



DOKUMENTATION

PROGRAMMIERPRAKTIKUM SBWL VERTIEFUNGSKURS I

Dr. Michael Hahsler
Abteilung für Informationswirtschaft
Institut für Informationsverarbeitung und Informationswirtschaft
Wirtschaftsuniversität Wien
Augasse 2-6
A-1090 Wien, AUSTRIA

STADT – LAND - FLUSS

MARKUS KREUZINGER	0351189	J 151
BETTINA SATTLER	0153023	J 151

h0351189@wu-wien.ac.at

h0153023@wu-wien.ac.at

ABGABEDATUM: 24.1.2006

INHALTSVERZEICHNIS

1. Problemdefinition: Stadt-Land-Fluss.....	3
2. Analyse.....	5
2.1 Use-Case-Diagramm	5
2.2 Use-Case-Beschreibung	6
3. Design - Klassendiagramm.....	9
4. Implementierung - Beschreibung der Klassen	10
5. Wartung.....	12
6. Installation.....	12
7. Programmcode.....	13
7.1 Klasse Spiel	13
7.2 Klasse Eingabefelder	13
7.3 Klasse Zufallsgenerator	26
7.4 Klasse Lexikon	28

1. Problemdefinition: Stadt-Land-Fluss

Datum:

20.10.2005

Autoren:

Markus Kreuzinger, 0351189, h0351189@wu-wien.ac.at

Bettina Sattler, 0153023, h0153023@wu-wien.ac.at

Aufgabe:

Aufgabe dieses Projektes ist, das Spiel Stadt-Land-Fluss zu erstellen.

Spielbeschreibung:

Das Spiel besteht aus zumindest 3 Feldern, nämlich Stadt, Land und Fluss. Durch einen Zufallsgenerator wird ein Buchstabe ausgewählt. Danach hat man für einen bestimmten Zeitraum (selber zu bestimmen), zum Beispiel 30 Sekunden, Zeit, sich für die vorherigen Themenfelder Begriffe einfallen zu lassen, die mit dem durch den Zufallsgenerator bestimmten Buchstaben beginnen. Nach Ablauf der zuvor bestimmten Zeit, wird vom Programm die Richtigkeit der Begriffe beurteilt und jeweils 10 Punkte für richtige Begriffe vergeben. Die Gesamtpunkte werden nach jeder Runde automatisch aufaddiert.

Spielablauf:

Das Spiel kennt die Länder der ganzen Welt, sowie Städte und Flüsse aus Österreich. Wollen Sie das Spiel auf weitere Städte und Flüsse ausdehnen, siehe Punkte 5 der Spielanleitung!

1. Start des Spiels

Drücken Sie zuerst auf die Button "Neue Runde Starten" um den Zufallsgenerator zu starten und einen zufälligen Buchstaben zu generieren.

2. Ausfüllen der Felder

Schreiben Sie in die Felder "Stadt", "Land" und "Fluss" die zugehörigen Begriffe mit den zuvor bestimmten Anfangsbuchstaben.

3. Überprüfung

Durch einen Klick auf den Button "Ueberpruefen" kontrolliert das Programm die eingegebenen Begriffe auf ihre Richtigkeit. Im anschließenden Feld wird das Ergebnis der Überprüfung angezeigt.

4. Runde beenden

Nachdem Sie den Button "Runde beenden" geklickt haben, erscheint der aktuelle Punktestand sowie die durchschnittliche Punktezahl pro Runde. Für jeden richtigen Begriff erhalten Sie 10 Punkte.

5. Begriffe hinzufügen

Da dieses Spiel nur Städte und Flüsse aus Österreich enthält, ist es möglich, das Spiel zu erweitern. Es ist jederzeit möglich, durch eintippen des Begriffes in eines der drei Felder sowie durch drücken des entsprechenden Buttons („Stadt hinzufügen“, „Land hinzufügen“ oder „Fluss hinzufügen“) die Nachschlagedatenbank mit diesem Begriff zu erweitern. Ein eigenes Fenster öffnet sich zur Bestätigung der erfolgreichen Eintragung. VORSICHT: Das Löschen von bereits in die Datenbank hinzugefügten Begriffen ist nur manuell durch das Bearbeiten der entsprechenden Programmdatei („stadt.txt“, „land.txt“ oder „fluss.txt“) im Unterordner „text“ möglich.

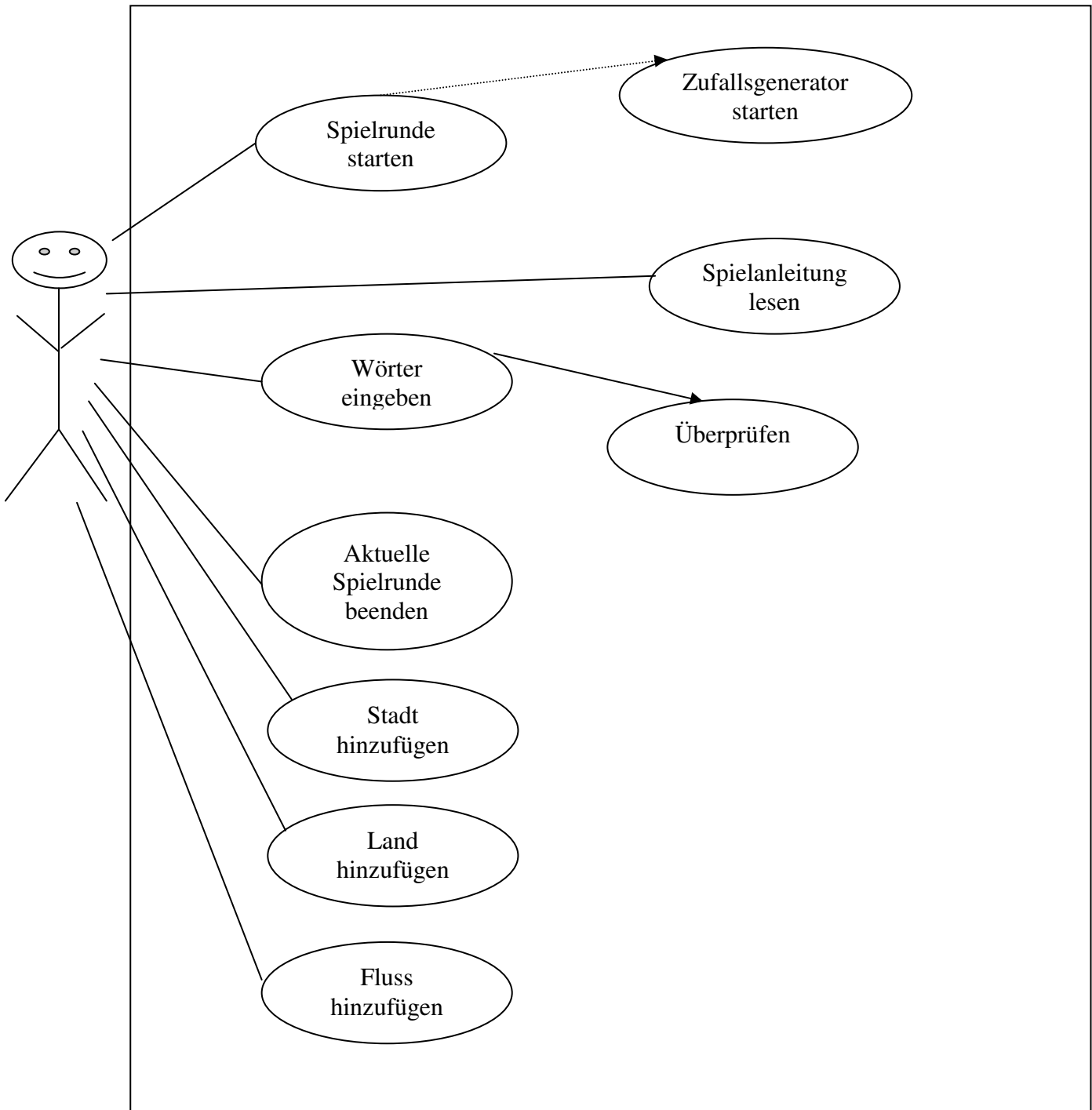
6. Fehlermeldungen

Bei einer fehlerhaften Bedienung des Spiels erscheint eine Fehlermeldung mit einem Hinweis, welcher Schritt als nächstes durchzuführen ist. Zum Beispiel ist es nicht möglich, in einer Runde mehrere Buchstaben zu generieren. Es ist erst nach Beendigung der aktuellen Spielrunde wieder möglich, einen neuen Buchstaben zu generieren. Auch ist es nicht möglich, durch mehrmaliges Drücken der Taste „Runde beenden“ den Punktestand irregulär zu erhöhen. Pro Runde darf nur einmal die Taste „Runde beenden“ betätigt werden.

Wir wünschen viel Spaß beim Spielen von "STADT - LAND - FLUSS"!

2. Analyse

2.1 Use-Case-Diagramm



2.2 Use-Case-Beschreibung

Beschreibung Use Case „Spielanleitung lesen“:

Use-Case Name:	Spielanleitung lesen
Akteure:	Spieler
Vorbedingungen:	Öffnen des Spiels
Nachbedingungen:	Spieler versteht die Spielregeln
Auslöser:	Klick auf den Button „Spielanleitung“
Ablaufbeschreibung:	- durch einen Klick auf den Button „Spielanleitung“ wird ein Textfeld mit der Spielanleitung geöffnet - der Spieler kann die Spielregeln lesen - der Spieler versteht die Spielregeln
Ergebnisse:	Der Spieler weiß über die Spielregeln bescheid und kann das Spiel laut Anleitung starten.

Beschreibung Use Case „Spielrunde starten“:

Use-Case Name:	Spielrunde starten
Akteure:	Spieler
Vorbedingungen:	Interesse am Spiel Stadt-Land-Fluss
Nachbedingungen:	Zufallsgenerator erzeugt Buchstaben
Auslöser:	Start des Programms
Ablaufbeschreibung:	- Das Programm Stadt-Land-Fluss wird gestartet - Zufallsgenerator startet den Durchlauf des Alphabets - Dem Spieler werden drei Felder angezeigt
Ergebnisse:	Der Spieler ist über die Feldanzahl sowie Feldbeschreibung (z.B. Feld „Stadt“) sowie über die Dauer einer Spielrunde informiert und kann das Spiel durch klicken des „Neue Runde starten“ starten.

Beschreibung Use Case „Wörter eingeben“ :

- Use-Case Name: Wörter eingeben
- Akteure: Spieler
- Vorbedingungen: Zufallsgenerator zeigt einen Buchstaben an
- Nachbedingungen: Wörter werden kontrolliert
- Auslöser: Buchstabe durch Zufallsgenerator freigegeben
- Ablaufbeschreibung: - Spieler kann Wörter in die Felder eintragen
-
- Variante: - Dem Spieler ist nicht zu jedem Themengebiet ein passendes Wort
eingefallen
- Nicht alle Felder sind ausgefüllt
-
- Ergebnisse: Der Spieler hat Wörter in die einzelnen Felder eingetragen.

Beschreibung Use Case „aktuelle Spielrunde beenden“:

- Use-Case Name: aktuelle Spielrunde beenden
- Akteure: Spieler
- Vorbedingungen: Wörter wurden eingegeben
- Nachbedingungen: Punkte vergeben
- Auslöser: Klick auf den Button „Aktuelle Spielrunde beenden“
- Ablaufbeschreibung: - Aktueller Punktstand wird angezeigt
- Für jede richtige Lösung werden 10 Punkte vergeben
- Durchschnittlicher Punktstand der bereits gespielten Runden wird
angezeigt
-
- Variation - Spieler beendet aktuelle Runde durch Drücken des “Runde beenden” –
Buttons vor dem Ausfüllen aller Felder
- Aktueller Punktstand wird angezeigt
- Für jede richtige Lösung werden 10 Punkte vergeben
- Durchschnittlicher Punktstand der bereits gespielten Runden wird
angezeigt

- Spieler bewertet seine Eingaben mit jeweils 10 Punkten für jede richtige Eingabe

Ergebnisse: Eine Spielrunde wurde damit durchgespielt. Die Eingaben des Spielers wurden mit Punkten bewertet. Anschließend startet die nächste Runde.

Beschreibung Use Case „Stadt hinzufügen“ :

Use-Case Name: Stadt hinzufügen

Akteure: Spieler

Vorbedingungen: Begriff in das Feld Stadt eingegeben

Nachbedingungen: Stadt wird in das Lexikon aufgenommen und in der nächsten Runde als richtig ausgewiesen

Auslöser: Stadt ist nicht im Lexikon enthalten

Ablaufbeschreibung: - Spieler kann Begriffe in das Lexikon hinzufügen
- durch Klicken auf den Button „Stadt hinzufügen“ wird die eingegeben Stadt im Lexikon aufgenommen

Variante: - Dem Spieler hat keinen Begriff eingegeben
- Es wird keine Stadt im Lexikon hinzugefügt

Ergebnisse: Eine bisher nicht im Lexikon enthaltene Stadt wurde in das Lexikon hinzugefügt

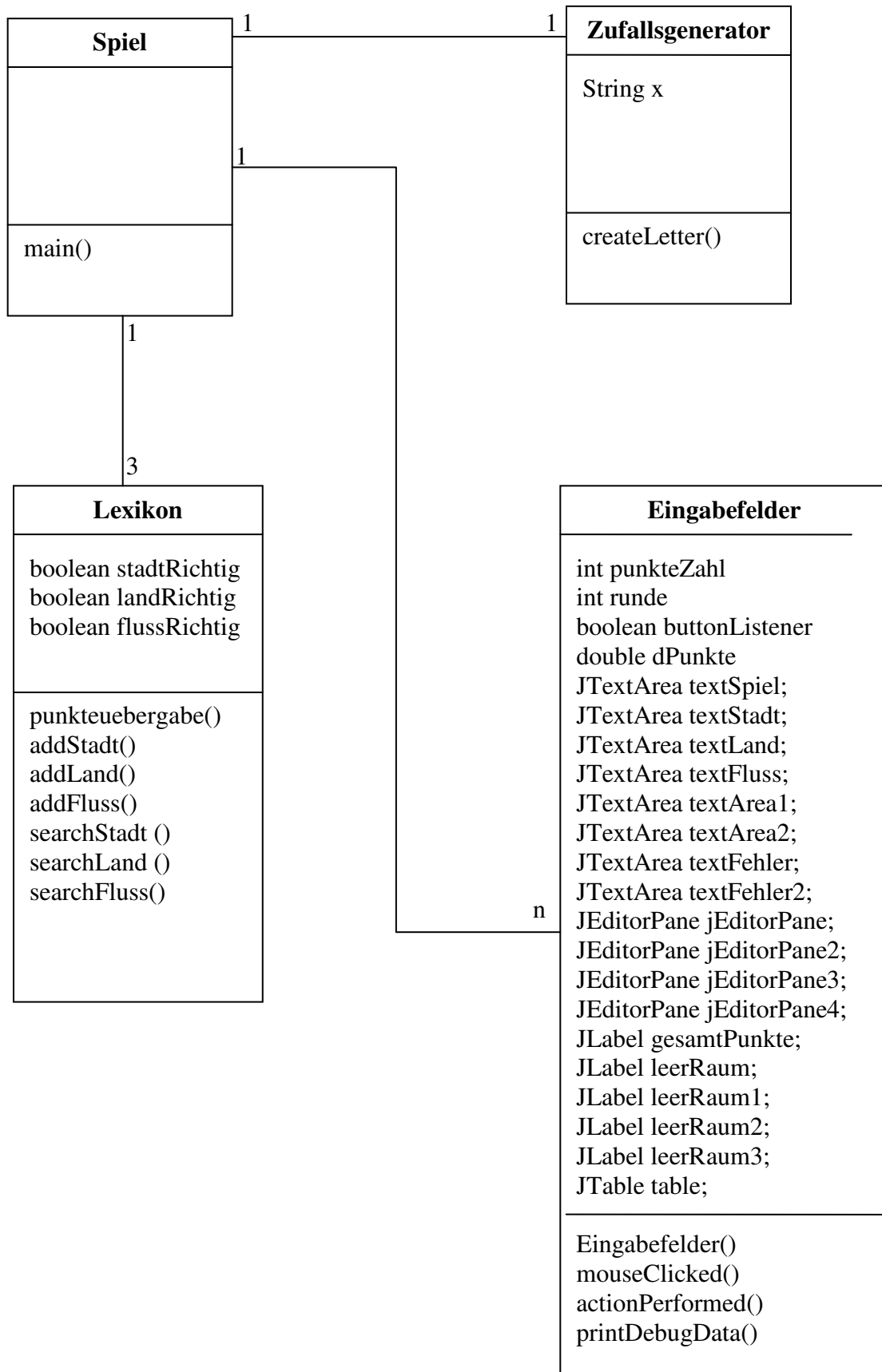
Beschreibung Use Case „Land hinzufügen“:

analog zu Use Case „Stadt hinzufügen“

Beschreibung Use Case „Fluss hinzufügen“:

analog zu Use Case „Stadt hinzufügen“

3. Design - Klassendiagramm



4. Implementierung - Beschreibung der Klassen

Klasse Spiel

Die Klasse Spiel bildet die „main-Klasse“. Von hier aus wird das Spiel gestartet.

Klasse Eingabefelder

In dieser Klasse wird das gesamte User-Interface erstellt (zuerst das Fenster, dann ein Raster = Grid und schließlich werden alle Buttons, Leerräume Eingabefelder und Textfelder zeilenweise eingebaut.

Daraufhin wird bei allen Buttons die dazugehörige Funktionalität ergänzt. Bei Betätigung des Buttons Spielanleitung wird ein neues Fenster geöffnet und der in der Datei „anleitung.txt“ eingegebene Inhalt darin ausgegeben.

Dem Button „Runde starten“ werden folgende Methoden und Funktionalitäten zugeteilt: Es wird der Zufallsgenerator aktiviert und gibt dieser einen Buchstaben sowie einen kurzen Erklärungstext aus. Weiters wird die Anzahl der gespielten Runden wiedergegeben. Es wird auch der „ButtonListener“ auf den Wert „true“ gesetzt und kann eine neue Runde erst dann wieder gestartet werden, wenn durch Betätigen des Buttons „Runde beenden“ der „ButtonListener“ damit wieder auf den Wert „false“ geändert wird, ansonsten wird ein Fenster geöffnet und darauf hingewiesen.

Der Button „Überprüfen“ schlägt in der Datenbank nach den eingegebenen Begriffen nach. Eine Punktwertung erfolgt hiermit aber noch nicht.

Erst nach dem drücken des „Runde beenden“ Buttons werden die Punkte hinzugefügt, ausgegeben und die durchschnittliche Punkteanzahl berechnet. Es wird auch der „ButtonListener“ auf „false“ geändert, damit wieder eine neue Runde gestartet werden kann. Auch wird wieder ein neues Fenster bei falscher Betätigung geöffnet.

Die Buttons „Stadt hinzufügen“, „Land hinzufügen“ und „Fluss hinzufügen“ fügen wie der Name schon sagt, den in das jeweilige Feld „Stadt“, „Land“ oder „Fluss“ eingegebenen Begriff in die Datenbank hinzu.

Klasse Zufallsgenerator

Die Klasse Zufallsgenerator erzeugt einen zufälligen Buchstaben aus dem Alphabet.

Klasse Lexikon

Die Klasse Lexikon beinhaltet alle Länder der Welt sowie die Städte und Flüsse Österreichs. Hier wird kontrolliert, ob die eingegebenen Begriffe richtig oder falsch sind. Weiters ist es möglich, zusätzlich Länder, Städte und Flüsse in das Lexikon hinzuzufügen.

5. Wartung

Für das Programm ist grundsätzlich keine Wartung vorgesehen. Treten dennoch Fehler auf, kann gerne eine E-Mail mit der Problembeschreibung an eine der folgenden Adressen gesendet werden: h0351189@wu-wien.ac.at oder h0153023@wu-wien.ac.at.

6. Installation

Um das Spiel auf Ihrem PC ausführen zu können, ist es vorerst notwendig den Code auf dem jeweiligen PC abzuspeichern. Anschließend ist es möglich das Spiel über die Kommandozeile zu laden. Dafür wechseln Sie zuerst in den Ordner in welches das Spiel gespeichert wurde und geben Sie anschließend „java Spiel“ ein.

1. Entpacken des Ordners "Stadt-Land-Fluss" in einen beliebig gewählten Ordner
2. Kompilieren der Datei "Spiel.java"
3. Start des Programmes: "java Spiel"

Vorsicht bei Verwendung des Betriebssystems Linux. Da im Programm Umlaute enthalten sind, ist vor der Kompilierung der entsprechende Zeichensatz im Betriebssystem einzustellen.

Das Kompilieren und das Starten des Spiels sind auch im Programm Eclipse möglich. Die Main-Klasse befindet sich in der Datei Spiel.java und deshalb muss diese Datei zum Starten des Spiels kompiliert werden.

7. Programmcode

7.1 Klasse Spiel

```
public class Spiel{  
    public static void main(String[] args) {  
        Eingabefelder newContentPane = new Eingabefelder();  
    }  
}
```

7.2 Klasse Eingabefelder

```
import javax.swing.*;  
import java.awt.*;  
import java.awt.event.MouseAdapter;  
import java.awt.event.MouseEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.ActionEvent;  
import java.io.BufferedReader;  
import java.io.File;  
import java.io.FileReader;  
import java.io.IOException;  
import java.lang.*;  
  
public class Eingabefelder implements ActionListener {  
  
    JTextArea textSpiel;  
    JTextArea textStadt;  
    JTextArea textLand;  
    JTextArea textFluss;  
    JTextArea textArea1;  
    JTextArea textArea2;  
    JTextArea textFehler;
```

```
JTextArea textFehler2;
JEditorPane jEditorPane;
JEditorPane jEditorPane2;
JEditorPane jEditorPane3;
JEditorPane jEditorPane4;
JLabel gesamtPunkte;
JLabel leerRaum;
JLabel leerRaum1;
JLabel leerRaum2;
JLabel leerRaum3;
JTable table;
int punkteZahl =0;
int runde = 0;
boolean buttonListener = false;
double dPunkte = 0.0;

public Eingabefelder() {
    JPanel pane = new JPanel(new GridLayout(8, 3));
    JFrame.setDefaultLookAndFeelDecorated(true);

    //Create and set up the window.
    JFrame frame = new JFrame("Herzlich Willkommen zum Spiel
STADT - LAND - FLUSS");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.add(pane);

    //Rand im ganzen Fenster einfügen
    pane.setBorder(BorderFactory.createEmptyBorder(
        30, //top
        30, //left
        10, //bottom
        30) //right
    );
}
```

```
//Leerraum erzeugen
leerRaum = new JLabel();
pane.add(leerRaum);

//Erzeugt den Button der den Zufallsgenerator aufruft
JButton b2 = new JButton("Neue Runde starten");
pane.add(b2);

//Buttonerzeugung zur Spielanleitung
JButton b6 = new JButton("Spielanleitung");
pane.add(b6);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Anzeige des Buchstaben
jEditorPane2 = new JEditorPane();
jEditorPane2.setEditable(false);
pane.add(jEditorPane2);
jEditorPane2.setForeground(java.awt.Color.red);
jEditorPane2.setFont(new java.awt.Font ("Arial", 0,40));

//Erklärungstext für Buchstabenanzeige und Rundenanzeige
jEditorPane = new JEditorPane();
jEditorPane.setEditable(false);
```

```
pane.add(jEditorPane);
jEditorPane.setForeground(java.awt.Color.red);
jEditorPane.setFont(new java.awt.Font ("Arial", 0,13));

//Anzeige der Rundenzahl
jEditorPane3 = new JEditorPane();
jEditorPane3.setEditable(false);
pane.add(jEditorPane3);
jEditorPane3.setForeground(java.awt.Color.red);
jEditorPane3.setFont(new java.awt.Font ("Arial", 0,40));
jEditorPane3.setText("Runde Nr. 1");

//Erzeugt Tabelle mit den Eingabefeldern
String[] columnNames = {"Stadt", "Land", "Fluss",};
Object [][] data = {
    {"", ""},
    {"", ""},
};

table = new JTable(data, columnNames);
table.setPreferredSize(new Dimension(300, 50));

//MouseListener, damit bei Feldwechsel die Daten aktualisiert werden
table.addMouseListener(new MouseAdapter() {
    public void mouseClicked(MouseEvent e) {
        printDebugData(table);
    }
});

//Scroll Pane erstellen und Tabelle hinzuf?gen
JScrollPane scrollPane = new JScrollPane(table);
pane.add(scrollPane);
```

```
//Buttonerzeugung
JButton b0 = new JButton("Ueberpruefen");
pane.add(b0);

textArea1 = new JTextArea();
pane.add(textArea1);

//3 Buttons f?r hinzuf?gen von neuen L?sungen
JButton b3 = new JButton("Stadt hinzufuegen");
pane.add(b3);

JButton b4 = new JButton("Land hinzufuegen");
pane.add(b4);

JButton b5 = new JButton("Fluss hinzufuegen");
pane.add(b5);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Leerraum erzeugen
leerRaum3 = new JLabel();
pane.add(leerRaum3);

//Buttonerzeugung f?r Gesamtpunkteaddition
JButton b1 = new JButton("Runde beenden");
pane.add(b1);
```

```
//Anzeige des Punktestandes
jEditorPane4 = new JEditorPane();
    jEditorPane4.setEditable(false);
    pane.add(jEditorPane4);
    jEditorPane4.setForeground(java.awt.Color.red);
    jEditorPane4.setFont(new java.awt.Font ("Arial", 0,20));

//Action-Listener zu Button b0 - b6 hinzufuegen

b0.addActionListener(this);
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
b6.addActionListener(this);

//Alle Komponenten des Fenster zusammenfassen und Fenster erzeugen
frame.pack();
frame.setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {

        //Aktionen darstellen
        System.out.println(e);

        if(e.getActionCommand().equals("Spielanleitung")){

            //JPanel f?r neues Fenster erzeugen

            JPanel pane1 = new JPanel(new GridLayout(1, 1));
            JFrame.setDefaultLookAndFeelDecorated(true);
            JFrame frame1 = new JFrame("Spielanleitung STADT - LAND -
                                    FLUSS");
```

```
frame1.add(panel);

//TextArea zu neuem Fenster hinzuf?gen
textSpiel = new JTextArea();
textSpiel.setEditable(false);
panel.add(textSpiel);

//BufferedReader zum Lesen der Datei "anleitung.txt"
String x = "";
String zeile;
BufferedReader eingabe;
FileReader text;
File anleitung;

try{
    anleitung= new File("text/anleitung.txt");
    text = new FileReader(anleitung);
    eingabe = new BufferedReader(text);
    while( (zeile = eingabe.readLine()) != null) {
        textSpiel.append(zeile);
        textSpiel.append("\n");
    }
    eingabe.close();
}

catch (IOException ex) {
    ex.printStackTrace();
}

//Fenster erstellen und anzeigen
frame1.pack();
frame1.setVisible(true);
}
if(e.getActionCommand().equals("Runde beenden")){
```

```
        if (buttonListener == true){
            javax.swing.table.TableModel model =table.getModel();

            //neue Instanz des Lexikons erstellen
            Lexikon lex = new Lexikon();

            //textArea1 l?schen
            textArea1.setText("");

            //Wort aus der Tabelle holen
            String feldStadt;
            feldStadt= ((String)model.getValueAt(0,0));

            //Stadt in Lexikon nachschlagen
            textArea1.append(lex.searchStadt(feldStadt));
            textArea1.setEditable(false);

            textArea1.setCaretPosition(textArea1.getDocument().getLength());

            String feldLand;
            feldLand= ((String)model.getValueAt(0,1));
            textArea1.append(lex.searchLand(feldLand));

            textArea1.setCaretPosition(textArea1.getDocument().getLength());

            String feldFluss;
            feldFluss= ((String)model.getValueAt(0,2));
            textArea1.append(lex.searchFluss(feldFluss));

            textArea1.setCaretPosition(textArea1.getDocument().getLength());

            //Punktestand ausgeben
            punkteZahl = (lex.punkteuebergabe(punkteZahl));
```

```
dPunkte = punkteZahl/runde;
jEditorPane4.setText("Aktueller      Punkttestand:      "      +
                    Integer.toString(punkteZahl)      +
                    "\nPunkte/Runde: " + dPunkte);

    buttonListener = false;
}
else {
    JPanel pane4 = new JPanel(new GridLayout(1, 1));
    JFrame.setDefaultLookAndFeelDecorated(true);
    JFrame frame4 = new JFrame("Fehler");
    frame4.add(pane4);
    textFehler2 = new JTextArea();
    textFehler2.setEditable(false);
    textFehler2.append("Nicht Schummeln!!! \n Sie müssen
                        zuerst eine neue Runde starten \n
                        bevor Sie diese dann wieder
                        beenden können.");
    pane4.add(textFehler2);
    frame4.pack();
    frame4.setVisible(true);
}
}
if(e.getActionCommand().equals("Ueberpruefen")){
    javax.swing.table.TableModel model =table.getModel();

    //neue Instanz des Lexikons erstellen
    Lexikon lex = new Lexikon();

    //Wort aus der Tabelle holen
    String feldStadt;
    feldStadt= ((String)model.getValueAt(0,0));
```

```
//textArea1 l?schen
textArea1.setText("");

//Stadt in Lexikon nachschlagen
textArea1.append(lex.searchStadt(feldStadt));
textArea1.setEditable(false);
textArea1.setCaretPosition(textArea1.getDocument().getLength());

String feldLand;
feldLand= ((String)model.getValueAt(0,1));
textArea1.append(lex.searchLand(feldLand));

textArea1.setCaretPosition(textArea1.getDocument().getLength());

String feldFluss;
feldFluss= ((String)model.getValueAt(0,2));
textArea1.append(lex.searchFluss(feldFluss));

textArea1.setCaretPosition(textArea1.getDocument().getLength());
}

if(e.getActionCommand().equals("Neue Runde starten")) {

//Wird nur gestartet, wenn Runde beendet wurde
if (buttonListener == false){

//textArea2 l?schen
jEditorPane.setText("");

//neue Instanz des Zufallsgenerator erstellen und ausf?hren
Zufallsgenerator zufall = new Zufallsgenerator();
jEditorPane.setText(" Bitte geben Sie die jeweiligen Begriffe
mit dem \n links dargestelltem
Anfangsbuchstaben ein! \n Auf der
```

rechten Seite sehen sie die akutele \n
Rundenzahl.");

```
jEditorPane2.setText(zufall.createLetter());
```

```
//Felder in Tabelle löschen
```

```
javax.swing.table.TableModel model =table.getModel();
```

```
model.setValueAt("",0,0);
```

```
model.setValueAt("",0,1);
```

```
model.setValueAt("",0,2);
```

```
//Rundenzahl ausgeben
```

```
runde++;
```

```
jEditorPane3.setText("Runde Nr. " + runde);
```

```
//setzt ButtonListener auf true, damit neue Runde nicht 2x  
gestartet werden kann
```

```
buttonListener = true;
```

```
}
```

```
//Wird nur gestartet, wenn Runde noch NICHT beendet wurde
```

```
else{
```

```
    JPanel pane3 = new JPanel(new GridLayout(1, 1));
```

```
    JFrame.setDefaultLookAndFeelDecorated(true);
```

```
    JFrame frame3 = new JFrame("Fehler");
```

```
    frame3.add(pane3);
```

```
    textFehler = new JTextArea();
```

```
    textFehler.setEditable(false);
```

```
    textFehler.append("Sie müssen zuerst die aktuelle Runde  
beenden \n bevor Sie eine neue Runde  
starten können.");
```

```
        pane3.add(textFehler);
        frame3.pack();
        frame3.setVisible(true);

    }
}

if(e.getActionCommand().equals("Stadt hinzufuegen")) {
    javax.swing.table.TableModel model =table.getModel();

    //Instanz von Lexikon erstellen, Wort holen und abspeichern
    Lexikon lex = new Lexikon();
    String feldStadt;
    feldStadt= ((String)model.getValueAt(0,0));
    lex.addStadt(feldStadt);

    //neue Fenster f?r Best?tigungstext erstellen
    JPanel pane2 = new JPanel(new GridLayout(1, 1));
    JFrame.setDefaultLookAndFeelDecorated(true);
    JFrame frame2 = new JFrame("Stadt Hinzuf?gen");
    frame2.add(pane2);
    textStadt = new JTextArea();
    textStadt.setEditable(false);
    textStadt.append("Die Stadt " + feldStadt + " wurde erfolgreich
        hinzugef?gt.");
    pane2.add(textStadt);
    frame2.pack();
    frame2.setVisible(true);

}

if(e.getActionCommand().equals("Land hinzufuegen")) {
    javax.swing.table.TableModel model =table.getModel();
```

```
//Instanz von Lexikon erstellen, Wort holen und abspeichern
```

```
Lexikon lex = new Lexikon();  
String feldLand;  
feldLand= ((String)model.getValueAt(0,1));  
lex.addLand(feldLand);
```

```
//neue Fenster f?r Best?tigungstext erstellen
```

```
JPanel pane3 = new JPanel(new GridLayout(1, 1));  
JFrame.setDefaultLookAndFeelDecorated(true);  
JFrame frame3 = new JFrame("Land Hinzuf?gen");  
frame3.add(pane3);  
textLand = new JTextArea();  
textLand.setEditable(false);  
textLand.append("Das Land " + feldLand + " wurde erfolgreich  
hinzugef?gt.");  
pane3.add(textLand);  
frame3.pack();  
frame3.setVisible(true);
```

```
}
```

```
if(e.getActionCommand().equals("Fluss hinzufuegen")) {  
    javax.swing.table.TableModel model =table.getModel();
```

```
//Instanz von Lexikon erstellen, Wort holen und abspeichern
```

```
Lexikon lex = new Lexikon();  
String feldFluss;  
feldFluss= ((String)model.getValueAt(0,2));  
lex.addFluss(feldFluss);
```

```
//neue Fenster f?r Best?tigungstext erstellen
```

```
JPanel pane4 = new JPanel(new GridLayout(1, 1));  
JFrame.setDefaultLookAndFeelDecorated(true);
```

```
        JFrame frame4 = new JFrame("Fluss Hinzuf?gen");
        frame4.add(pane4);
        textFluss = new JTextArea();
        textFluss.setEditable(false);
        textFluss.append("Der Fluss " + feldFluss + " wurde erfolgreich
                        hinzugef?gt.");
        pane4.add(textFluss);
        frame4.pack();
        frame4.setVisible(true);
    }
}

//Tabelleninhalt ausgeben
private void printDebugData(JTable table) {
    int numRows = table.getRowCount();
    int numCols = table.getColumnCount();
    javax.swing.table.TableModel model = table.getModel();

    System.out.println("Value of data: ");
    for (int i=0; i < numRows; i++) {
        System.out.print("  row " + i + " :");
        for (int j=0; j < numCols; j++) {
            System.out.print(" " + j + " " + model.getValueAt(i, j));
        }

        System.out.println();
    }

    System.out.println("-----");
}
}
```

7.3 Klasse Zufallsgenerator

```
import java.util.Random;
```

```
public class Zufallsgenerator {
```

```
// Zufallsgenerator erzeugt einen Buchstaben
```

```
public String createLetter() {
```

```
    String x = "";
```

```
    Random rand = new Random(); {
```

```
        String[] buchstabe;
```

```
        buchstabe = new String[26];
```

```
            buchstabe[0] = "A";
```

```
            buchstabe[1] = "B";
```

```
            buchstabe[2] = "C";
```

```
            buchstabe[3] = "D";
```

```
            buchstabe[4] = "E";
```

```
            buchstabe[5] = "F";
```

```
            buchstabe[6] = "G";
```

```
            buchstabe[7] = "H";
```

```
            buchstabe[8] = "I";
```

```
            buchstabe[9] = "J";
```

```
            buchstabe[10] = "K";
```

```
            buchstabe[11] = "L";
```

```
            buchstabe[12] = "M";
```

```
            buchstabe[13] = "N";
```

```
            buchstabe[14] = "O";
```

```
            buchstabe[15] = "P";
```

```
            buchstabe[16] = "Q";
```

```
            buchstabe[17] = "R";
```

```
            buchstabe[18] = "S";
```

```
            buchstabe[19] = "T";
```

```
            buchstabe[20] = "U";
```

```
            buchstabe[21] = "V";
```

```
            buchstabe[22] = "W";
```

```
            buchstabe[23] = "X";
```

```
        buchstabe[24] = "Y";  
        buchstabe[25] = "Z";  
        return x = buchstabe[rand.nextInt(26)+1];  
    }  
}  
}
```

7.4 Klasse Lexikon

```
import java.io.*;  
  
public class Lexikon {  
    boolean stadtRichtig = false;  
    boolean landRichtig = false;  
    boolean flussRichtig= false;  
  
    public int punkteuebergabe(int a){  
        System.out.println(a);  
        if (stadtRichtig == true){  
            a+=10;  
        }  
        if (landRichtig == true){  
            a+=10;  
        }  
        if (flussRichtig == true){  
            a+=10;  
        }  
        return a;  
    }  
    // neue Stadt hinzuf?gen  
    public void addStadt (String a){
```

```
try{
    BufferedWriter out = new BufferedWriter (new FileWriter
                                                ("text/stadt.txt", true));

    out.write(a);
    out.newLine();
    out.close();
}
catch (Exception e) {System.out.println(e);
}
}
```

// neues Land hinzuf?gen

```
public void addLand (String a){
```

```
try{
    BufferedWriter out = new BufferedWriter (new FileWriter
                                                ("text/land.txt", true));

    out.write(a);
    out.newLine();
    out.close();
}
catch (Exception e) {System.out.println(e);
}
}
```

// neuen Fluss hinzuf?gen

```
public void addFluss (String a){
```

```
try{
    BufferedWriter out = new BufferedWriter (new FileWriter
                                                ("text/fluss.txt", true));

    out.write(a);
    out.newLine();
    out.close();
}
```

```
        catch (Exception e) {System.out.println(e);  
        }  
    }
```

// eingegebene Stadt im Lexikon suchen

```
public String searchStadt(String a) {
```

```
    String x = "";
```

```
    String zeile;
```

```
    BufferedReader eingabe;
```

```
    FileReader text;
```

```
    File stadt;
```

```
        // Nachschlagen und ausgeben, ob eingegebene Stadt richtig oder falsch ist
```

```
    try{
```

```
        stadt= new File("text/stadt.txt");
```

```
        text = new FileReader(stadt);
```

```
        eingabe = new BufferedReader(text);
```

```
        while( (zeile = eingabe.readLine()) != null) {
```

```
            if (zeile.equals (a))
```

```
            {
```

```
                x= "Stadt " + a + " ist RICHTIG"+ "\n";
```

```
                eingabe.close();
```

```
                stadtRichtig = true;
```

```
            }
```

```
        else {
```

```
            x= "Stadt " + a + " ist FALSCH"+ "\n";
```

```
        }
```

```
    }
```

```
        eingabe.close();
```

```
    }
```

```
    catch (FileNotFoundException e){x= "Datei Stadt.txt nicht gefunden!" + "\n";
```

```
    }
    catch (IOException ex) {
        ex.printStackTrace();
    }
    return x;
}

// eingegebenes Land im Lexikon suchen
public String searchLand(String a) {

    String x = "";
    String zeile;
    BufferedReader eingabe;
    FileReader text;
    File land;

    // Nachschlagen und ausgeben, ob eingegebenes Land richtig oder falsch ist
    try{
        land= new File("text/land.txt");
        text = new FileReader(land);
        eingabe = new BufferedReader(text);
        while( (zeile = eingabe.readLine()) != null) {
            if (zeile.equals (a))
            {
                x= "Land " + a + " ist RICHTIG" + "\n";
                eingabe.close();
                landRichtig = true;
            }
            else {
                x= "Land " + a + " ist FALSCH"+ "\n";
            }
        }
        eingabe.close();
    }
    catch (FileNotFoundException e){x= "Datei Land.txt nicht gefunden!" + "\n";
```

```
    }  
    catch (IOException ex) {  
        ex.printStackTrace();  
    }  
    return x;  
}
```

// eingegebenen Fluss im Lexikon suchen

```
public String searchFluss(String a) {
```

```
    String x = "";
```

```
    String zeile;
```

```
    BufferedReader eingabe;
```

```
    FileReader text;
```

```
    File fluss;
```

// Nachschlagen und ausgeben, ob eingegebener Fluss richtig oder falsch ist

```
try{
```

```
    fluss= new File("text/fluss.txt");
```

```
    text = new FileReader(fluss);
```

```
    eingabe = new BufferedReader(text);
```

```
    while( (zeile = eingabe.readLine()) != null) {
```

```
        if (zeile.equals (a))
```

```
        {
```

```
            x= "Fluss " + a + " ist RICHTIG"+ "\n";
```

```
            eingabe.close();
```

```
            flussRichtig = true;
```

```
        }
```

```
        else {
```

```
            x= "Fluss " + a + " ist FALSCH"+ "\n";
```

```
        }
```

```
    }
```

```
    eingabe.close();
```

```
}
```

```
        catch (FileNotFoundException e){x= "Datei Fluss.txt nicht gefunden!" + "\n";  
        }  
        catch (IOException ex) {  
            ex.printStackTrace();  
        }  
        return x;  
    }  
}
```