Graphische Nutzerschnittstellen

- Dipl.-Inf., Dipl.-Ing. (FH) Michael Wilhelm
- Hochschule Harz
- FB Automatisierung und Informatik
- mwilhelm@hs-harz.de
- http://www.miwilhelm.de
- Raum 2.202
- Tel. 03943 / 659 338

Inhalt

Java Historie

- Frame
  - Einfaches Fenster (à Taschenrechner), AWT
- JFrame
  - Einfaches Fenster (à Taschenrechner), Swing
- SWTFrame
  - Einfaches Fenster (à Taschenrechner), SWT (Eclipse)
- Stage
  - Java FX

Typen der Fenster

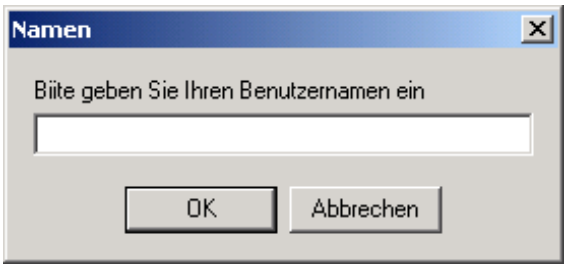
- Dialoge
  - Einfaches Fenster (à Taschenrechner)
- Single Document Interface (SDI)
  - Mit Menüs, Schalterleisten
- Multi Document Interface
  - Hauptfenster mit dem DesktopPane
  - Clientfenster als JInternalFrame
  - Mit Menüs, Schalterleisten
- Register-Frames
  - à la Browser

Dialog - Benutzeroberflächen in Java

Ein Dialogfenster ist ein Fenster, in dem der Benutzer Daten eingeben kann. Das Fenster mindestens einen Schalter (Ok). Mit Betätigen wird eine Aktion ausgelöst.

Beispiel: Meldung

Beispiele für Dialogfenster

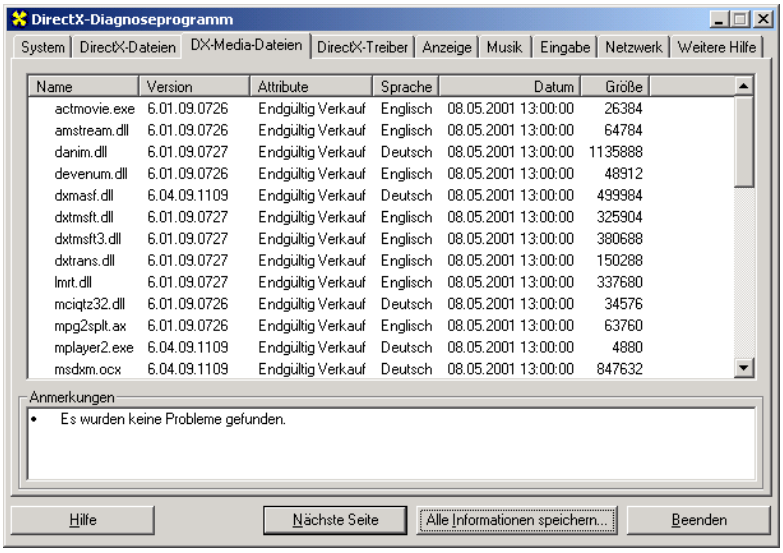


Eigenschaften:

- Titel
- Aktives Element
- Schalter Abbruch
- Schalter Ok

Mit Betätigen des Schalters „Ok“ wird eine weitere Aktion ausgeführt.

Beispiele für Dialogfenster



GUI-Elemente in Java (aktive Elemente)

Label	Anzeigefeld
TextField	einzeiliges Editorfeld
JButton	Schalter (Text, Bild)
ToggleButton	Schalter (Text, Bild)
RadioButton	Element zur Auswahl, korrespondiert mit anderen
CheckBox	Element zur Auswahl
ChoiceBox	Element zur Auswahl
ButtonGroup	Umfasst Checkbox, Radiobuttons
ComboBox	Aufklappbare Liste
ListView	Liste
TextArea	Editor, ASCII
TableView	Tabelle
TreeView	Anzeige der Elemente in einem Baum
TreeTableView	
WebView	Internet

GUI-Elemente in Java (Container)

JPanel	Rahmen
Accordion	à la CardLayout
StackPane	à la CardLayout
Pagination	à la Installationsprogramme
TabPane / Tab	Register
Scrollbar	Speichert GUI-Element, Scroll-Mechanismus
SplitPane	Automatischer Aufteiler (MDI)
Layouts	VBox, HBox, BorderPane, AnchorPane, FlowPane, StackPane, TitlePane, GridPane

GUI-Elemente in Java (Menüs, Schalter)

MenuBar	Speichert einzelne Menüs
ContextMernu	Lokales Menü
ColorChooser	Farbe
FileChooser	Auswahl einer Datei

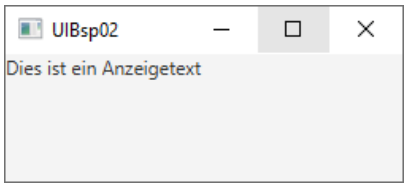
Aufbau eines Dialogfenster in Java

```
import javafx.application.Application;
import javafx.stage.Stage;

public class UIBsp01 extends Application {

    @Override
    public void start(Stage stage) {
        stage.show();
    }

    public static void main(String[] argv) {
        launch(argv);
    }
}
```

Originalframe. Ohne weitere GUI-Elemente

Dialogfenster in Java: UIBsp02.java

```
public class Test2 extends Application {

    @Override
    public void start(Stage stage) {
        VBox root = new VBox();
        Label label = new Label("Dies ist ein Anzeigetext");
        root.getChildren().add(label);
        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp02");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] argv) {
        launch(argv);
    }
}
```



GUI-Element: Label

Ein Label dient zur Beschriftung von GUI-Elementen.
Es kann eine Zeile Text dargestellt werden.

Konstruktoren:

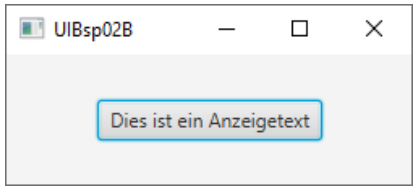
Label()	// Leerstring als Text
Label(String text)	// Label mit Text
Label(String text, Node node)	// Zusatzelement (Image)

UI-Basisklasse: Labeled (Label, Button)

- **void setAlignment(Pos value)**
- void setContentDisplay(ContentDisplay value)
- void setEllipsisString(String value)
- **void setFont(Font value)**
- void setGraphic(Node value)
- void setGraphicTextGap(double value)
- void setLineSpacing(double value)
- void setMnemonicParsing(boolean value)
- void **setText**(String value)
- void **setTextAlignment**(TextAlignment value)
- void setTextFill(Paint value)
- void setTextOverrun(OverrunStyle value)
- void setUnderline(boolean value)
- void **setWrapText**(boolean value)

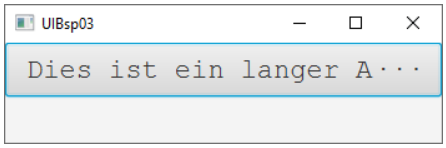
Dialogfenster in Java: UIBsp02B.java

```
public class UIBsp02 extends Application {

    @Override
    public void start(Stage stage) {
        VBox root = new VBox();
        Button bn = new Button("Dies ist ein Anzeigetext");
        root.getChildren().add(bn);
        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp02B");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] argv) {
        launch(argv);
    }
}
```



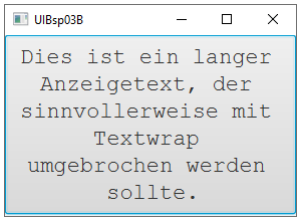
Dialogfenster in Java: UIBsp03.java

```
public class UIBsp03 extends Application {

    @Override
    public void start(Stage stage) {
        VBox root = new VBox();
        root.getChildren().add(new Button("Dies ist ein langer Anzeigetext, der
sinnvollerweise mit Textwrap umgebrochen werden sollte.") );
        bn.setEllipsisString("…");
        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp03");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] argv) {
        launch(argv);
    }
}
```



Dialogfenster in Java: UIBsp03B.java

```
public class UIBsp03 extends Application {
    @Override
    public void start(Stage stage) {
        VBox root = new VBox();
        root.getChildren().add(new Button("Dies ist ein langer Anzeigetext, der
sinnvollerweise mit Textwrap umgebrochen werden sollte.") );
        bn.setWrapText(true);
        bn.setTextAlignment(TextAlignment.CENTER);
        Scene scene= new Scene(root, 290, 180);
        stage.setTitle("UIBsp03B");
        stage.setScene(scene);
        stage.show();
    }
    public static void main(String[] argv) {
        launch(argv);
    }
}
```



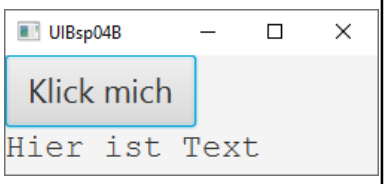
Dialogfenster in Java: UIBsp04.java

```
@Override
    public void start(Stage stage) {
        StackPane root = new StackPane();
        Button bn = new Button("Klick mich") ;
        bn.setFont(new Font(22));
        root.getChildren().add(bn);

        Label label = new Label("Hier ist Text") ;
        label.setFont(new Font("Courier New",22));
        root.getChildren().add(label);

        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp04");
        stage.setScene(scene);
        stage.show();
    }
```

StackPane: CardLayout

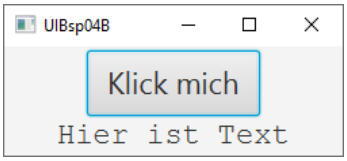
Dialogfenster in Java: UIBsp04.java

```
public void start(Stage stage) {
        VBox root = new VBox();

        Button bn = new Button("Klick mich") ;
        bn.setWrapText(true);
        bn.setTextAlignment(TextAlignment.CENTER);
        bn.setFont(new Font(22));
        root.getChildren().add(bn);

        Label label = new Label("Hier ist Text") ;
        label.setWrapText(true);
        label.setTextAlignment(TextAlignment.CENTER);
        label.setFont(new Font("Courier New",22));
        root.getChildren().add(label);

        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp04B");
        stage.setScene(scene);
        stage.show();
}
```



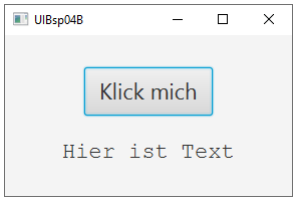
Dialogfenster in Java: UIBsp04B.java

```
public void start(Stage stage) {
        VBox root = new VBox();
        root.setAlignment(Pos.CENTER);

        Button bn = new Button("Klick mich") ;
        bn.setWrapText(true);
        bn.setTextAlignment(TextAlignment.CENTER);
        bn.setFont(new Font(22));
        root.getChildren().add(bn);

        Label label = new Label("Hier ist Text") ;
        label.setWrapText(true);
        label.setTextAlignment(TextAlignment.CENTER);
        label.setFont(new Font("Courier New",22));
        root.getChildren().add(label);

        Scene scene= new Scene(root, 250, 80);
        stage.setTitle("UIBsp04B");
        stage.setScene(scene);
        stage.show();
}
```

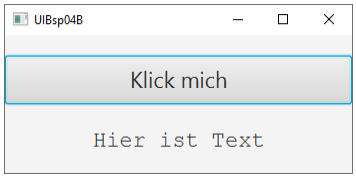


Dialogfenster in Java: UIBsp04B.java

VBox root = new VBox(22);

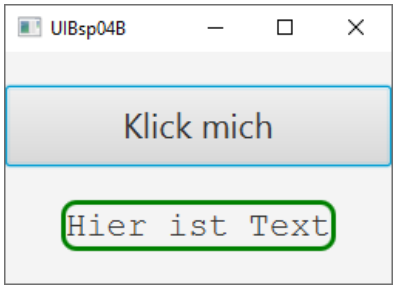


setFillWidth(true)
bn.setMaxWidth(Double.POSITIVE_INFINITY);



Dialogfenster in Java: UIBsp04B.java

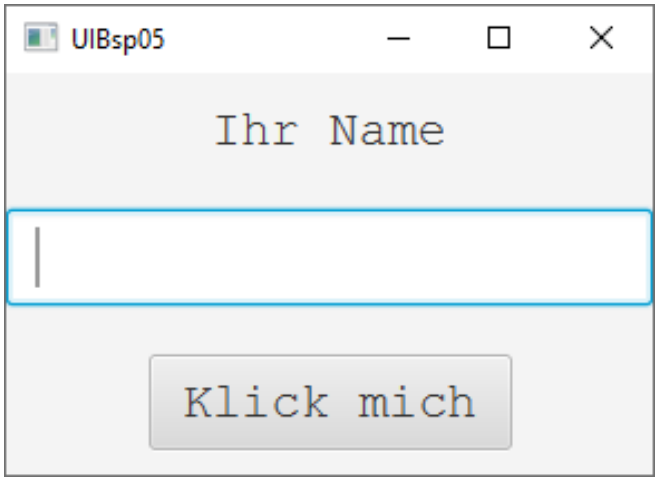
```
label.setBorder(
    new Border(
        new BorderStroke(
            Color.GREEN,
            BorderStrokeStyle.SOLID,
            new CornerRadii(10),
            new BorderWidths(3)
        )
    )
);
```



Dialogfenster in Java: UIBsp05.java

```
VBox root = new VBox(22);
root.setAlignment(Pos.CENTER);
root.setFillWidth(true);

Label label = new Label("Ihr Name") ;
label.setFont(new Font("Courier New",22));
root.getChildren().add(label);

TextField textfield = new TextField("") ;
textfield.setFont(new Font("Courier New",22));
root.getChildren().add(textfield);

Button bn = new Button("Klick mich") ;
bn.setFont(new Font("Courier New",22));
root.getChildren().add(bn);
```

Dialogfenster in Java: UIBsp05.java



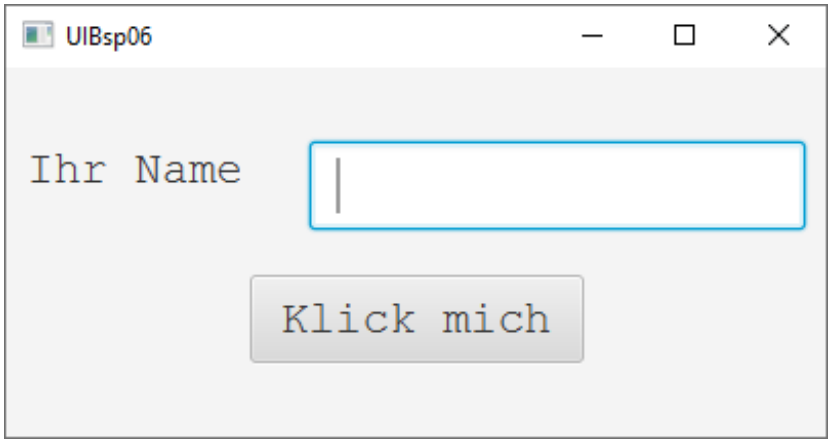
Dialogfenster in Java: UIBsp06.java

```
public void start(Stage stage) {
      VBox root = new VBox(22);
       HBox hbox = new HBox(22);
       root.setFillWidth(true);
       hbox.setMaxWidth(Double.POSITIVE_INFINITY);
       root.getChildren().add(hbox);
       Button bn = new Button("Klick mich") ;
       bn.setFont(new Font("Courier New",22));
       root.getChildren().add(bn);

       Label label = new Label("Ihr Name") ;
       label.setFont(new Font("Courier New",22));
       hbox.getChildren().add(label);
       hbox.setMargin(label, new Insets(0, 0, 0, 10) );  // TRBL
```

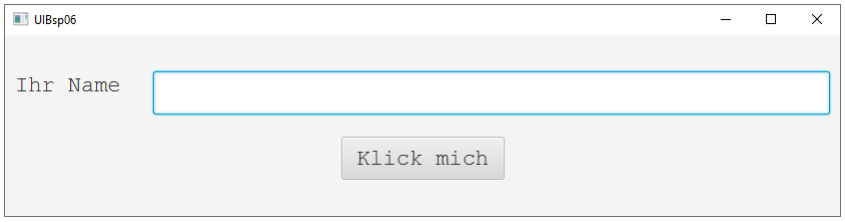
Dialogfenster in Java: UIBsp06.java

```
TextField textfield = new TextField("") ;
textfield.setFont(new Font("Courier New",22));
textfield.setMaxWidth(Double.POSITIVE_INFINITY);
hbox.getChildren().add(textfield);
hbox.setHgrow(textfield, Priority.ALWAYS);
hbox.setMargin(textfield, new Insets(0, 10, 0, 10) );  // TRBL

Scene scene= new Scene(root, 400, 180);
stage.setTitle("UIBsp05");
stage.setScene(scene);
stage.show();
```


Dialogfenster in Java: UIBsp06.java



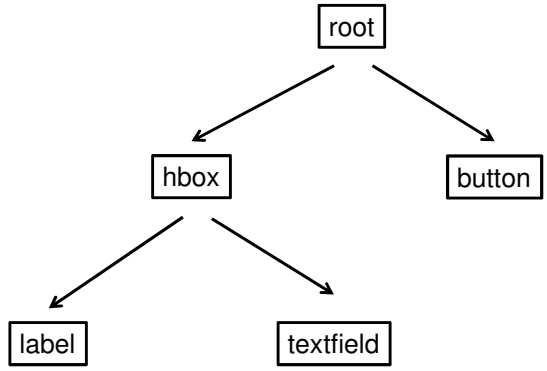
Dialogfenster in Java: UIBsp06.java



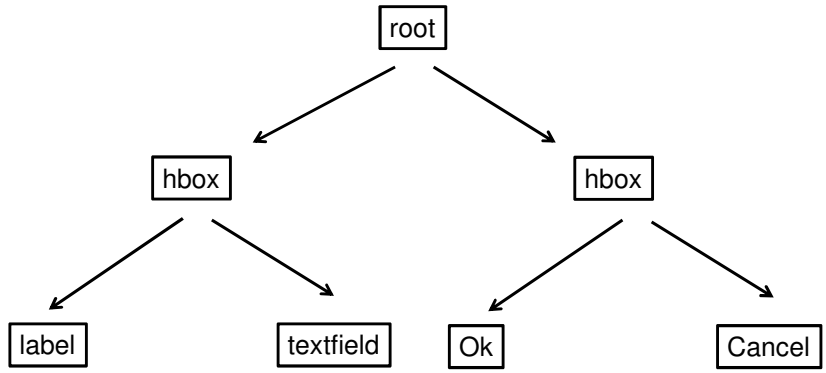
Dialogfenster in Java: UIBsp06B.java

import javafx.geometry.Pos;
hbox.setAlignment(Pos.BASELINE_LEFT);

Dialogfenster in Java: UIBsp06.java



Dialogfenster in Java: UIBsp07.java



Dialogfenster in Java: UIBsp07.java

Bildschirmgestaltung: Layout / Pane

- Die Bildschirmstruktur wird durch ein Containerobjekt oder Layoutverwalter realisiert.
- Darstellung ist unabhängig von der Größe des Fensters
- Layout stellt eine Beziehung zwischen den Elementen dar.

Layout-Verwalter:

- VBox
- HBox
- BorderPane
- GridPane
- FlowPane
- AchorPane fester Abstand zu den Ränder
- TilePane
- TextFlow
- StackPane (CardLayout)

Layout:FlowPane

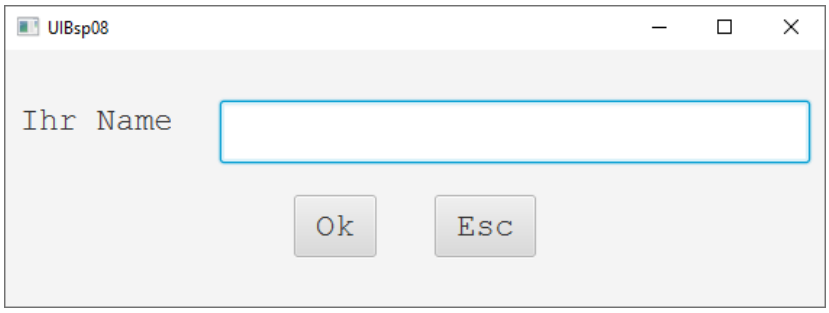
Konstruktor:

- new FlowPane()
- new FlowPane(double hgap, double vgap)
- new FlowPane(Orientation orientation)

Methoden:

- void setAlignment(Pos value)
- void setColumnHalignment(HPos value)
- void setHgap(double value)
- void setOrientation(Orientation value)
- void setPrefWrapLength(double value)
- void setRowValignment(VPos value)
- void setVgap(double value)
- void setMargin(Node child, Insets value)

Dialogfenster in Java: UIBsp08.java



Dialogfenster in Java: UIBsp08.java

```
FlowPane boxpane = new FlowPane(20,20);
boxpane.setAlignment(Pos.CENTER);
boxpane.setMaxWidth(Double.POSITIVE_INFINITY);
root.getChildren().add(boxpane);

Button bnOk = new Button("Ok") ;
bnOk.setFont(new Font("Courier New",22));
bnOk.setMaxWidth(Double.POSITIVE_INFINITY);
boxpane.getChildren().add(bnOk);
boxpane.setMargin(bnOk, new Insets(0, 10, 0, 10) );  // TRBL

Button bnEsc = new Button("Esc") ;
bnEsc.setFont(new Font("Courier New",22));
bnEsc.setMaxWidth(Double.POSITIVE_INFINITY);
boxpane.getChildren().add(bnEsc);
boxpane.setMargin(bnEsc, new Insets(0, 10, 0, 10) );  // TRBL
```


Ereignisse:

Ereignisse durch ein GUI-Element werden durch folgende Aktionen ausgelöst.

- Ein GUI-Element angeklickt
- Der Inhalt einer Textzeile wird verändert
- Die Bezeichnung eines JLabels wird verändert
- Mausklick auf einen Schalter
- Auswahl eines Menüs
- Auswahl eines Popupmenüs
- Maus über einem Element
- Maustaste wurde gedrückt
- Maustaste wurde losgelassen
- Taste wurde gedrückt
- Taste wurde losgelassen
- etc

Ereignisse:

Beispiel Schalter:

- Schalter hat für jede „Aktion“ eine Liste mit möglichen Empfängern.
- Um ein bestimmtes Ereignis zu erhalten, muss man sich registrieren lassen (ActionListener).
- Der Schalter ruft nun durch die Definition des Listener die interne Funktion auf.
- Diese internen Funktionen werden nacheinander aufgerufen. Verkettete Liste.

1. Beispiel Schalter: UIBspAction1: 2 Klassen / Dateien

```
@Override
    public void start(Stage stage) {
        VBox root = new VBox(22);

        Button bn = new Button("Klick mich") ;
        bn.setFont(new Font(22));
        root.getChildren().add(bn);
        bn.setOnAction(new MyEventHandler () );

        Scene scene= new Scene(root, 250, 150);
        stage.setTitle("UIBsp05");
        stage.setScene(scene);
        stage.show();
    }
```

```
public class MyEventHandler implements EventHandler {
    //@ Override
    public void handle(Event event) {
        System.out.println("Der Schalter wurde angeklickt");
    }
}
```

Beispiel Schalter: UIBspAction2: innere Klassen

```
public class UIBspAction2 extends Application {
  Button bn=null;
   @Override
   public void start(Stage stage) {
        VBox root = new VBox(22);
        bn = new Button("Klick mich") ;
        root.getChildren().add(bn);
        bn.setOnAction(new MyEventHandler () );

        Scene scene= new Scene(root, 250, 150);
        stage.setTitle("UIBsp05");
        stage.setScene(scene);
        stage.show();
   }
}
class MyEventHandler implements EventHandler {
   public void handle(Event event) {
      System.out.println("Der Schalter wurde angeklickt");
      bn.setText("hallo");
    }
}
```


Beispiel Schalter: UIBspAction3: anonyme Klasse

```
   @Override
   public void start(Stage stage) {
        VBox root = new VBox(22);
        Button bn = new Button("Klick mich") ;
        root.getChildren().add(bn);
        bn.setOnAction(  new EventHandler<ActionEvent>(){
          @Override
          public void handle(ActionEvent event) {
             label.setText("Der Schalter wurde angeklickt");
          }
        });
        );

        Scene scene= new Scene(root, 250, 150);
        stage.setTitle("UIBsp05");
        stage.setScene(scene);
        stage.show();
   }
}
```

Beispiel Schalter: UIBspAction4: Lambda-Ausdrücke

```
    @Override
    public void start(Stage stage) {
        VBox root = new VBox(22);
        Button bn = new Button("Klick mich") ;
        root.getChildren().add(bn);

        bn.setOnAction(e -> label.setText("Der Schalter wurde angeklickt"));

        Scene scene= new Scene(root, 250, 150);
        stage.setTitle("UIBsp05");
        stage.setScene(scene);
        stage.show();
    }
}
```


Beispiel Schalter: UIBspAction5: Lambda-Ausdrücke

```
    Label label = null;

    @Override
    public void start(Stage stage) {
        VBox root = new VBox(22);
        Button bn = new Button("Klick mich") ;
        root.getChildren().add(bn);
        bn.setOnAction( e -> bnClick() );
        Scene scene= new Scene(root, 250, 150);
        stage.setTitle("UIBsp05");
        stage.setScene(scene);
        stage.show();
    }

    private void bnClick() {
      label.setText("Der Schalter wurde angeklickt");
    }
}
```

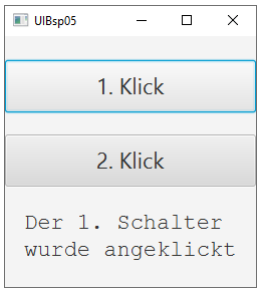
Beispiel Schalter: UIBspAction5: Lambda-Ausdrücke

```
    Label label = null;

    @Override
    public void start(Stage stage) {
          VBox root = new VBox(22);
          Button bn = new Button("Klick mich") ;
          root.getChildren().add(bn);
          bn.setOnAction( a -> bnClick() );
          Scene scene= new Scene(root, 250, 150);
          stage.setTitle("UIBsp05");
          stage.setScene(scene);
          stage.show();
    }

    private void bnClick() {
      label.setText("Der Schalter wurde angeklickt");
    }
}
```


Beispiel Schalter: UIBspAction6: interface

```
public class UIBspAction4 extends Application implements EventHandler{

    @Override
    public void start(Stage stage) {
          VBox root = new VBox(22);
          root.setAlignment(Pos.CENTER);
          root.setFillWidth(true);

          bn1 = new Button("1. Klick") ;
          bn1.setOnAction(this);
          bn2 = new Button("2. Klick") ;
          bn2.setOnAction(this);
          root.getChildren().addAll(bn1, bn2);

          label = …
          Scene scene= new Scene(root, 250, 250);
          stage.setTitle("UIBspAction6");
          stage.setScene(scene);
          stage.show();
    }
```

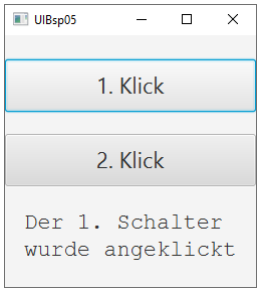
```
    @Override
    public void handle(Event e) {
      if(e.getSource() == bn1){
        label.setText("1. Schalter");
      }
      if(e.getSource() == bn2){
        label.setText("2. Schalter");
      }
    }
```



Beispiel Schalter: UIBspAction7: interface (getId,setId)

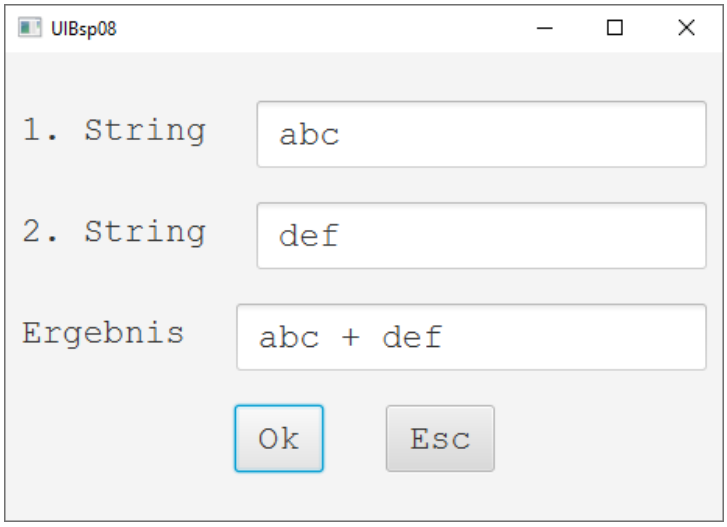
```
public class UIBspAction4 extends Application implements EventHandler{
    private  final String BN1="BN1";
    private  final String BN2="BN2";
    @Override
    public void start(Stage stage) {
          VBox root = new VBox(22);
          root.setAlignment(Pos.CENTER);
          root.setFillWidth(true);

          Button bn1 = new Button("1. Klick") ;
          bn1.setOnAction(this);
          bn1.setId(BN1);
          Button  bn2 = new Button("2. Klick") ;
          bn2.setOnAction(this);
          bn1.setId(BN2);
          root.getChildren().addAll(bn1, bn2);

          label = …
          Scene scene= new Scene(root, 250, 250);
          stage.setTitle("UIBspAction7");
          stage.setScene(scene);
          stage.show();
    }
```

```
    @Override
    public void handle(Event e) {
      Control ctrl=(Control)e.getSource();
       String id = control.getId();
       if(id.equals(BN1)){
          label.setText("1. Schalter");
       }
        if(id.equals(BN1)){
            label.setText("2. Schalter");
        }
    }
```

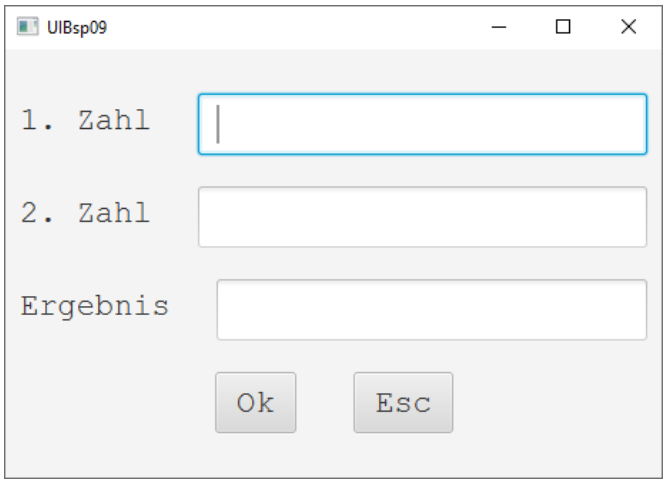


Dialogfenster in Java: UIBsp07.java

Dialogfenster in Java: UIBsp08.java



Dialogfenster in Java: UIBsp09.java



Dialogfenster in Java: UIBsp09.java

```
import javafx.application.Platform;

import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.event.Event;


public class UIBsp09 extends Application  implements EventHandler{

    private TextField textfield1 = new TextField("") ;
    private TextField textfield2 = new TextField("") ;
    private TextField textfield3 = new TextField("") ;

    private  Button bnOk = new Button("Ok") ;
    private  Button bnEsc = new Button("Esc") ;
```


Dialogfenster in Java: UIBsp09.java

```
private HBox createLine(TextField textfield, String caption) {
        HBox hbox = new HBox(22);
        hbox.setFillHeight(true);
        hbox.setMaxWidth(Double.POSITIVE_INFINITY);

        Label label = new Label(caption) ;
        label.setFont(new Font("Courier New",22));
        hbox.getChildren().add(label);
        hbox.setMargin(label, new Insets(5, 0, 0, 10) );  // TRBL


        textfield.setFont(new Font("Courier New",22));
        textfield.setMaxWidth(Double.POSITIVE_INFINITY);
        hbox.getChildren().add(textfield);
        hbox.setHgrow(textfield, Priority.ALWAYS);
        hbox.setMargin(textfield, new Insets(0, 10, 0, 10) );  // TRBL
        return hbox;
    }
```

Dialogfenster in Java: UIBsp09.java

```
public void start(Stage stage) {
        VBox root = new VBox(22);
        root.setAlignment(Pos.CENTER);
        root.setFillWidth(true);


        HBox hbox = createLine(textfield1, "1. Zahl");
        root.getChildren().add(hbox);

        hbox = createLine(textfield2, "2. Zahl");
        root.getChildren().add(hbox);

        hbox = createLine(textfield3, "Ergebnis");
        root.getChildren().add(hbox);



                hbox.setAlignment(Pos.BASELINE_LEFT);
```


Dialogfenster in Java: UIBsp09.java

```
        FlowPane boxpane = new FlowPane(20,20);
        boxpane.setAlignment(Pos.CENTER);
        boxpane.setMaxWidth(Double.POSITIVE_INFINITY);
        root.getChildren().add(boxpane);

        bnOk.setFont(new Font("Courier New",22));
        bnOk.setMaxWidth(Double.POSITIVE_INFINITY);
        bnOk.setOnAction(this);
        boxpane.getChildren().add(bnOk);

        bnEsc.setFont(new Font("Courier New",22));
        bnEsc.setMaxWidth(Double.POSITIVE_INFINITY);
        bnEsc.setOnAction(this);
        boxpane.getChildren().add(bnEsc);

        Scene scene= new Scene(root, 460, 300);
        stage.setTitle("UIBsp09");
        stage.setScene(scene);
        stage.show();
  }
```

Dialogfenster in Java: UIBsp09.java

```
@Override
    public void handle(Event e) {
        if(e.getSource() == bnOk){
          calc();
        }
        if(e.getSource() == bnEsc){
          Platform.exit();
        }
    }
```


Dialogfenster in Java: UIBsp09.java

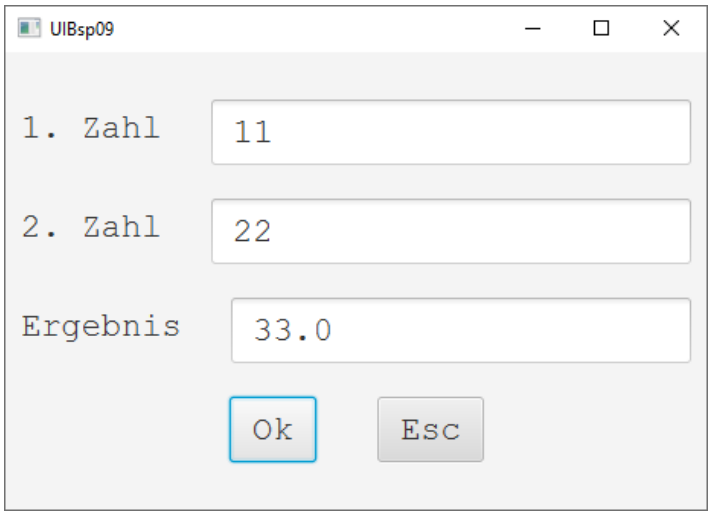
```
double getDoubleNumber(String str_zahl) {
      try {
            double zahl = Double.parseDouble(str_zahl);
            return zahl;
      }
      catch (NumberFormatException e) {
            return Double.NaN;
      }  // try
}  // getDoubleNumber


public void errorBox( String message, String title) {
      JOptionPane.showConfirmDialog(null,message,title,
        JOptionPane.DEFAULT_OPTION,
        JOptionPane.ERROR_MESSAGE);  // Symbol
 } // ErrorBox
```

import javax.swing.JOptionPane; https://code.makery.ch/blog/javafx-2-dialogs/

Dialogfenster in Java: UIBsp09.java

```
private void calc() {
      double x = getDoubleNumber(textfield1.getText());
      if (Double.isNaN(x)) {
        errorBox("Die 1. Eingabe ist keine Zahl","Hinweis");
        return;
      }

      double y = getDoubleNumber(textfield2.getText());
      if (Double.isNaN(y)) {
        errorBox("Die 2. Eingabe ist keine Zahl","Hinweis");
        return;
      }
      double erg = x+y;
      textfield3.setText(Double.toString(erg));

    }
```


Dialogfenster in Java: UIBsp09.java

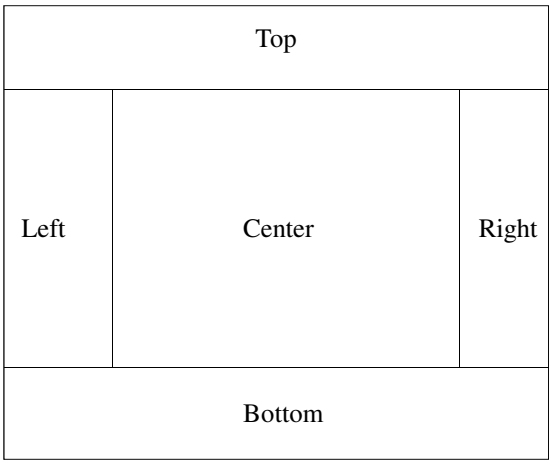


Dialogfenster in Java: UIBsp10.java

```
        FlowPane boxpane = new FlowPane(20,20);
        boxpane.setAlignment(Pos.CENTER);
        boxpane.setMaxWidth(Double.POSITIVE_INFINITY);
        root.getChildren().add(boxpane);

        bnOk.setFont(new Font("Courier New",22));
        bnOk.setMaxWidth(Double.POSITIVE_INFINITY);
        bnOk.setOnAction( e -> calc() );
        boxpane.getChildren().add(bnOk);

        bnEsc.setFont(new Font("Courier New",22));
        bnEsc.setMaxWidth(Double.POSITIVE_INFINITY);
        bnEsc.setOnAction( e ->  Platform.exit() );
        boxpane.getChildren().add(bnEsc);

        Scene scene= new Scene(root, 460, 300);
        stage.setTitle("UIBsp09");
        stage.setScene(scene);
        stage.show();
    }
```

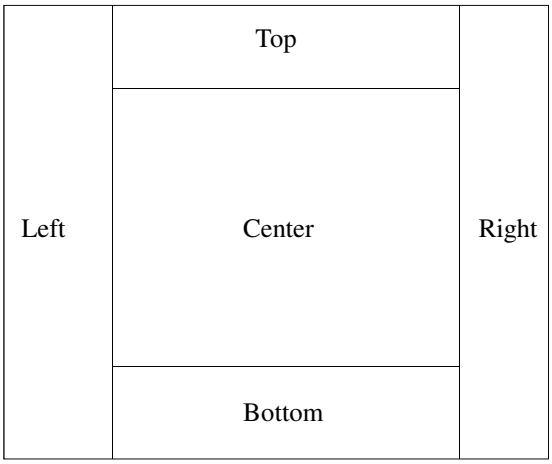

Border-Pane

```
border = new BorderPane();
border.setTop(…);
border.setLeft(…);
border.setCenter(…);
border.setRight(…);
border.setBottom(…);
```



Anchor-Pane

- AnchorPane.setBottomAnchor(Node child, Double value)
- AnchorPane.setLeftAnchor(Node child, Double value)
- AnchorPane.setRightAnchor(Node child, Double value)
- AnchorPane.setTopAnchor(Node child, Double value)



Border-Pane

- BorderPane root = new BorderPane();
- Button bnTop = new Button("Top");
- root.setTop(bnTop);

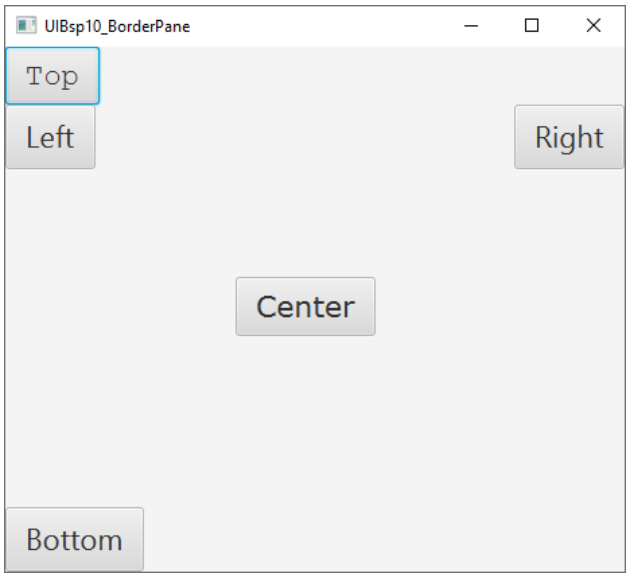
- Button bnLeft = new Button("Left");
- root.setLeft(bnLeft);

- Button bnCenter = new Button("Center");
- root.setCenter(bnCenter);

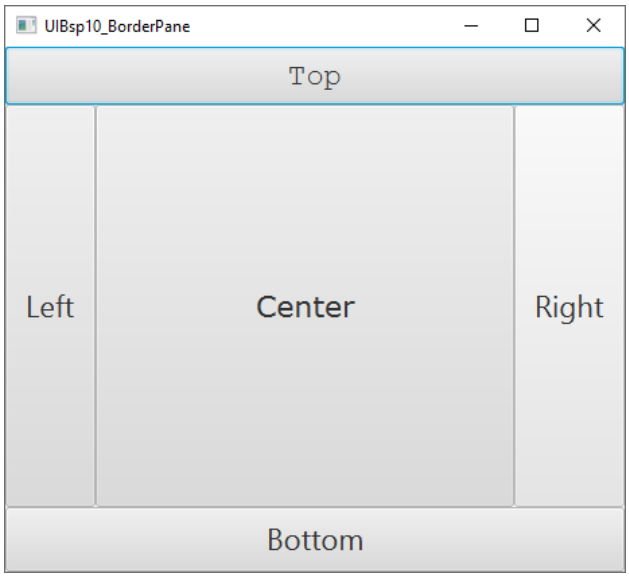
- Button bnRight = new Button("Right");
- root.setRight(bnRight);

- Button bnBottom = new Button("Bottom");
- root.setBottom(bnBottom);

Border-Pane



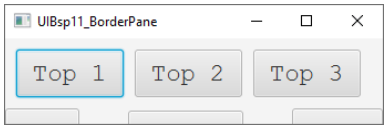
Border-Pane



Border-Pane

```
bnTop.setMaxWidth(Double.POSITIVE_INFINITY);
bnLeft.setMaxHeight(Double.POSITIVE_INFINITY);

bnCenter.setMaxWidth(Double.POSITIVE_INFINITY);
bnCenter.setMaxHeight(Double.POSITIVE_INFINITY);

bnRight.setMaxHeight(Double.POSITIVE_INFINITY);
bnBottom.setMaxWidth(Double.POSITIVE_INFINITY);
```


Border-Pane:




```
FlowPane panel = new FlowPane();
Button bn1 = new Button("Top 1");
Button bn2 = new Button("Top 2");
Button bn3 = new Button("Top 3");
panel.getChildren().addAll(bn1, bn2, bn3);

bn1.setFont(new Font("Courier New",22));
panel.setMargin(bn1, new Insets(10, 10, 10, 10) );  // TRBL
bn1.setMaxWidth(Double.POSITIVE_INFINITY);

root.setTop(panel);
```

Java FX und Menüs

Klassen:

- MenuBar
- Menu

- MenuItem
- RadioMenuItem
- CheckMenuItem

Menü

```
private  MenuItem menuOpen = new MenuItem("Open File...");
private  MenuItem menuSave = new MenuItem("Save File...");
private  MenuItem menuClose = new MenuItem("Close");

public void start(Stage stage) {
        this.stage = stage;
        BorderPane root = new BorderPane();
        root.setTop(  setTopElements()  );

        Scene scene= new Scene(root, 460, 390);
        stage.setTitle("UIBspMenu01");
        stage.setScene(scene);
        stage.show();
  }
```

setTopElements

```
private Pane  setTopElements() {
        VBox vbox = new VBox(22);

        MenuBar menuBar = new MenuBar();
        Menu menuFile = new Menu("File");
        menuBar.getMenus().add(menuFile);

        menuFile.getItems().addAll(menuOpen, menuSave,
            new SeparatorMenuItem());
        menuFile.getItems().add(menuClose);

        vbox.getChildren().add(menuBar);
        return vbox ;
    }
```


MenuItem: Weitere Methoden

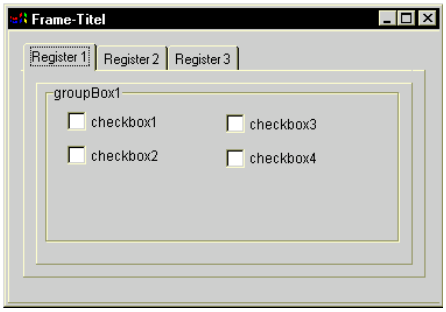
```
menuOpen.setOnAction(this);
menuSave.setOnAction(this);
menuClose.setOnAction(this);

menuOpen.setAccelerator(KeyCombination.keyCombination("Ctrl+O"));
menuSave.setAccelerator(KeyCombination.keyCombination("Ctrl+S"));

menuCut.setAccelerator(KeyCombination.keyCombination("Ctrl+X"));
menuCopy.setAccelerator(KeyCombination.keyCombination("Ctrl+C"));
menuPaste.setAccelerator(KeyCombination.keyCombination("Ctrl+V"));
```

Menu: CSS

```
scene.getStylesheets().add("Menu.css");

.menu, .menu-item {
 -fx-text-fill: blue ;
 -fx-font-size:22;
}

.menu-item .label{
    -fx-text-fill: greenyellow;
}

*.popup-menu {
    -fx-background-color:
        linear (0%,0%) to (0%,100%) stops (0%,#383838) (100%,#141414);
}
```


TabPane

Das TabPane platziert Komponenten (gewöhnlich Bedienfelder) wie Karten in einem Stapel nebeneinander. Es wird jeweils nur eine Komponente bzw. ein Bedienfeld angezeigt. Man hat die Möglichkeit, jeweils ein anderes Bedienfeld nach vorne zu legen und somit durch die Register anzuzeigen.



TabPane: Struktur

```
public void start(Stage stage) {
        this.stage = stage;
        BorderPane root = new BorderPane();
        TabPane tabpane = new TabPane();
        root.setCenter( tabpane  );

        Scene scene= new Scene(root, 660, 490);
        stage.setTitle(" UIBTabPane01");
        stage.setScene(scene);
        stage.show();
   }
```

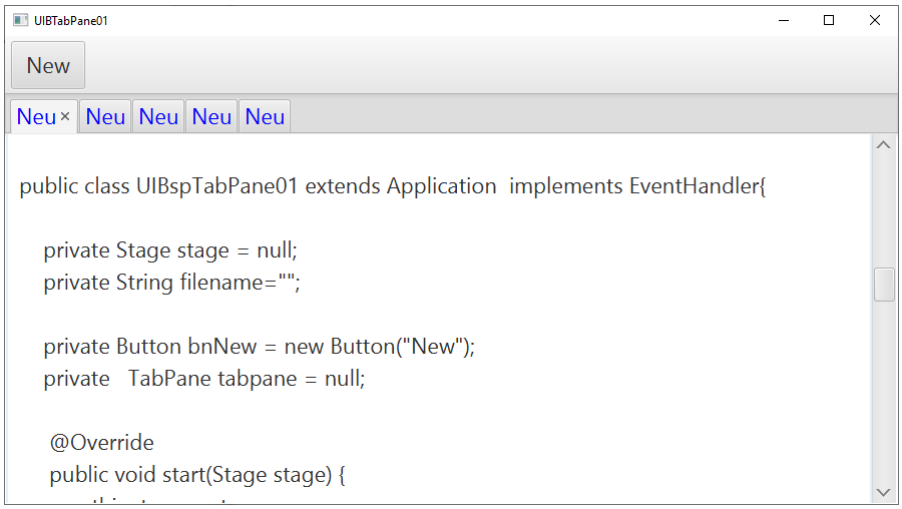

TabPane: Struktur

```
public void start(Stage stage) {
        this.stage = stage;
        BorderPane root = new BorderPane();
        TabPane tabpane = new TabPane();
        root.setCenter( tabpane  );

        Scene scene= new Scene(root, 660, 490);
        stage.setTitle(" UIBTabPane01");
        stage.setScene(scene);
        stage.show();
   }
```

TabPane: Einfügen

```
TextArea editor = new TextArea();

editor.setFont(new Font(22));

Tab tab = new Tab("Neu");

tab.setContent(editor);

tabpane.getTabs().add(tab);
```

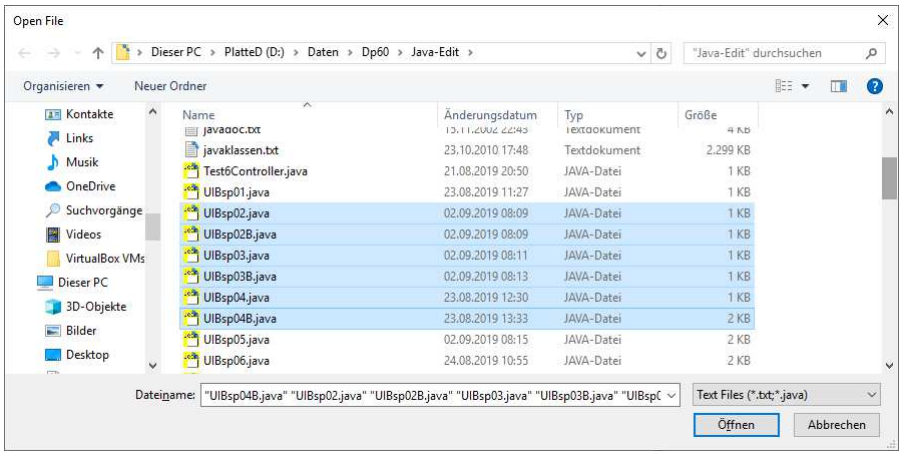

TabPane



TabPane: Zugriff

```
private Tab getActualTab() {
      // javafx.collections.*;
   ObservableList<Tab> tabs = tabpane.getTabs();

   if (tabs.size()>0) {
      for (Tab tab : tabs) {
         if (tab.isSelected()) {
            return tab;
         }
      }
      return null;
   }
   else {
      return null;
   }
}
```


Opendialog: FileChooser



Opendialog: FileChooser

import javafx.stage.FileChooser;
import javafx.stage.FileChooser.ExtensionFilter;
import java.util.List;

- void setInitialDirectory(File value)
- void setTitle(String value)

- File showOpenDialog(Window ownerWindow)
- List<File> showOpenMultipleDialog(Window ownerWindow)
- File showSaveDialog(Window ownerWindow)

FileChooser: Opendialog:

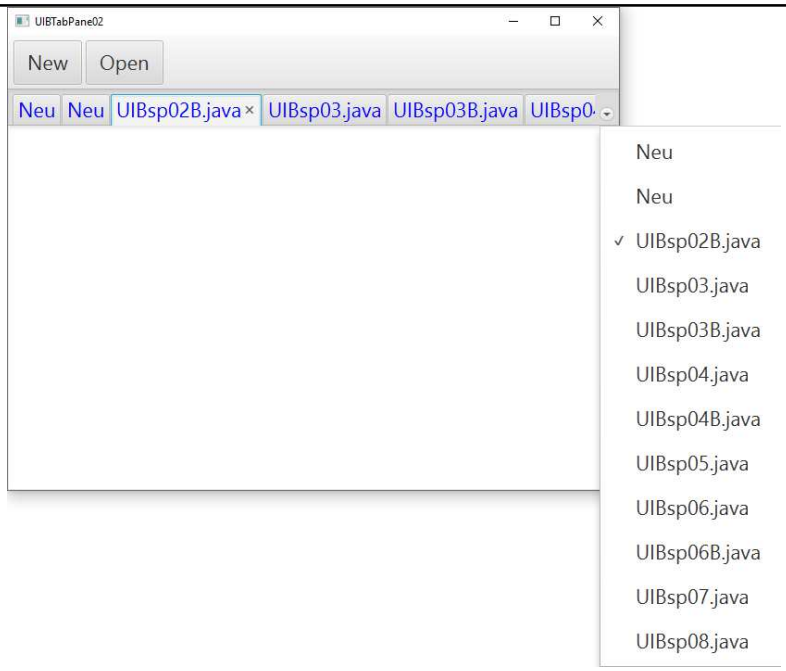
```
FileChooser fileChooser = new FileChooser();
fileChooser.setTitle("Open File");

fileChooser.getExtensionFilters().addAll(
   new ExtensionFilter("Text Files", "*.txt"),
    new ExtensionFilter("Java Files", "*.java", "*.css"),
   new ExtensionFilter("All Files", "*.*"));

fileChooser.setInitialDirectory(
          new File(System.getProperty("user.home")));

File selectedFile = fileChooser.showOpenDialog(this.stage);

if (selectedFile != null) {
      this.filename=selectedFile.getPath();
     readFile();
}
```

FileChooser: Opendialog:

```
FileChooser fileChooser = new FileChooser();

fileChooser.setTitle("Open File");

fileChooser.getExtensionFilters().addAll(
   new ExtensionFilter("Text Files", "*.txt""),
   new ExtensionFilter("Java Files", "*.java", "*.css"),
   new ExtensionFilter("All Files", "*.*"));

fileChooser.setInitialDirectory(
      new File(System.getProperty("user.home")));

List<File> list = fileChooser.showOpenMultipleDialog(stage);

if (list != null) {
  for (File file : list) {
    insertTab(file.getPath());
  }
}
```


FileChooser: Savedialog:

```
FileChooser fileChooser = new FileChooser();

fileChooser.setTitle("Save File");

fileChooser.getExtensionFilters().addAll(
 new ExtensionFilter("Text Files", "*.txt""),
   new ExtensionFilter("Java Files", "*.java", "*.css"),
   new ExtensionFilter("All Files", "*.*"));

fileChooser.setInitialDirectory(
      new File(System.getProperty("user.home")));

File selectedFile = fileChooser.showSaveDialog(this.stage);

if (selectedFile != null) {
    String filename=selectedFile.getPath();
    writeFile(filename);
}
```



Benutzeroberflächen: Zeichnen in Java FX

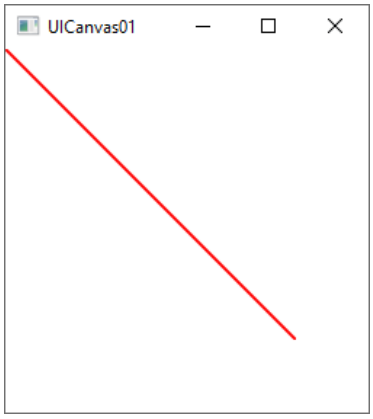
- Linien (Farbe, Strichdicke, Strichart)
- Zeichenblatt (Canvas)
- Zeichenmethoden

Benutzeroberflächen: Zeichnen in Java

- **fillOval**(double x, double y, double w, double h)
- **fillPolygon**(double[] xPoints, double[] yPoints, int nPoints)
- **fillRect**(double x, double y, double w, double h)
- **fillRoundRect**(double x, double y, double w, double h, double arcWidth, double arcHeight)
- **fillText**(String text, double x, double y)
- **fillText**(String text, double x, double y, double maxWidth)
- **lineTo**(double x1, double y1)
- **moveTo**(double x0, double y0)
- **rect**(double x, double y, double w, double h)
- **strokeLine**(double x1, double y1, double x2, double y2)
- **strokeOval**(double x, double y, double w, double h)
- **strokePolygon**(double[] xPoints, double[] yPoints, int nPoints)
- **strokePolyline**(double[] xPoints, double[] yPoints, int nPoints)
- **strokeRect**(double x, double y, double w, double h)
- **strokeRoundRect**(double x, double y, double w, double h, double arcWidth, double arcHeight)

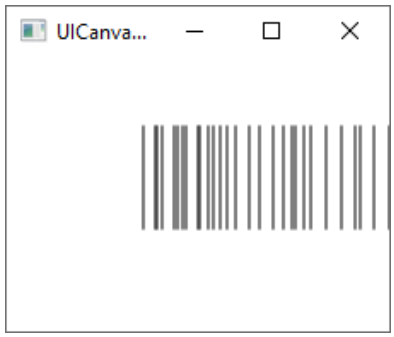
Zeichnen in Java: Linien

```
Canvas canvas = new Canvas(MAX, MAX);
root.setTop(  canvas  );

GraphicsContext gc =
canvas.getGraphicsContext2D();

gc.beginPath();
gc.setLineWidth(2);
gc.setStroke(Color.RED);  // Linienfarbe
gc.strokeLine(0,0, MAX, MAX);

gc.stroke();
```



Die Variable gc bezeichnet man auch als Grafikkontext

Zeichnen in Java: Linien

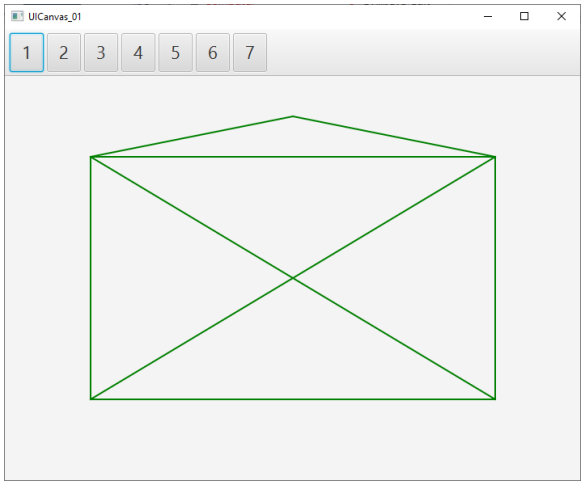


import javafx.scene.paint.*;
import javafx.scene.canvas.*;

Linien zeichnen: UIBspCanvas02

```
Canvas canvas = new Canvas(MAX, MAX);
root.setTop(  canvas  );

GraphicsContext gc = canvas.getGraphicsContext2D();
gc.beginPath();
int x = 80;
for (int i=0; i<60; i++) {
    gc.strokeLine(x,40,x,100);
    x += 1+ 10*Math.random();
}
gc.stroke();  // closePath()
```


Zeichnen in Java: Linien (CanvasBsp04)



Linien zeichnen

Links /oben : 100,100
Rechts / unten: 600,400

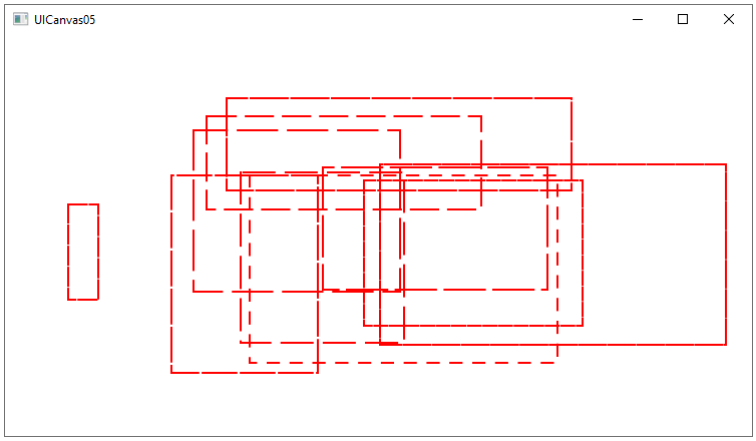
mitte / oben: 350, 50

- Vorlesungsaufgabe

Vorlesungsaufgabe:

```
public void paint (Graphics g)
 {
   g.drawLine(100,100, 600,100);  // waagerecht oben
   g.drawLine(600,100, 600,400);  // senkrecht rechts
   g.drawLine(100,400, 600,400);  // waagerecht unten
   g.drawLine(100,100, 100,400);  // senkrecht links

   g.drawLine(100,100, 600,400);  // links oben nach rechts unten
   g.drawLine(100,400, 600,100);  // links unten nach rechts oben

   g.drawLine(100,100, 350, 50);  // links oben nach rechts unten
   g.drawLine(350, 50, 600,100);  // links unten nach rechts oben
 }
```

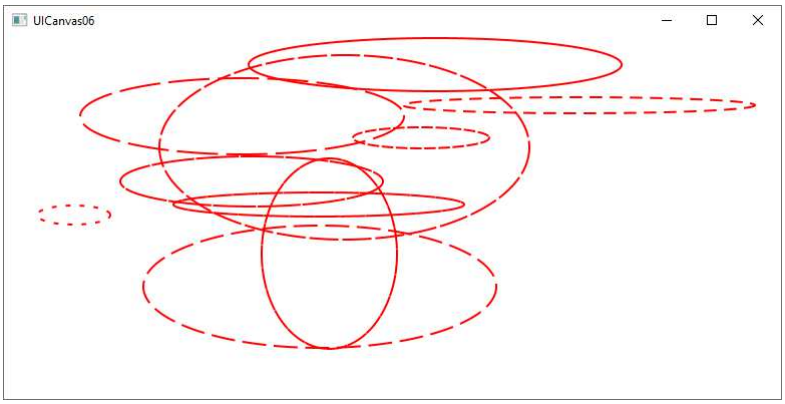

Zeichnen in Java: Rechtecke (CanvasBsp05)



Zeichnen in Java: Rechteck

```
int x, y;  // Mittelpunkt
int w, h;  // Breite Höhe

for (int i=0; i<10; i++) {
   gc.setLineDashes(2+40*Math.random(),
2+10*Math.random());

    x = (int) (  400*Math.random() );  // Links Oben
    y = (int) (  200*Math.random() );   // Links Oben
    w = (int) (  400*Math.random() ); // Width
     h = (int) (  200*Math.random() ); // Height
     gc.strokeRect(x,y,w,h);
}
```

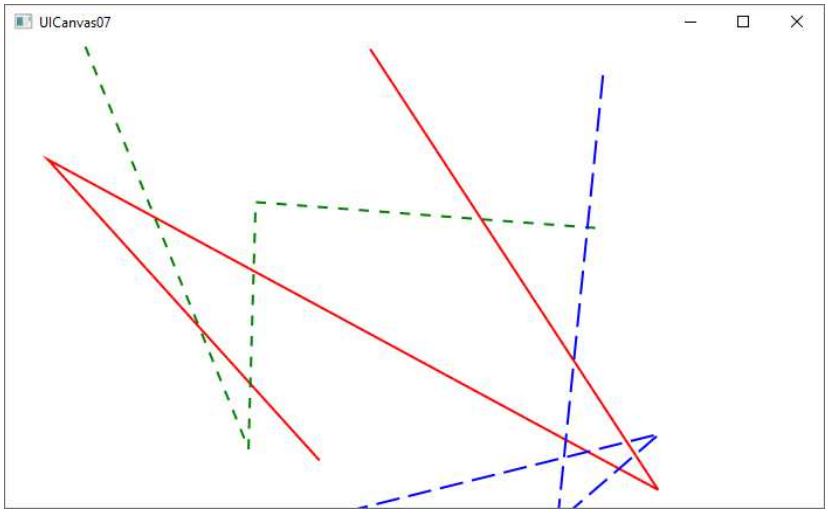

Zeichnen in Java: Ellipsen



Zeichnen in Java: Ellipsen

```
int x, y;  // Mittelpunkt
int w, h;  // Breite Höhe

for (int i=0; i<10; i++) {
  gc.setLineDashes(2+40*Math.random(), 2+10*Math.random());

  x = (int) (  400*Math.random() );     // Links Oben
  y = (int) (  200*Math.random() );      // Links Oben
  w = (int) (  400*Math.random() );     // Width
  h = (int) (  200*Math.random() );     // Height
  gc.strokeOval(x,y,w,h);
}

gc.stroke();
```


Zeichnen in Java: PolyLine



Zeichnen in Java: PolyLine

```
gc.beginPath();
        gc.setLineWidth(2);
        gc.setStroke(Color.RED);
        double[] xPoints=new double[10];
        double[] yPoints=new double[10];
         //void      strokePolyline(double[] xPoints, double[] yPoints, int nPoints)

        gc.setLineDashes(2+40*Math.random(), 2+10*Math.random());
        for (int j=0; j<6; j++) {
          xPoints[j] = 2+600*Math.random();
          yPoints[j] = 5+470*Math.random();
        }

        gc.strokePolyline(xPoints,yPoints,4);
```


Die Java Klassenbibliothek

- Beispiele
  - java.lang: Sprachunterstützung (Threads, String)
  - java.util: Hilfeklassen, allgemeine Datenstrukturen (Stack, Vector)
  - java.io: Ein/Ausgabe-Klassen (Dateien, Streams)
  - java.nio: Ein/Ausgabe-Klassen (Neue Variante)
  - java.nio2: Ein/Ausgabe-Klassen (JDK 7)
  - java.net: Sockets, URL
  - java.applets, java.awt: Java Windows Toolkit
- Erweiterungen (Auswahl)
  - java.swing: AWT mit anpassbaren „Look & Feel“)
  - speech: Sprachein- und Sprachausgabe
  - JDBC: Datenbankanbindung
  - Beans: wiederverwendbare Komponenten
  - JMF: Java Media Foundation

Java: Exception

```
public class exception1 {

  public static void main(String args[]) {
    int j,k;
    j=0;
    k=3 / j;
  }  // main
}
```


Java: Exception

```
D:\HS-Harz\GUI>java exception1
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at exception1.main(exception1.java:8)

D:\HS-Harz\GUI>
```

Java: Exception

Ursachen für eine Exception:

•Fehlerhafte Eingabe (Numerische Eingabe, URL)
•Gerätfehler (Drucker, Webseite, Diskette)
•Physikalische Grenzen (mangelnder Speicher, Freier Speicherplatz)
•Programmierfehler (Arrayindex, Stackfehler, Overflow, Underflow)

Exception:

Bei einer Exception wird
- die aktuelle Prozedur sofort verlassen
- es wird ein Exceptionobjekt erzeugt
- es wird kein Returncode erzeugt
- der Aufrufcode wird nicht weiterabgearbeitet
- es beginnt die Suche nach einem „exception handler“, der für diese Exception zuständig ist

Java: Exception-Arten

Standard Runtime Exceptions:

ArithmeticException:	An exceptional arithmetic situation has arisen, such as an integer division (§ 15.16.2) operation with a zero divisor.
ArrayStoreException:	An attempt has been made to store into an array component a value whose class is not assignment compatible with the component type of the array (§ 10.10, § 15.25.1).
ClassCastException:	An attempt has been made to cast (§ 5.4, § 15.15) a reference to an object to an inappropriate type.
IllegalArgumentException:	A method was passed an invalid or inappropriate argument or invoked on an inappropriate object. Subclasses of this class include:
IllegalThreadStateException:	A thread was not in an appropriate state for a requested operation.
NumberFormatException:	An attempt was made to convert a String to a value of a numeric type, but the String did not have an appropriate format.
IllegalMonitorStateException:	A thread has attempted to wait on (§ 20.1.6, § 20.1.7, § 20.1.8) or notify (§ 20.1.9, § 20.1.10) other threads waiting on an object that it has not locked.
IndexOutOfBoundsException:	Either an index of some sort (such as to an array, a string, or a vector) or a subrange, specified either by two index values or by an index and a length, was out of range.

Java: Exception

Package java:

java.io.IOException:	A requested I/O operation could not be completed normally. Subclasses of this class include:
java.io.EOFException:	End of file has been encountered before normal completion of an input operation.
java.io.FileNotFoundException:	A file with the name specified by a file name string or path was not found within the file system.
java.io.InterruptedIOException:	The current thread was waiting for completion of an I/O operation, and another thread has interrupted the current thread, using the interrupt method of class Thread (§ 20.20.31).
java.io.UTFDataFormatException:	A requested conversion of a string to or from Java modified UTF-8 format could not be completed (§ 22.1.15, § 22.2.14) because the string was too long or because the purported UTF-8 data was not the result of encoding a Unicode string into UTF-8.

Standard Runtime Exceptions:

NullPointerException:	An attempt was made to use a null reference in a case where an object reference was required.

Java: Exception (Anfangsbeispiel)

```
import java.io.*;

public class exception1 {

  public static void main(String argv[]) {
    int j,k;
    j=0;
    k=3 / j;
  }  // main
}
```

Java: Exception (interne Abfrage)

```
import java.io.*;

public class exception2 {

  public static void main(String argv[]) {
    int j,k;
    j=0;
    try {        // Excecption Block
      k=3 / j;
      syso(k);
    }
    catch (ArithmeticException f) {
        System.err.println(" ArithmeticException : " + f);
    }
  } // main
}
```

Kein Absturz für den Anwender

exception2

Java: finally Klausel

Beim Auslösen einer Exception wird sofort das jeweilige Modul sofort verlassen.

Problem:

- Lokale Resourcen werden nicht sauber entfernt:
- Eine Datei ist geöffnet, wird nicht geschlossen

Abhilfe:

finally Klausel

Java: finally Klausel

```
Resource resource;
try {
  resource = ...;
  resource.methodThrowsException();
  resource.close();
}
catch (Exception e) {
 ...
}
finally {
   resource.close();
}
```


Java: finally Klausel

```
FileInputStream fin=null;
DataInputStream din=null;
byte b;
try {
  fin = new FileInputStream( sFilename );
  din = new DataInputStream(fin);
  while  ( din.available()>0 ) {
    b  = din.readByte();
    System.out.println((char) (b) );
  }
 }
catch ( FileNotFoundException e )  {
  System.err.println( "Die Datei ist nicht vorhanden!" );
}
catch ( IOException e )  {
  System.err.println( "Schreib-/Leseprobleme!" );
}
finally {
  if ( fin != null )
   try { fin.close(); }
   catch ( IOException e ) {
    e.printStackTrace();
   }
}
```

Java: finally Klausel

```
void createGrafik(String sFileName) {
Graphics g = image.getGraphics(sFileName);
try {
        code mit der Möglichkeit einer Exception
}
catch(FileNotFoundException e) {
        done = false;
}
catch(EOFException e) {
        done = true;
}
finally {
        g.dispose();  // Speicher freigeben
}
}
```

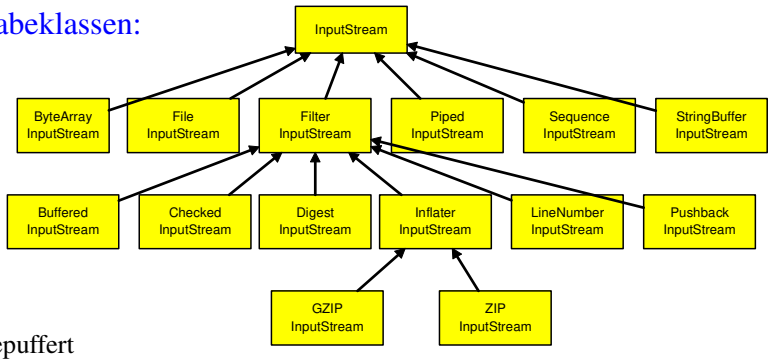

Java: finally Klausel

```
void createGrafik(String sFileName) {
Graphics g = image.getGraphics(sFileName);
try {
        code mit der Möglichkeit einer Exception
}
catch(IOException e) {
        done = false;
}
catch(Exception e) {
        done = true;
}
finally {
        g.dispose();  // Speicher freigeben
}
}
```

Datenverarbeitung in Java

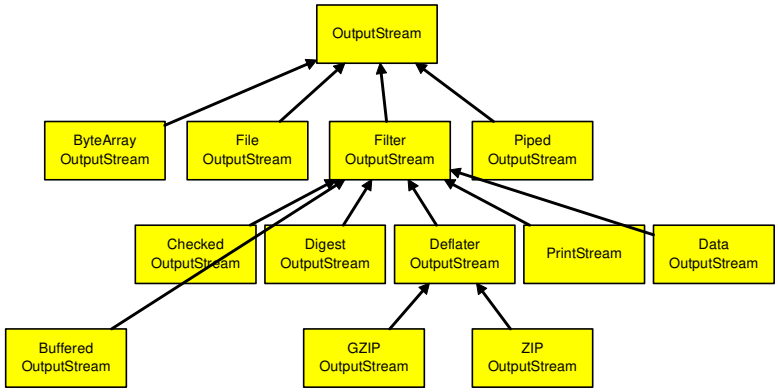
Es steht eine Vielzahl von Klassen/Modulen zur Verfügung (58):

Eingabeklassen:



- gepuffert
- Filter für Dateinamen
- für Zeichen, Zeichenketten, Objekte, Token
- mit Pipe-Verfahren

Ausgabe in eine Datei



- gepuffert
- Filter für Dateinamen
- für Zeichen, Zeichenketten, Objekte, Token
- mit Pipe-Verfahren

Klassen des io-Packages

BufferedInputStream	Zwischenbuffer
BufferedOutputStream	Zwischenbuffer
BufferedReader	Zeilenweise
BufferedWriter	Zwischenbuffer
ByteArrayInputStream	ZwischenArray
ByteArrayOutputStream	ZwischenArray
CharArrayReader	CharacterStream
CharArrayWriter	CharacterStream
Console	à la DOS
DataInputStream	int byte double
DataOutputStream	int byte double
File	abstrakte Klasse
FileDescriptor	arbeitet mit handles
FileInputStream	byte-weise
FileOutputStream	byte-weise
FilePermission	Rechte
FileReader	Charakterbasierend
FileWriter	Charakterbasierend

Klassen des io-Packages

FilterInputStream	Transformierung
FilterOutputStream	Transformierung
FilterReader	Transformierung und Charakterbasierend
FilterWriter	Transformierung und Charakterbasierend
InputStream	Superclass
InputStreamReader	Charakterbasierend
LineNumberInputStream	Zeilenweise
LineNumberReader	Zeilenweise
ObjectInputStream	Serialize
ObjectOutputStream	Serialize
ObjectStreamClass	Serialize
ObjectStreamField	Serialize
OutputStream	Superclass
OutputStreamWriter	Charakterbasierend
PipedInputStream	| zwei Kanäle
PipedOutputStream	| zwei Kanäle
PipedReader	| zwei Kanäle
PipedWriter	| zwei Kanäle

Klassen des io-Packages

PrintStream	println, syso
PrintWriter	println, syso
PushbackInputStream	preview
PushbackReader	preview, RandomAccessFile seek, HexEditor
Reader	abstrakte Klasse
SequenceInputStream	liest mehrere Stream hintereinander
SerializablePermission	Serialize
StreamTokenizer	lesen mit definierten Tokens (Parser)
StringBufferInputStream	deprecated,
StringReader	Liest aus einem String
StringWriter	Schreibt in einem String
Writer	abstrakte Klasse

Java: Ausgabe in eine Datei

Basis aller Ausgaben ist die Klasse OutputStream

Weitere Spezifikation durch z.B:

- FileOutputStream

Weitere Spezifikation durch:

- BufferedOutputStream
- PrintStream
- DataOutStream
- ZIPOutStream
- CheckedOutStream
- PipeOutStream
- RandomAccessFile mit seek

Java: Ausgabe in eine Datei (Double, int)

```
import java.io.*;
public class print2 {
      // Ausgabe ohne Buffer,  Ausgabe von Double Zahlen, binäres Format
  public static void main(String argv[]) {
     double d;
    try {
        FileOutputStream fout = new FileOutputStream("1.bin");
        DataOutputStream dout = new DataOutputStream(fout);
        d = 1234.0;
        dout.writeDouble(d);
        d = 1234.11111;
        dout.writeDouble(d);
        dout.writeInt(-1234);
        dout.close();
      }
      catch (IOException e) {
         System.err.println("IOException: " + e);
      }
  }  // main
}
```


Java: Methoden der Klasse DataOutputStream

DataOutputStream:

writeBoolean
writeByte (Schreiben einer 8-Bit Vorzeichenzahl)
writeChar (Schreiben einer 16-Bit vorzeichenlosen Zahl)
writeDouble (Schreiben einer Double-Zahl)
writeFloat (Schreiben einer Single-Zahl)
writeInt (Schreiben einer 32-Bit Vorzeichenzahl)
writeLong (Schreiben einer 64-Bit Vorzeichenzahl)
writeShort (Schreiben einer 16-Bit Vorzeichenzahl)
writeUTF // // Unicode Transformation Format
writeChars

Java: Lesen einer ASCII-Datei (neu)

```
public static void main(String argv[])  throws IOException {
   FileInputStream fin;
   InputStreamReader iin;
   LineNumberReader din;
// aktuelle Version zum Einlesen einer ASCII-Datei
   String sLine;
   try {
     fin = new FileInputStream("read3.java");
     iin = new InputStreamReader(fin);
     din = new LineNumberReader(iin);
     while  ( din.ready()  ) {
       sLine = din.readLine();
       System.out.println(sLine);
     }
   }
   catch (IOException e) {
         System.err.println("IOException: " + e);
   }
}
```


Java: Lesen einer Binärdatei: read4

```
import java.io.*;
public static void main(String argv[])  throws IOException {
   FileInputStream fin;
   DataInputStream din;
   char ch;
   byte b;
   try {
     fin = new FileInputStream("1.dat");
     din = new DataInputStream(fin);
     while  (true) {                          // eventuell Absturz mit Exception
       b =din.readByte();
       ch = (char) (b);
       System.out.println(ch);
     }
   }
   catch (IOException e) {
         System.err.println("IOException: " + e);
   }
   pause();
 }
```

Java: Lesen einer Binärdatei: read4

```
import java.io.*;

public class read4 {

 private static void read(String sFilename){
  FileInputStream fin;
  DataInputStream din;
  char ch;
  byte b;
  try {
   fin = new FileInputStream( sFilename );
   din = new DataInputStream(fin);
   while  ( din.available()>0 ) {                // ohne Fehler
    b  = din.readByte();
    ch = (char) (b);
    System.out.println(ch);
   }
  }
  catch (IOException e) {
       System.err.println("IOException: " + e);
  }
 }
```


Java: Lesen einer Binärdatei

```
import java.io.*;
// Einlesen aus einer binären Datei, in der int-Werte gespeichert wurden
public class read4 {
 private static void read(String sFilename){
    int i;
    try {
                FileInputStream Fin = new FileInputStream(sFilename);
                DataInputStream DataIn = new DataInputStream(Fin);
                i = DataIn.readInt();
                System.out.println(i);
                i = DataIn.readInt();                  // Einlesen von int Zahlen
                System.out.println(i);                 // Einlesen der Zahlen
                i = DataIn.readInt();                public static void main(String argv[]) {
                System.out.println(i);                 read("1.int");
                DataIn.close();                        pause();
    }                                                 } // main
    catch (IOException e) { }                        } // class
 }
```

Java: Schreiben und Lesen einer Binärdatei

```
import java.io.*;
    // Ausgabe ohne Buffer, Ausgabe von int Zahlen, Einlesen der Zahlen

 public class readwrite1 {

 public static void main(String argv[]) {
    int i;
    try {
      FileOutputStream fout = new FileOutputStream("rw1.int");
      DataOutputStream dout = new DataOutputStream(fout);
      i = 1234;
        dout.writeInt(i);
      i = 4321;
        dout.writeInt(i);
        dout.writeInt(-1234);
      dout.close();
```


Java: Schreiben und Lesen einer Binärdatei

```
       // weiter im Sourcecode
      FileInputStream fin = new FileInputStream("rw1.int");
            // Einlesen der Dateien
      DataInputStream din = new DataInputStream(fin);
      i = din.readInt();
       System.out.println(i);
      i = din.readInt();
       System.out.println(i);
      i = din.readInt();
       System.out.println(i);
      din.close();
    }
    catch (IOException e) {
       System.err.println("IOException: " + e);
    }
  System.out.println("Es wurde in die Datei rw1.int geschrieben und gelesen");
} // main
}
```

Java: Methoden der Klasse DataInputStream

DataInputStream:

readBoolean
readByte (…) Lesen einer 8-Bit Vorzeichenzahl
readChar (…) Lesen einer 16-Bit vorzeichenlosen Zahl
readDouble (…) Lesen einer Double-Zahl
readFloat (…) Lesen einer Single-Zahl
readInt (…) Lesen einer 32-Bit Vorzeichenzahl
readLong (…) Lesen einer 64-Bit Vorzeichenzahl
readShort (…) Lesen einer 16-Bit Vorzeichenzahl
readUTF(…) // // Unicode Transformation Format

Read/Write mit RandomAccessFile

Methoden:

- int read(byte[] b, int off, int len)
- boolean readBoolean()
- byte readByte()
- char readChar()
- double readDouble()
- float readFloat()
- int readInt()
- String readLine()
- long readLong()
- short readShort()
- String readUTF()

- int readUnsignedByte()
- int readUnsignedShort()

- void setLength(long newLength)
- int skipBytes(int n)

Read/Write mit RandomAccessFile

Methoden:

- length — Ausgabe der Länge der Datei
- void seek(long pos)
- void setLength(long newLength)
- int **skipBytes**(int n)

- void readFully(byte[] b)
  Reads b.length bytes from this file into the byte array, starting at the current file pointer.

- void readFully(byte[] b, int off, int len)
  Reads exactly len bytes from this file into the byte array, starting at the current file pointer.

Read/Write mit RandomAccessFile

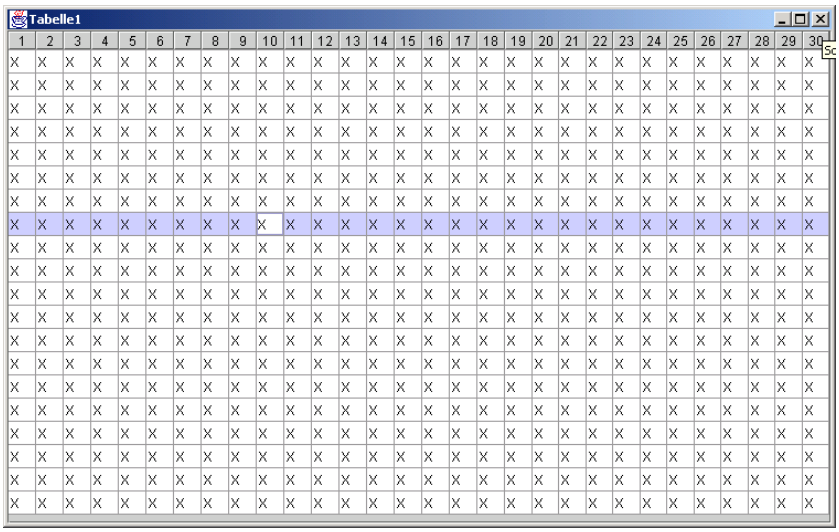
Beispiel:

```
private byte[] read(int count, int offset, String sFilename) {
      RandomAccessFile fin =  new RandomAccessFile( sFilename, "r");
      if (offset>0) fin.seek(offset);

      byte [] feld = new byte[count];
     int anz = fin.read(feld, 0, count);
     fin.close();
      return feld;
}
```

Int-Datentypen in Java

byte
from -128 to 127
short
from -32768 to 32767
int
from -2147483648 to 2147483647
long
from -9223372036854775808 to 9223372036854775807
char
from '\u0000' to '\uffff', =>, from 0 to 65535

Beispiel: TableView



Beispiel: TableView

import javafx.scene.control.TableView

Konstruktor:

- public TableView()
- public TableView(ObservableList<S> items))

Wichtige Methoden:

- •ObservableList<T> getItems()
  - void getItem().add(…)
  - void getItem().addAll(…)
  - void getItem().remove(int from, int to)
  - void getItem().removeAll(T, T, T)
  - void getItem().setAll(Collection<? extends E> colllection)
  - void getItem().setAll(T, T, T)
  - void getItem().sorted()
  - void getItem().sorted(comparator)
  - void getItems().filtered(Predicate<E> predicate)

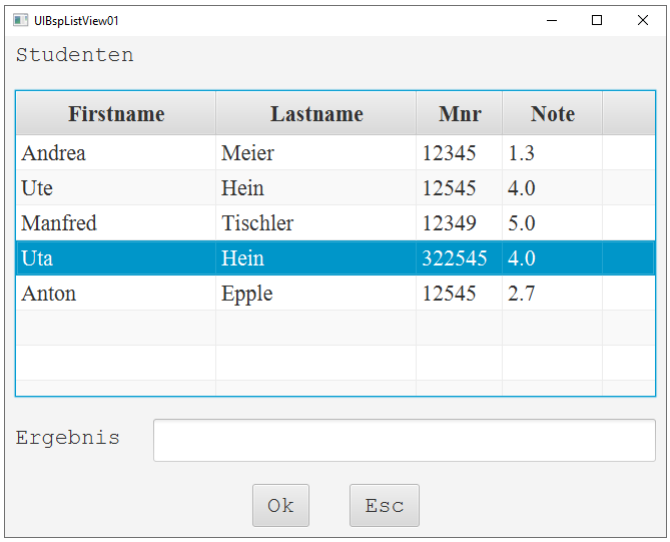
Beispiel: TableView

Weitere Modethoden:

- setCellFactory(Callback<ListView<T>,ListCell<T>> value)
- void setEditable(boolean value)
- void setFixedCellSize(double value)
- setOrientation(Orientation value)
- setPlaceholder(Node value)
- void setSelectionModel(MultipleSelectionModel<T> value)

- Import javafx.scene.control.MultipleSelectionModel<T>
- getSelectionModel().setSelectionMode(SelectionMode.SINGLE);
- getSelectionModel().setSelectionMode(SelectionMode.MULTIPLE);

Beispiel: UIBspTableView01.java



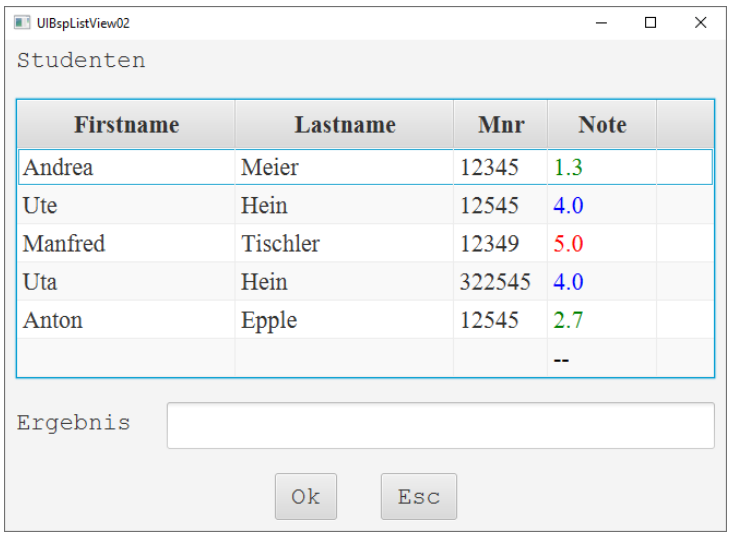
Beispiel: UIBspTableView01.java

```
private TableView tableview = new TableView() ;
   private  ObservableList<Student> listeStd =
FXCollections.observableArrayList();

listeStd.addAll(
            new Student("Andrea","Meier",12345,1.3),
            new Student("Ute","Hein",12545,4.0),
            new Student("Manfred","Tischler",12349,5.0),
            new Student("Anton","Epple",12545,2.7)
          );

 tableview.setItems(listeStd);
 tableview.setPlaceholder(new Label("Bitte etwas auswaehlen"));
 tableview.setStyle("-fx-font: 22px \"Serif\";");

tableview.getSelectionModel().setSelectionMode(SelectionMode.MULTIPLE);
```

Beispiel: UIBspTableView01.java

```
 TableColumn<Student, String> colFirstname  = new TableColumn<Student,
String>("Firstname");
 colFirstname.setMinWidth(200); // Mindestbreite setzen
 colFirstname.setCellValueFactory(   new PropertyValueFactory<Student,
String>("firstname"));

 TableColumn<Student, Integer> colMnr  = new TableColumn<Student,
Integer>("Mnr");
 colMnr.setCellValueFactory(   new PropertyValueFactory<Student,
Integer>("mnr")  );

 TableColumn<Student, Double> colNote  = new TableColumn<Student,
Double>("Note");
 colNote.setMinWidth(100); // Mindestbreite setzen
 colNote.setCellValueFactory(   new PropertyValueFactory<Student,
Double>("note")  );

tableview.getColumns().addAll(colFirstname, colLastname,colMnr, colNote);
```


UIBspTableView02.java (Farbe)



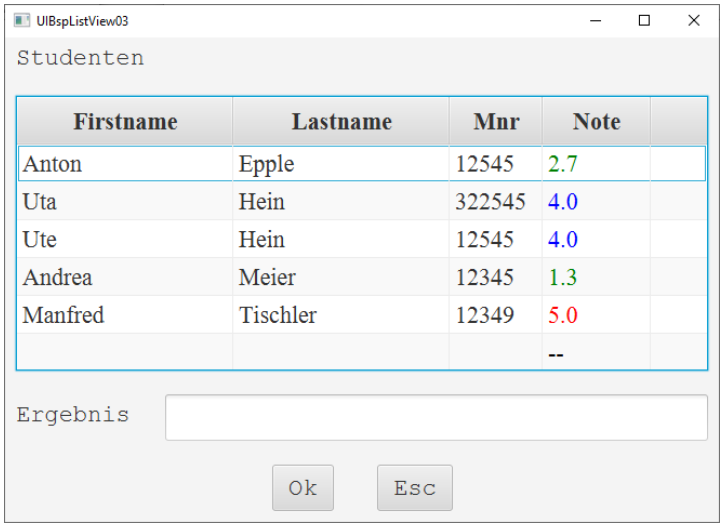
Beispiel: UIBspTableView02.java

```
class NoteCell extends TableCell<Student,Double> {

    @Override
    protected void updateItem(Double item, boolean empty) {
       super.updateItem(item, empty);
       if (item!=null && !empty) {
         setText(item.toString());
         if (item<3.0)
            setTextFill(Color.GREEN);
         else if (item<5.0)
            setTextFill(Color.BLUE);
         else
            setTextFill(Color.RED);
       }
       else {
         setText("--");
         setTextFill(Color.BLACK);
       }
    }
} // class NoteCell
```


Beispiel: UIBspTableView02.java

```
        // callback:  import javafx.util.callback;

colNote.setCellFactory(new
Callback<TableColumn<Student,Double>,
                              TableCell<Student,Double>>() {
     @Override
    public TableCell <Student,Double> call(
         TableColumn<Student,Double> param) {
              return new NoteCell();
          }
});
```

Beispiel: UIBspTableView03.java (sortiert)



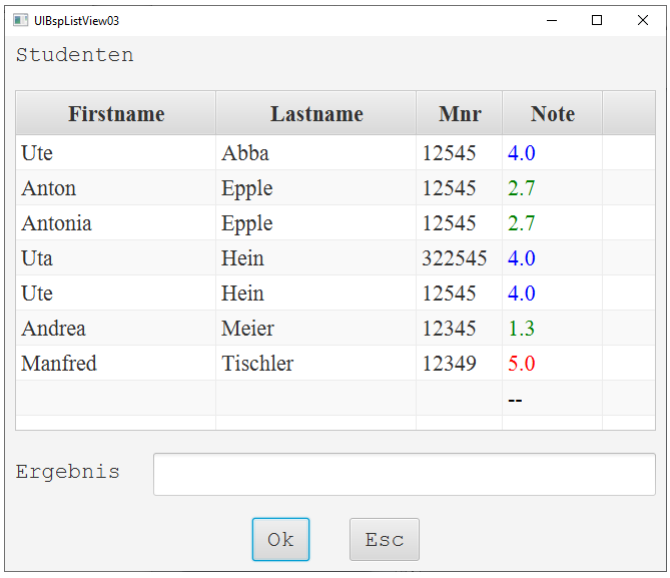
Beispiel: UIBspTableView03.java

```
TableView tableview = new TableView() ;
ObservableList<Student> listeStd = FXCollections.observableArrayList();
SortedList<Student> listeSortedStd = null;

Comparator<Student> stdComparer = new Comparator<Student>() {
   public int compare(Student std1, Student std2) {
      int result = std1.getLastname().compareTo(std2.getLastname());
      if (result==0) {
         result = std1.getFirstname().compareTo(std2.getFirstname());
      }
      return result;
   }
};

listeSortedStd = new SortedList<>(listeStd,stdComparer );
tableview.setItems(listeSortedStd);
```

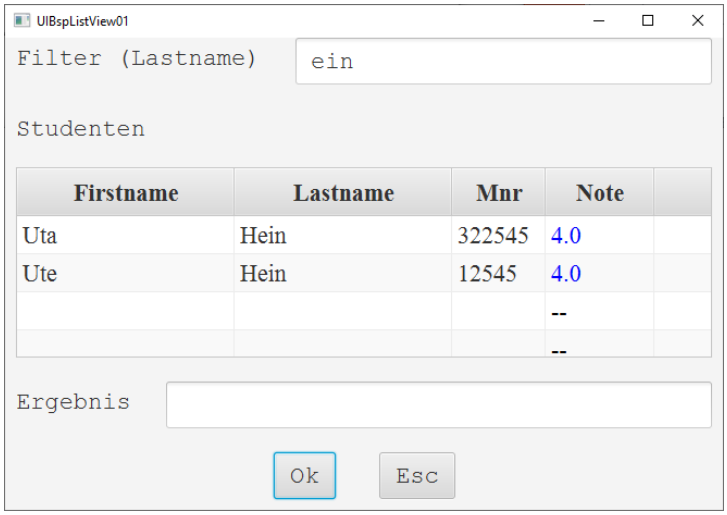
Beispiel: UIBspTableView03.java

```
private void insertStd() {

   listeStd.addAll(
            new Student("Ute","Abba",12545,4.0),
            new Student("Antonia","Epple",12545,2.7)
   );


}
```


Beispiel: UIBspTableView03.java (sortiert)



Beispiel: UIBspTableView04.java (Filter)



Beispiel: UIBspTableView04.java (Filter)

```
import javafx.collections.transformation.FilteredList;
import javafx.collections.transformation.SortedList;


TableView tableview = new TableView() ;

ObservableList<Student> listeStd =
        FXCollections.observableArrayList();

SortedList<Student> listeSortedStd = null;
```

Beispiel: UIBspTableView03.java (Filter)

```
FilteredList<Student> filteredList = new FilteredList<>(listeStd, p -> true);
tFilter.textProperty().addListener((observable, oldValue, newValue) -> {
    filteredList.setPredicate(std -> {
        if (newValue == null || newValue.isEmpty()) {
            return true;
        }
        String lowerCaseFilter = newValue.toLowerCase();
        if (std.getLastname().toLowerCase().contains(lowerCaseFilter)) {
            return true;
        } else if (std.getFirstname().toLowerCase().contains(lowerCaseFilter)) {
            return true;
        }
        return false; // Does not match.
    });
});
            listeSortedStd = new SortedList<>(filteredList,stdComparer );
            tableview.setItems(listeSortedStd);
```


Beispiel: UIBspTableView03.java (Filter)