Übung 5

# Aufgabe 17

## A17.java

class A17 {

public static void main(String[] args) {

Calculator calc = **new** Calculator();

calc.pack();

calc.setLocation(400, 400);

calc.setVisible(**true**);

}

}

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.awt.font.\*;

class Calculator **extends** Frame **implements** ActionListener

{

private TextField eingabe;

private Button b1, b2, b3, b4, b5, b6, b7, b8,

b9, bC, b0, bsign, bplus, bminus, bequal;

/\* Zuletzt eingegebene Zahl \*/

private int temp\_number = 0;

/\* Dient zum Speichern des zuletzt eingegebenen

\* Rechenzeichens \*/

private String temp\_calc = "x";

/\* Aktuelles Ergebnis \*/

private int result = 0;

/\* Dient als Indikator, ob als letzte Operation

\* eine Zahl oder ein Zeichen eingegeben wurde \*/

boolean is\_number = **false**;

## Calculator.java

public Calculator()

{

**super**();

setTitle("Rechner");

setBackground(Color.LIGHT\_GRAY);

/\* Groessere Schrift \*/

Font font1 = **new** Font("SansSerif", Font.PLAIN, 20);

setLayout(**new** BorderLayout(0, 10));

Panel tasten = **new** Panel(**new** GridLayout(0, 3, 3, 5));

eingabe = **new** TextField();

eingabe.setPreferredSize(**new** Dimension(200, 50));

eingabe.setFont(font1);

eingabe.setEditable(**false**);

eingabe.setText("");

tasten.add(b1 = **new** Button("1"));

tasten.add(b2 = **new** Button("2"));

tasten.add(b3 = **new** Button("3"));

tasten.add(b4 = **new** Button("4"));

tasten.add(b5 = **new** Button("5"));

tasten.add(b6 = **new** Button("6"));

tasten.add(b7 = **new** Button("7"));

tasten.add(b8 = **new** Button("8"));

tasten.add(b9 = **new** Button("9"));

tasten.add(bC = **new** Button("C"));

tasten.add(b0 = **new** Button("0"));

tasten.add(bsign = **new** Button("+/-"));

tasten.add(bplus = **new** Button("+"));

tasten.add(bminus = **new** Button("-"));

tasten.add(bequal = **new** Button("="));

b1.addActionListener(**this**);

b2.addActionListener(**this**);

b3.addActionListener(**this**);

b4.addActionListener(**this**);

b5.addActionListener(**this**);

b6.addActionListener(**this**);

b7.addActionListener(**this**);

b8.addActionListener(**this**);

b9.addActionListener(**this**);

bC.addActionListener(**this**);

b0.addActionListener(**this**);

bsign.addActionListener(**this**);

bplus.addActionListener(**this**);

bminus.addActionListener(**this**);

bequal.addActionListener(**this**);

add(eingabe, BorderLayout.NORTH);

add(tasten, BorderLayout.CENTER);

addWindowListener(**new** Closer());

}

public void actionPerformed(ActionEvent e)

{

String s = e.getActionCommand();

**switch** (s)

{

**case** "+/-":

**if** (is\_number) {

temp\_number \*= -1;

show(temp\_number);

} **else** {

**if** (eingabe.getText().equals("-"))

show("");

**else**

show("-");

}

**break**;

**case** "C":

temp\_number = 0;

temp\_calc = "x";

is\_number = **false**;

result = 0;

show("");

**break**;

**case** "=":

calc();

temp\_number = 0;

is\_number = **false**;

temp\_calc = "x";

show(result);

**break**;

**case** "+": **case** "-":

calc();

temp\_calc = s;

is\_number = **false**;

temp\_number = 0;

show(result);

**break**;

**default**:

/\* Zahlbutton wurde gedrueckt \*/

int i = Integer.parseInt(s);

temp\_number \*= 10;

**if** (temp\_number < 0)

temp\_number -= i;

**else**

temp\_number += i;

**if** (temp\_calc.equals("x"))

result = 0;

**if** (eingabe.getText().equals("-"))

temp\_number \*= -1;

is\_number = **true**;

show(temp\_number);

}

/\* Testausgabe in der Konsole zum Ueberpruefen und besseren

\* Verstaendnis der verwendeten Variablen \*/

System.out.println("result: " + result + " temp\_number: " + temp\_number + " temp\_calc: "

+ temp\_calc + " is\_number: " + is\_number);

}

private void calc()

{

**if** (temp\_calc.equals("+"))

result += temp\_number;

**else** **if** (temp\_calc.equals("-"))

result -= temp\_number;

**else**

result = temp\_number;

}

private void show(int number)

{

eingabe.setText(Integer.toString(number));

}

private void show(String s)

{

eingabe.setText(s);

}

protected static final class Closer **extends** WindowAdapter

{

public void windowClosing(WindowEvent e)

{

System.exit(0);

}

}

}

# Aufgabe 18

## A18.java

**import** java.awt.\*;

class A18

{

public static void main(String[] args)

{

Kontakt k1 = **new** Kontakt("Max Mueller");

Kontakt k2 = **new** Kontakt("Petra Schmid");

k1.setEmail("maxller@gmail.com");

k2.setEmail("petra@gmx.de");

Adressbuch adressbuch = **new** Adressbuch("Professoren");

adressbuch.add(k1);

adressbuch.add(k2);

VerteilerDialog dia = **new** VerteilerDialog(**new** Frame(), adressbuch);

dia.setSize(500, 300);

dia.setLocation(300, 300);

dia.setVisible(**true**);

}

}

## Adressbuch.java

**import** java.util.\*;

public class Adressbuch **implements** Iterable<Kontakt>

{

private String name;

private ArrayList<Kontakt> kontakte;

public Adressbuch(String name)

{

setName(name);

kontakte = **new** ArrayList<Kontakt>();

}

public String getName()

{

**return** name;

}

public void setName(String name)

{

**this**.name = name;

}

public void add(Kontakt k)

{

kontakte.add(k);

}

public boolean delete(Kontakt k)

{

**return** kontakte.remove(k);

}

public Iterator<Kontakt> iterator()

{

**return** kontakte.iterator();

}

}

## Kontakt.java

public class Kontakt

{

private String email;

private String name;

public Kontakt(String name)

{

setName(name);

}

public String getEmail()

{

**return** email;

}

public void setEmail(String email)

{

**this**.email = email;

}

public String getName()

{

**return** name;

}

public void setName(String name)

{

**this**.name = name;

}

public String toString()

{

**return** name;

}

}

## Verteiler.java

**import** java.awt.Dialog.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.util.ArrayList;

**import** java.util.Iterator;

class VerteilerDialog **extends** Dialog **implements** ActionListener

{

private TextField tf\_name;

private List lt\_adress;

private List lt\_verteiler;

private Button save;

private Button arrow;

public Dialog verteiler;

public VerteilerDialog(Frame f, Adressbuch adress)

{

**super**(f, "Mailverteiler erstellen");

/\* Initialisieren der Dialogkomponenten \*/

setLayout(**new** BorderLayout());

Panel pan1 = **new** Panel();

Panel pan2 = **new** Panel();

verteiler = **new** Dialog(f, "Mailverteiler erstellen");

tf\_name = **new** TextField();

lt\_adress = **new** List(10, **false**);

lt\_verteiler = **new** List(10, **false**);

save = **new** Button("Speichern");

arrow = **new** Button(">>");

arrow.addActionListener(**this**);

save.addActionListener(**this**);

lt\_adress.addActionListener(**this**);

tf\_name.setPreferredSize(**new** Dimension(80,25));

pan1.add(**new** Label("Name des Verteilers"));

pan1.add(tf\_name);

pan1.add(save);

Iterator<Kontakt> it = adress.iterator();

Kontakt k;

/\* Nur Kontakte anzeigen, die eine Emailadresse haben \*/

**while** (it.hasNext()) {

k = it.next();

**if** (k.getEmail() != **null**) {

lt\_adress.add(k.getName());

}

}

pan2.add(lt\_adress);

pan2.add(arrow);

pan2.add(lt\_verteiler);

add(pan1, BorderLayout.NORTH);

add(pan2, BorderLayout.CENTER);

addWindowListener(**new** Closer());

}

public void actionPerformed(ActionEvent e)

{

**if** ((e.getSource() == arrow) || (e.getSource() == lt\_adress)) {

**if** (lt\_adress.getSelectedItem() != **null**) {

lt\_verteiler.add(lt\_adress.getSelectedItem());

lt\_adress.remove(lt\_adress.getSelectedItem());

} **else** {

/\* Hier Fehlerbehandlung möglich, wenn erwünscht \*/

}

} **else** **if** (e.getSource() == save) {

/\* Hier Ereignisbehandlung für Button "Speichern" \*/

}

}

protected static final class Closer **extends** WindowAdapter

{

public void windowClosing(WindowEvent e)

{

System.exit(0);

}

}

}

# Pong (nur die bearbeiteten Klassen)

## HighscoreDialog.java

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.util.Iterator;

class HighscoreDialog **extends** Dialog **implements** ActionListener {

private Button b\_akt;

private List l\_scores;

private HighscoreContainer hc;

public HighscoreDialog(Frame f, HighscoreContainer hc) {

**super**(f, "Highscores");

**this**.hc = hc;

Panel pan = **new** Panel(**new** BorderLayout());

b\_akt = **new** Button("Aktualisieren");

l\_scores = **new** List(15, **false**);

b\_akt.addActionListener(**this**);

pan.add(b\_akt, BorderLayout.NORTH);

pan.add(l\_scores, BorderLayout.CENTER);

add(pan);

setSize(500,400);

setLocation(300,300);

}

public void actionPerformed(ActionEvent e) {

l\_scores.removeAll();

Iterator<Highscore> it = hc.iterator();

**while** (it.hasNext()) {

Highscore hs = it.next();

l\_scores.add(hs.getScore() + ": " + hs.getName() + " - " + hs.getDate());

}

}

}

## WarnDialog.java

**import** java.awt.\*;

**import** java.awt.event.\*;

class WarnDialog **extends** Dialog **implements** ActionListener{

Button b;

public WarnDialog(Window w, String s) {

**super**((Frame) w, "", **true**);

Panel pan = **new** Panel(**new** GridLayout(0,1));

pan.add(**new** Label(s));

b = **new** Button("Schliessen");

b.addActionListener(**this**);

pan.add(b);

add(pan);

setLocation(300,300);

pack();

setVisible(**true**);

}

public void actionPerformed(ActionEvent e) {

dispose();

}

}

## GameConfigurationDialog.java

**import** java.awt.\*;

**import** java.awt.event.\*;

class GameConfigurationDialog **extends** Dialog **implements** ActionListener {

TextField tf\_score;

TextField tf\_speed;

TextField tf\_acc;

Button ok;

Button abbrechen;

Configuration config;

public GameConfigurationDialog(Frame f, Configuration config) {

**super**(f, "Konfigurationen");

**this**.config = config;

Panel pan = **new** Panel(**new** GridLayout(0,2));

pan.add(**new** Label("Siegpunkte:"));

pan.add(tf\_score = **new** TextField());

pan.add(**new** Label("Ballgeschwindigkeit: "));

pan.add(tf\_speed = **new** TextField());

pan.add(**new** Label("Beschleunigung: "));

pan.add(tf\_acc = **new** TextField());

pan.add(ok = **new** Button("OK"));

pan.add(abbrechen = **new** Button("Abbrechen"));

ok.addActionListener(**this**);

abbrechen.addActionListener(**this**);

load();

add(pan);

pack();

setLocation(300,300);

setVisible(**true**);

}

private void save() {

**try** {

config.setWinScore(Integer.parseInt(tf\_score.getText()));

config.setInitialBallSpeed(Double.parseDouble(tf\_speed.getText()));

config.setBallSpeedup(Double.parseDouble(tf\_acc.getText()));

} **catch** (Exception ex) {

WarnDialog warn = **new** WarnDialog(**this**, "Ungültige Eingabe!");

}

}

private void load() {

tf\_score.setText(Integer.toString(config.getWinScore()));

tf\_speed.setText(Double.toString(config.getInitialBallSpeed()));

tf\_acc.setText(Double.toString(config.getBallSpeedup()));

}

public void actionPerformed(ActionEvent e) {

**if** (e.getSource() == ok) {

save();

dispose();

} **else** {

dispose();

}

}

}

## MainWindow.java

**import** java.awt.BorderLayout;

**import** java.awt.Font;

**import** java.awt.Frame;

**import** java.awt.GridLayout;

**import** java.awt.Label;

**import** java.awt.Menu;

**import** java.awt.MenuBar;

**import** java.awt.MenuItem;

**import** java.awt.Panel;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.awt.event.KeyEvent;

**import** java.awt.event.KeyListener;

**import** java.awt.event.WindowAdapter;

**import** java.awt.event.WindowEvent;

/\*\*

\* The MainWindow in which Pong is played

\*/

public class MainWindow **extends** Frame **implements** ActionListener, KeyListener {

/\*\*

\* Uniquely identifies this class

\*/

//Das Weglassen dieses Attributs erzeugt eine Compilerwarnung

private static final long serialVersionUID = 1994955233489697869L;

/\*\*

\* The instance of Pong which is played

\*/

private Pong pong;

MenuBar mb;

Menu mu\_game;

Menu mu\_config;

/\*\*

\* Displays parameters of Pong such as Ball position, ...

\*/

private PongDisplay display;

/\*\*

\* Display the scores of both players

\*/

private Label score1, score2;

/\*\*

\* Set to true when the user wants to start a new game

\*/

private boolean gameStarting;

/\*\*

\* Contructor for class MainWindow

\* initializes all the gui elements

\*

\* **@param** title The title of the window

\* **@param** p The instance of Pong which is played

\*/

public MainWindow(String title, Pong p) {

**super**(title);

**this**.pong = p;

gameStarting = **false**;

mb = **new** MenuBar();

setMenuBar(mb);

mu\_game = **new** Menu("Spiel");

mu\_game.add(**new** MenuItem("Starten"));

mu\_game.add(**new** MenuItem("Abbrechen"));

mu\_game.add(**new** MenuItem("Highscore Anzeigen"));

mu\_game.add(**new** MenuItem("Beenden"));

mb.add(mu\_game);

mu\_game.addActionListener(**this**);

mu\_config = **new** Menu("Einstellungen");

mu\_config.add(**new** MenuItem("Spieler"));

mu\_config.add(**new** MenuItem("Spiel"));

mb.add(mu\_config);

mu\_config.addActionListener(**this**);

addWindowListener(**new** WindowAdapter() {

public void windowClosing(WindowEvent e) {

exitGame();

}

});

BorderLayout bl = **new** BorderLayout();

setLayout(bl);

display = **new** PongDisplay();

add(display, BorderLayout.CENTER);

**this**.addKeyListener(**this**);

display.addKeyListener(**this**);

Panel scorePanel = **new** Panel();

scorePanel.setLayout(**new** GridLayout(1,2));

Font scoreFont = **new** Font("Arial", Font.BOLD, 24);

score1 = **new** Label("");

score1.setAlignment(Label.LEFT);

score1.setFont(scoreFont);

scorePanel.add(score1);

score2 = **new** Label("");

score2.setAlignment(Label.RIGHT);

score2.setFont(scoreFont);

scorePanel.add(score2);

add(scorePanel, BorderLayout.NORTH);

**this**.setSize(400, 400);

**this**.setLocation(300,300);

**this**.setVisible(**true**);

**if**(Configuration.instance().getPlayer1Name().isEmpty()

|| Configuration.instance().getPlayer2Name().isEmpty()) {

**new** WarnDialog(**this**, "Bitte geben Sie die Spielereinstellungen an!");

**new** PlayerConfigurationDialog(**this**);

}

update();

}

/\*\*

\* Called when the users clicks on a button

\*

\* **@param** e An ActionEvent which contains information about what was clicked

\*/

public void actionPerformed(ActionEvent e) {

String s = e.getActionCommand();

**switch**(s) {

**case** "Starten": gameStarting = **true**; **break**;

**case** "Abbrechen": abortGame(); **break**;

**case** "Highscore Anzeigen": **new** HighscoreDialog(**this**, HighscoreContainer.instance()); **break**;

**case** "Beenden": exitGame(); **break**;

**case** "Spieler": **new** PlayerConfigurationDialog(**this**); **break**;

**case** "Spiel": **new** GameConfigurationDialog(**this**, Configuration.instance()); **break**;

}

}

/\*\*

\* Called when the users presses a key

\*

\* **@param** e A KeyEvent which contains information about which key was pressed

\*/

public void keyPressed(KeyEvent e) {}

/\*\*

\* Called when the users releases a key

\*

\* **@param** e A KeyEvent which contains information about which key was released

\*/

public void keyReleased(KeyEvent e) {

**if**(!pong.getGameRunning())

**return**;

Configuration c = Configuration.instance();

**if**(e.getKeyCode() == c.getPlayer1KeyDown()) {

pong.getLinkPlayer(0).getLinkPaddle().move(10);

} **else** **if**(e.getKeyCode() == c.getPlayer1KeyUp()) {

pong.getLinkPlayer(0).getLinkPaddle().move(-10);

} **else** **if**(e.getKeyCode() == c.getPlayer2KeyDown()) {

pong.getLinkPlayer(1).getLinkPaddle().move(10);

} **else** **if**(e.getKeyCode() == c.getPlayer2KeyUp()) {

pong.getLinkPlayer(1).getLinkPaddle().move(-10);

}

pong.getLinkBall().step();

update();

}

/\*\*

\* Called when the users presses and releases a key

\*

\* **@param** e A KeyEvent which contains information about which key was typed

\*/

public void keyTyped(KeyEvent e) {

**if**(gameStarting) {

startGame();

gameStarting = **false**;

}

}

/\*\*

\* Called when the user chooses to start a new Game in the Menu

\*/

private void startGame() {

pong.gameStart();

}

/\*\*

\* Called when the user chooses to abort the currently running Game

\*/

private void abortGame() {

pong.gameAbort();

}

/\*\*

\* Called when the user chooses to exit the game or closes the window

\*/

private void exitGame() {

pong.gameExit();

**this**.dispose();

System.exit(0);

}

/\*\*

\* Called to update the gui elements with their new values

\*/

public void update() {

Configuration con = Configuration.instance();

score1.setText(con.getPlayer1Name() + ": " + Integer.toString(pong.getLinkPlayer(0).getScore()));

score2.setText(con.getPlayer2Name() + ": " + Integer.toString(pong.getLinkPlayer(1).getScore()));

display.setBallPosition(pong.getLinkBall().getXPosition(), pong.getLinkBall().getYPosition());

display.setPaddle1Position(pong.getLinkPlayer(0).getLinkPaddle().getPosition());

display.setPaddle2Position(pong.getLinkPlayer(1).getLinkPaddle().getPosition());

}

}