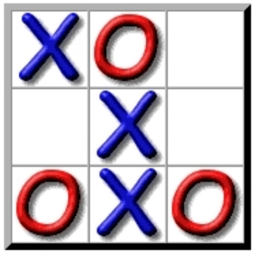
Dr. Johannes Mayer
Hans Braxmeier
Ralph Guderlei
Abteilung Angewandte Informationsverarbeitung

Design und Architektur von Softwaresystemen (SS 2005)

Abgabetermin: 24. Juni 2005

1 Tic–Tac–Toe und Web–GUI (10 Punkte)

Implementieren Sie das Spiel Tic–Tac–Toe als Webbasierte Anwendung (Servlet) mit einer Web–GUI (GUI mit ereignis-orientierter Verarbeitung so ähnlich wie Swing), die auf den folgenden Seiten beschrieben ist. Tic–Tac–Toe wird auf einem 3x3 großen Feld gespielt, wobei die Spieler abwechselnd einen Stein setzen dürfen. Gelingt es einem Spieler 3 Steine in einer Reihe (senkrecht, waagrecht oder diagonal) zu platzieren, hat er gewonnen. Das Spiel endet unentschieden, falls kein Spieler eine 3er–Reihe bilden konnte und alle Felder belegt sind.



Ein mögliche Spielsituation

Implementieren Sie eine Web–GUI (d.h. allgemeine GUI-Klassen und spezielle für Tic–Tac–Toe) entsprechend des auf der letzten Seite dargestellten Klassendiagramms. Ihre GUI-Klassen sollen folgendes Programm lauffähig machen. Entwickeln sie analog zu folgendem Programm auch ein Hauptprogramm für Tic–Tac–Toe.

Die Klasse WebGUI (*nicht im Klassendiagramm dargestellt*)

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.*;
import javax.servlet.http.*;

public class WebGUI extends HttpServlet {

        public void doGet(HttpServletRequest req, HttpServletResponse res)
                throws IOException, ServletException {

                res.setContentType("text/html");
                HttpSession sess = req.getSession();

                PrintWriter out = res.getWriter();

                if (sess.isNew()){

                        WWindow ww = new WWindow();

                        // Einzelkomponenten erzeugen
                        WButton b1 = new WButton("Click B1");
                        WButton b2 = new WButton("Click B2");
                        final WComponent l1 = new WLabel("Label L1");
                        WLabel l2 = new WLabel("Label L2");
                        WLabel l3 = new WLabel("Label L3");
                        WLabel l4 = new WLabel("Label L4");
                        WLabel l5 = new WLabel("Label L5");
                        WLabel l6 = new WLabel("Label L6");
                        WLabel l7 = new WLabel("Label L7");
                        WLabel l8 = new WLabel("Label L8");
                        final WTextField f1 = new WTextField("TextField F1");
                        WPanel wp = new WPanel(3); // Tabelle mit 3 Spalten erzeugen

                        // Action Listener hinzufuegen
                        b1.addActionListener(new WActionListener(){
                        public void actionPerformed() {
                                ((WLabel)l1).setText("Button 1 pressed");}});

                        b2.addActionListener(new WActionListener(){
                        public void actionPerformed() {
                                ((WLabel)l1).setText("Button 2 pressed");}});
                        b2.addActionListener(new WActionListener(){
                        public void actionPerformed() {
```

```
                    ((WTextField)f1).setText("Button 2 pressed"); }});

                // Einzelkomponenten hinzufuegen
                wp.addComponent(l1);
                wp.addComponent(l2);
                wp.addComponent(l3);
                wp.addComponent(l4);
                wp.addComponent(l5);
                wp.addComponent(l6);
                wp.addComponent(l7);
                wp.addComponent(l8);

                ww.addComponent(b1);
                ww.addComponent(b2);
                ww.addComponent(f1);
                ww.addComponent(wp);

                // HTML Code erzeugen
                ww.generateHtml(req, res);

                // GUI in Session speichern
                sess.setAttribute("GUI", ww);
            }
            else {
                WWindow ww = (WWindow)(sess.getAttribute("GUI"));
                ww.process(req);
                ww.generateHtml(req, res);
            }
        }

        public void doPost(HttpServletRequest req, HttpServletResponse res)
                throws IOException, ServletException {
                doGet(req, res);
        }
}
```

Beim Start einer neuen Session wird die Web–GUI mit all ihren Komponenten erzeugt (vgl. Quelltext). Anschließend wird die komplette GUI als Attribut in der Session gespeichert.

Bei jeder Anfrage wird die GUI zunächst aus der Session geholt oder (zu Beginn) neu erzeugt. Dann werden die Aktionen verarbeitet: Auf Basis des Requests reagieren die GUI-Elemente (z.B. Button erzeugt Event, daraufhin ändert das Eingabefeld seinen Wert). Zum Schluss wird die HTML-Ausgabe erzeugt.

Die Klasse WWindow

Die Klasse `WWindow` enthält eine Reihe von Elementen von Typ `WComponent`. Diese können u.U. via WPanel geschachtelt werden. Beim Aufruf der `process()`-Methode wird dieser Aufruf einfach rekursiv an alle enthaltenen Komponenten weitergeleitet und so weiter. Die entsprechenden Komponenten (Buttons und TextFields) reagieren dann auf die „Events“. Beim Aufruf der `generateHtml()`-Methode findet auch ein rekursiver Aufruf über alle Komponenten statt. Jede Komponente erzeugt erst jetzt (!) den HTML-Code und dieser wird von der `generateHtml()`-Methode von `WWindow` „zusammen gebaut“.

Das Interface WComponent

Das Interface `WComponent` stellt Methoden für die Komponenten zur Verfügung.

Die Klasse WPanel

Mit Hilfe der Methode `addComponent` können einem Panel beliebig viele Komponenten zugeordent werden. In der Methode `generateHtml()` wird dann über alle Komponenten iteriert um diese anschließend in tabellarischer Form auszugeben. Die Anzahl der Spalten kann über den Konstruktor festgelegt werden. Die `process()`-Methode wird an alle enthaltenen Komponenten weiter geleitet – ebenso wie `generateHtml()`.

Wie im Beispiel zu sehen ist, wurde mit Hilfe des Panels eine Tabelle mit 3 Spalten angelegt, in der sich 8 Komponenten befinden.

Die Klasse WButton

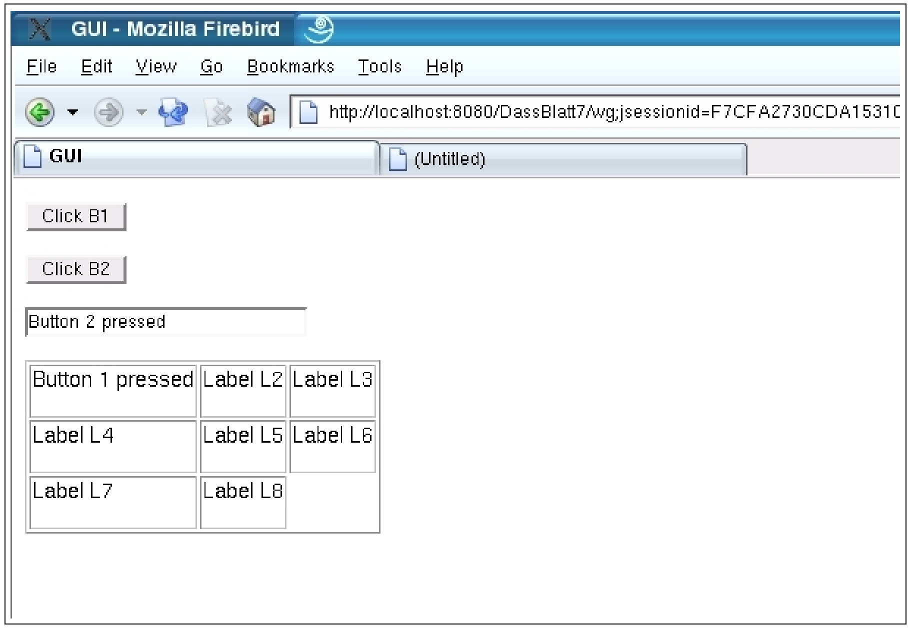
Mit dieser Klasse können Submit-Buttons repräsentiert werden. Beim Drücken eines solchen Buttons (dies merkt man bei bei `process()`, wenn ein entsprechender Parameter im Request enthalten ist) werden alle registrierten `ActionListener` benachrichtigt, d.h. es wird die Methode `ActionPerformed` aufgerufen.

Die Klasse WTextField

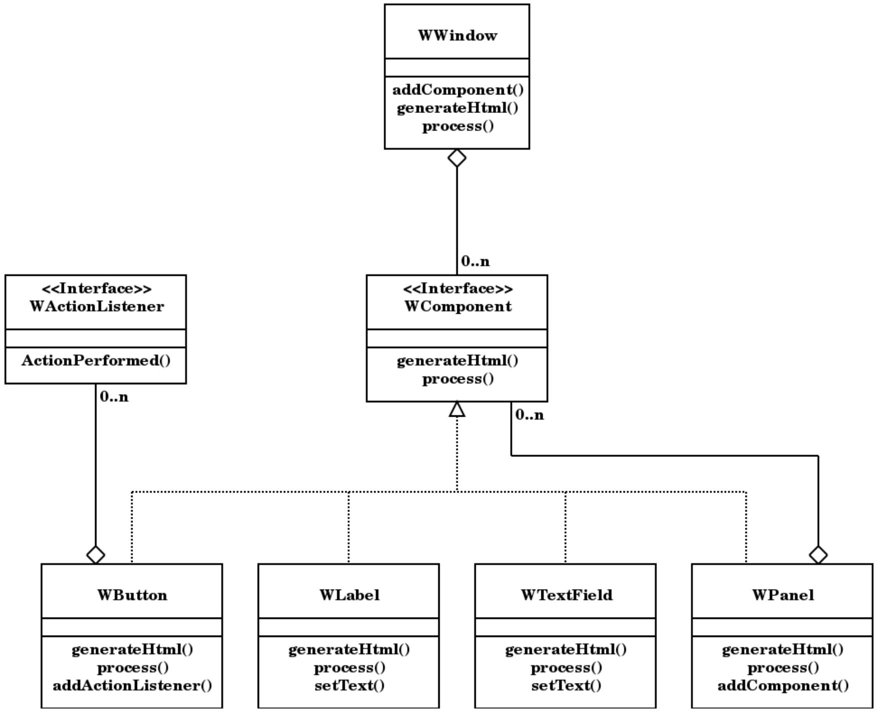
Hiermit können Formular-Eingabefelder repräsentiert werden. Bei `process()` aktualisiert dieses evtl. den Text, wenn ein entsprechender Parameter existiert.

Die Klasse WebId *(nicht im Klassendiagramm dargestellt)*

Die Klasse `WebId` dient zur Erzeugung von eindeutigen Ids. Achten Sie hierbei auf Synchronisation, um die Eindeutigkeit der Id’s zu garantieren.



Web–GUI mit verschiedenen GUI–Komponenten



Klassendiagramm

Viel Erfolg!