerPanel.java

|  |
| --- |
| **package** ch.swaechter.javafx.imageviewer;  **import** javafx.scene.effect.DropShadow; **import** javafx.scene.image.Image; **import** javafx.scene.image.ImageView; **import** javafx.scene.layout.StackPane; **import** javafx.scene.paint.Color; **import** javafx.scene.shape.Rectangle; **import** javafx.scene.text.Font; **import** javafx.scene.text.FontSmoothingType; **import** javafx.scene.text.Text;  **import** java.io.File; **import** java.net.MalformedURLException;  **public class** ImageViewerPanel **extends** StackPane {   **private** Image **currentimage**;   **private** ImageView **imageview** = **new** ImageView();   **private** Rectangle **border** = **new** Rectangle();   **private** Text **text** = **new** Text();   **public** ImageViewerPanel() {  setMinSize(0, 0);   **imageview**.setPreserveRatio(**true**);  **imageview**.fitWidthProperty().bind(widthProperty().subtract(10));  **imageview**.fitHeightProperty().bind(heightProperty().subtract(10));  **imageview**.setEffect(**new** DropShadow(10, 5, 5, **new** Color(0, 0, 0, 0.3)));   **border**.setFill(**null**);  **border**.setStroke(Color.***BLACK***);   **text**.setFont(**new** Font(24));  **text**.setFontSmoothingType(FontSmoothingType.***LCD***);   getChildren().addAll(**imageview**, **border**, **text**);   widthProperty().addListener(event -> {  drawBorder();  });   heightProperty().addListener(event -> {  drawBorder();  });  }   **public void** setPicture(File file) {  **try** {  **currentimage** = **new** Image(file.toURI().toURL().toString());  **imageview**.setImage(**currentimage**);  **if** (**currentimage**.getHeight() == 0) {  **text**.setText(file.getName());  } **else** {  **text**.setText(**new** String());  }  drawBorder();  } **catch** (MalformedURLException e) {  }  }    **public void** drawBorder() {  **if** (**currentimage** == **null**) **return**;  **double** originalratio = **currentimage**.getWidth() / **currentimage**.getHeight();  **double** displayedratio = **imageview**.getFitWidth() / **imageview**.getFitHeight();  **if** (originalratio < displayedratio) {  **border**.setWidth(**imageview**.getFitHeight() \* originalratio + 3);  **border**.setHeight(**imageview**.getFitHeight() + 3);  } **else** {  **border**.setWidth(**imageview**.getFitWidth() + 3);  **border**.setHeight(**imageview**.getFitWidth() / originalratio + 3);  }  } } |

# Beispiel Calculator

## CalculatorApplication.java

|  |
| --- |
| **package** ch.swaechter.javafx.imageviewer;  **import** javafx.application.Application; **import** javafx.scene.Scene; **import** javafx.stage.Stage;  **import** java.io.IOException;  **public class** CalculatorApplication **extends** Application {   **public static void** main(String[] args) {  *launch*(args);  }   @Override  **public void** start(Stage primarystage) **throws** IOException {  Scene scene = **new** Scene(**new** CalculatorWindow());  scene.getStylesheets().add(**this**.getClass().getResource(**"/Style.css"**).toExternalForm());   primarystage.setScene(scene);  primarystage.setTitle(**"Image Viewer"**);  primarystage.setMinWidth(400);  primarystage.setMinHeight(300);  primarystage.show();  } } |

## CalculatorWindow.java

|  |
| --- |
| **package** ch.swaechter.javafx.imageviewer;  **import** javafx.scene.control.Button; **import** javafx.scene.control.TextField; **import** javafx.scene.layout.ColumnConstraints; **import** javafx.scene.layout.GridPane; **import** javafx.scene.layout.Priority; **import** javafx.scene.layout.RowConstraints;  **public class** CalculatorWindow **extends** GridPane {   **public** CalculatorWindow() {  setId(**"gridpane"**);   ColumnConstraints columnconstraincs = **new** ColumnConstraints();  columnconstraincs.setHgrow(Priority.***ALWAYS***);  getColumnConstraints().addAll(columnconstraincs, columnconstraincs, columnconstraincs, columnconstraincs);   RowConstraints rowconstraints = **new** RowConstraints();  rowconstraints.setVgrow(Priority.***ALWAYS***);  getRowConstraints().addAll(**new** RowConstraints(), rowconstraints, rowconstraints, rowconstraints, rowconstraints, rowconstraints);   TextField lcdpanel = **new** TextField();  lcdpanel.setEditable(**false**);  lcdpanel.setFocusTraversable(**false**);   Button buttonclear = **new** Button(**"C"**);  buttonclear.setMaxWidth(Double.***MAX\_VALUE***);  buttonclear.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonplusminus = **new** Button(**"+/-"**);  buttonplusminus.setMaxWidth(Double.***MAX\_VALUE***);  buttonplusminus.setMaxHeight(Double.***MAX\_VALUE***);   Button buttondivide = **new** Button(**"/"**);  buttondivide.setMaxWidth(Double.***MAX\_VALUE***);  buttondivide.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonmultiply = **new** Button(**"\*"**);  buttonmultiply.setMaxWidth(Double.***MAX\_VALUE***);  buttonmultiply.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonseven = **new** Button(**"7"**);  buttonseven.setMaxWidth(Double.***MAX\_VALUE***);  buttonseven.setMaxHeight(Double.***MAX\_VALUE***);   Button buttoneight = **new** Button(**"8"**);  buttoneight.setMaxWidth(Double.***MAX\_VALUE***);  buttoneight.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonnine = **new** Button(**"9"**);  buttonnine.setMaxWidth(Double.***MAX\_VALUE***);  buttonnine.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonsubstract = **new** Button(**"-"**);  buttonsubstract.setMaxWidth(Double.***MAX\_VALUE***);  buttonsubstract.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonfour = **new** Button(**"4"**);  buttonfour.setMaxWidth(Double.***MAX\_VALUE***);  buttonfour.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonfive = **new** Button(**"5"**);  buttonfive.setMaxWidth(Double.***MAX\_VALUE***);  buttonfive.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonsix = **new** Button(**"6"**);  buttonsix.setMaxWidth(Double.***MAX\_VALUE***);  buttonsix.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonadd = **new** Button(**"+"**);  buttonadd.setMaxWidth(Double.***MAX\_VALUE***);  buttonadd.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonone = **new** Button(**"1"**);  buttonone.setMaxWidth(Double.***MAX\_VALUE***);  buttonone.setMaxHeight(Double.***MAX\_VALUE***);   Button buttontwo = **new** Button(**"2"**);  buttontwo.setMaxWidth(Double.***MAX\_VALUE***);  buttontwo.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonthree = **new** Button(**"3"**);  buttonthree.setMaxWidth(Double.***MAX\_VALUE***);  buttonthree.setMaxHeight(Double.***MAX\_VALUE***);   Button buttonequal = **new** Button(**"="**);  buttonequal.setMaxWidth(Double.***MAX\_VALUE***);  buttonequal.setMaxHeight(Double.***MAX\_VALUE***);  buttonequal.setId(**"equal"**);   Button buttonzero = **new** Button(**"0"**);  buttonzero.setMaxWidth(Double.***MAX\_VALUE***);  buttonzero.setMaxHeight(Double.***MAX\_VALUE***);   Button buttondot = **new** Button(**"."**);  buttondot.setMaxWidth(Double.***MAX\_VALUE***);  buttondot.setMaxHeight(Double.***MAX\_VALUE***);   add(lcdpanel, 0, 0, 4, 1);   add(buttonclear, 0, 1, 1, 1);  add(buttonplusminus, 1, 1, 1, 1);  add(buttondivide, 2, 1, 1, 1);  add(buttonmultiply, 3, 1, 1, 1);   add(buttonseven, 0, 2, 1, 1);  add(buttoneight, 1, 2, 1, 1);  add(buttonnine, 2, 2, 1, 1);  add(buttonsubstract, 3, 2, 1, 1);   add(buttonfour, 0, 3, 1, 1);  add(buttonfive, 1, 3, 1, 1);  add(buttonsix, 2, 3, 1, 1);  add(buttonadd, 3, 3, 1, 1);   add(buttonone, 0, 4, 1, 1);  add(buttontwo, 1, 4, 1, 1);  add(buttonthree, 2, 4, 1, 1);  add(buttonequal, 3, 4, 1, 2);   add(buttonzero, 0, 5, 2, 1);  add(buttondot, 2, 5, 1, 1);  } } |

## Style.css

|  |
| --- |
| .**button**:**hover** {  **-fx-background-color**: **derive**(**grey**, -50%); }  **#equal** {  **-fx-background-color**: **orange**; } |