FACHHOCHSCHULE GIESSEN FRIEDBERG

Dokumentenstrukturen

Dokumentation zum Projekt
„Die Welt von *BreakOut*“

von:	*Kristian Kraft* *Email: Kristian.Kraft@gmx.de* *Matrikelnummer: 80 59 17*
	Dirk Vincent Kops *Email: Vincent.Kops@gmx.de* *Matrikelnummer: 80 61 81*
Studiengang:	*Medieninformatik (Master of Science)*
Dozent:	*Prof. Dr. Stephan Euler*
Stand:	*06. 03. 2010*

Inhalt

Abbildungsverzeichnis

Aufgabenstellung

Die Welt von BreakOut
Eine Basisimplementierung des Spiels BreakOut in Java liegt vor. Mit XML-Technologien sollen folgende Erweiterungen realisiert werden:

a. Level-Verwaltung (XML-Format, Editor, …)
b. Benutzer und Bestenliste
c. Konfiguration (Farben, Sounds, …)

Anleitung JDOM

„JDOM ist eine XML-Darstellung in der Programmiersprache Java“. (Wikipedia)
Um JDOM verwenden zu können muss als erstes die benötigte API heruntergeladen werden.

Einbinden von JDOM in Eclipse

Die momentan (05.03.10) aktuelle Version ist unter dem folgenden Link verfügbar:
http://www.jdom.org/dist/binary/jdom-1.1.1.zip

Nach dem erfolgreichen Download muss die API eingebunden werden. In Eclipse müssen Sie hierfür das Projekt, in welches Sie JDOM einbinden wollen markieren, und nach einem Rechtsklick den Punkt „Properties“ auswählen. Dies können Sie auch durch die Tastenkombination „Alt + Enter“ erreichen. Daraufhin öffnet sich ein Fenster, in dem auf der linken Seite „Java Build Path“ ausgewählt werden muss. Nun muss auf der rechten Seite der Button „Add External JARs“ (Tastenkombination „Alt + x“) geklickt werden. In dem sich öffnenden Dialog muss die zuvor heruntergeladene „jdom.jar“ ausgewählt werden. Wenn Sie das heruntergeladene Zip-Paket extrahiert haben, liegt die gesuchte .jar-Datei im Ordner „jdom/build/“. Dies muss noch mit einem Klick auf „OK“ bestätigt werden. Auf demselben Weg müssen noch die sechs weiteren Dateien importiert werden, die im Ordner „jdom/lib/“ liegen.

Codebeispiel

Für das Schreiben und Lesen einer XML-Datei ist folgend ein kommentiertes Code-Beispiel, mit dem eine XML-Datei geschrieben wird. Für die Ausgabe besteht jeweils ein Beispiel für die gesamte XML-Datei, ein einzelnes Attribut eines Elements oder der Inhalt eines Elements.

Der nachfolgende Code ist ebenfalls Digital auf der CD abgelegt.

```
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;

import org.jdom.*;
import org.jdom.output.Format;
import org.jdom.output.XMLOutputter;

import org.xml.sax.InputSource;

import org.jdom.input.SAXBuilder;
```

```
public class xmlSchreiben {
	Element root = new Element("Knoten1");
	Document dokument = new Document(root);

	public xmlSchreiben() throws IOException, JDOMException{


		DocType doctype = new DocType("highscore", "beispiel.dtd");
		dokument.setDocType(doctype);
		HashMap piMap = new HashMap( 2 );
	piMap.put( "type", "text/xsl" );
		piMap.put( "href", "beispiel.xsl" );
		ProcessingInstruction procIns = new ProcessingInstruction("xml-
stylesheet",piMap);
		dokument.getContent().add(0, procIns);

		Element KnotenZwei = new Element("Knoten2");
		KnotenZwei.setAttribute("Attribute","BeispielAttribute");

		Element KnotenZweiEins = new Element("Knoten21");
		KnotenZweiEins.addContent("Beispielinhalt 1");

		Element KnotenZweiZwei = new Element("Knoten22");
		KnotenZweiZwei.addContent("Beispielinhalt 2");

		Element KnotenZweiDrei = new Element("Knoten23");
		KnotenZweiDrei.addContent("Beispielinhalt 3");

		KnotenZwei.addContent(KnotenZweiEins);
		KnotenZwei.addContent(KnotenZweiZwei);
		KnotenZwei.addContent(KnotenZweiDrei);

		root.addContent(KnotenZwei);

		XMLOutputter outputter = new
XMLOutputter(Format.getPrettyFormat());
		FileOutputStream output;

		try {
		output = new FileOutputStream("xml/beispiel.xml");
		  outputter.output(dokument,output);
		 } catch (FileNotFoundException e) {
		  // TODO Auto-generated catch block
		  e.printStackTrace();
		 }
		 xmlLesen();
	}

	public void xmlLesen() throws IOException, JDOMException{

		//Um das gesamte XML-Dokument in der Konsole auszugeben:
		/*Format format = Format.getPrettyFormat();
		format.setEncoding("iso-8859-1");
		XMLOutputter out = new XMLOutputter(format);
		out.output( dokument, System.out );*/

		//Um einen bestimmten Knoten ausgeben zu können:
		SAXBuilder builder = new SAXBuilder();
		Document doc = builder.build( "xml/beispiel.xml" );
		Element beispiel = doc.getRootElement();
```

```
            //Beispiel für Inhalt eines Knotens
            System.out.println("Beispiel für Inhalt eines Knotens:");

      System.out.println(beispiel.getChild("Knoten2").getChild("Knoten21").
getText());

            //Beispiel für Attribute eines Knotens
            System.out.println("Beispiel für Attribute eines Knotens");

      System.out.println(beispiel.getChild("Knoten2").getAttribute("Attribu
te").getValue());

      }
      /**
       * @param args
       * @throws JDOMException
       * @throws IOException
       */
      public static void main(String[] args) throws JDOMException,
IOException {
            // TODO Auto-generated method stub
                  xmlSchreiben xmlSchreibenTest = new xmlSchreiben();
      }
}
```

Für die geschriebene XML-Datei sind zusätzlich eine DTD-, eine CSS und eine XSL-Datei angelegt. Innerhalb der DTD wird die Struktur der XML vorgeschrieben; die XSL jedoch ist für die formatierte und transformierte Ansicht der XML-Datei vorhanden. Mit der CSS können zusätzliche Formatierungen auf die Datei angewendet werden.

Erweiterungen des Spiels „BreakOut“

Im folgenden Kapitel werden die Erweiterungen des Spiels „BreakOut“ genannt und kurz erläutert.

GameOver

Nachdem der Spieler sein letztes Leben verloren hat, muss er als erstes seinen Namen eintragen und kann daraufhin auswählen, ob er nochmal spielen oder das Spiel beenden will.

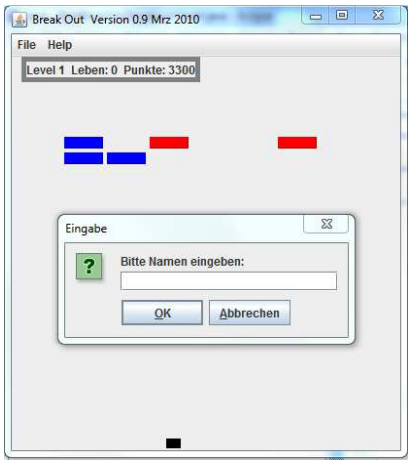

Abbildung 1 - Eintragen des Namens nach verlorenem Spiel

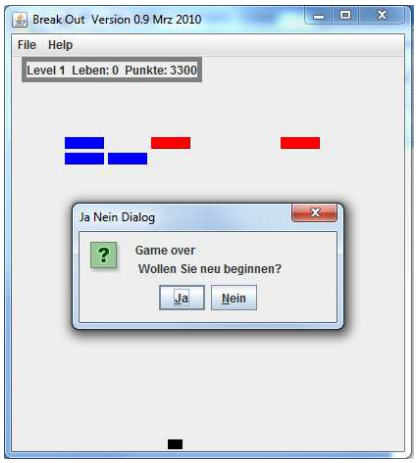

Abbildung 2 - Game Over - Abfrage "Wollen Sie neu beginnen?"

Spielerliste

Der nach dem Ende des Spiels eingetragene Spielername und dessen Punktzahl werden in einer XML-Datei gespeichert. Zusätzlich zu der XML-Datei gibt es zur formatierten Ausgabe noch eine CSS-, DTD- und XSL-Datei. Über die XSL wird die Ausgabe transformiert. Die Punkte der einzelnen Spieler werden zusammen gezählt und ausgegeben.

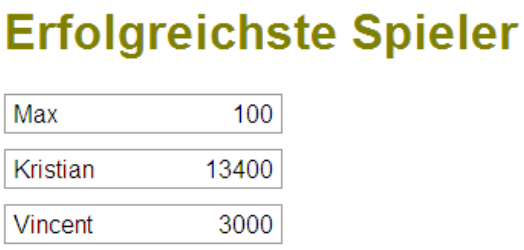

Abbildung 3 – Spielerliste.xml im Browser betrachtet

Highscore

Ähnlich zur Spielerliste werden in der Highscoreliste die Spielernamen und die erreichte Punktzahl abgespeichert. Der Unterschied besteht lediglich darin, dass in der Highscoreliste immer nur die fünf besten Spieler gespeichert sind, und in der Spielerliste alle gespeichert werden. Die Highscoreliste darf zurzeit nicht leer sein, da es sonst zu einem Fehler kommt.

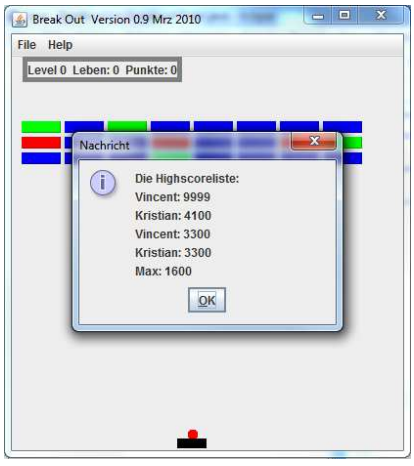

Abbildung 4 - Anzeigen der Highscoreliste

Farbliste

Über verschiedene Farbschemas kann das Aussehen der Spiels verändert werden. Diese können vom Nutzer gewechselt werden, indem im Menü unter „File“ auf „ChangeColorset“ geklickt wird. Das aktuelle Farbschema wird in einer XML-Datei (colorset.xml) abgespeichert und ist so auch beim nächsten Aufruf ausgewählt.

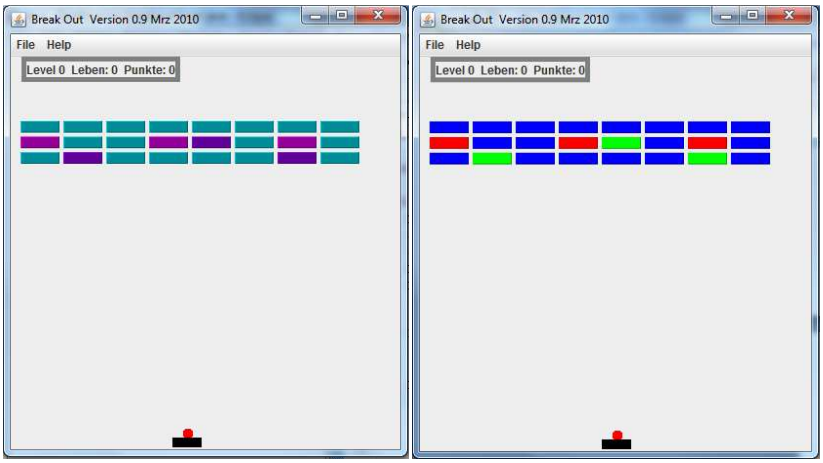

Abbildung 5 - Farbschema 1 und 2

Abbildung 6 - Darstellung der beiden Farbschemas

Spielstopp

Sobald sich der Nutzer mit der Maus außerhalb des Spielbereichs befindet, stoppt das Spiel, wodurch der Nutzer sich beispielsweise die Highscoreliste angucken oder in einem anderen Fenster arbeiten kann, ohne dass das Spiel weiterläuft.

Sounds

Ebenfalls über eine XML-Datei wird das Spiel um Sounds erweitert. Zurzeit stehen noch keine passenden Sounds zur Verfügung. Sobald dies soweit ist, können diese jedoch einfach eingebunden werden. Momentan sind Sounds aus dem Internet in das Spiel eingebunden, die jedoch nicht gekauft wurden und nur die Funktion darstellen sollen.

Levelverwaltung

Die Levelverwaltung ist bisher noch nicht in das Spiel integriert. Sie soll später dazu dienen eigene Level zu erstellen, die dann gespielt werden können. Die Datei „level.xml“ enthält zwei Level, die auf der von uns erstellen Struktur aus der Datei „level.dtd“ basieren. Die XML-Datei wird via XSL in XHTML transformiert und über CSS formatiert, so dass sich die Level im Browser anzeigen lassen.

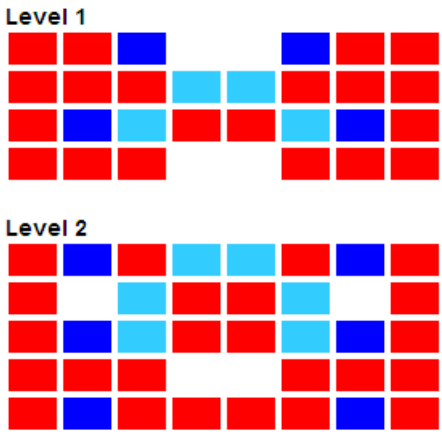

Abbildung 7 - Darstellung der Level im Browser