UNIKASSEL
VERSITÄT

Research Group
Software Engineering

Programmiermethodik

Übung 7

Wintersemester 2011 / 12

Fachgebiet Software Engineering

Tobias George

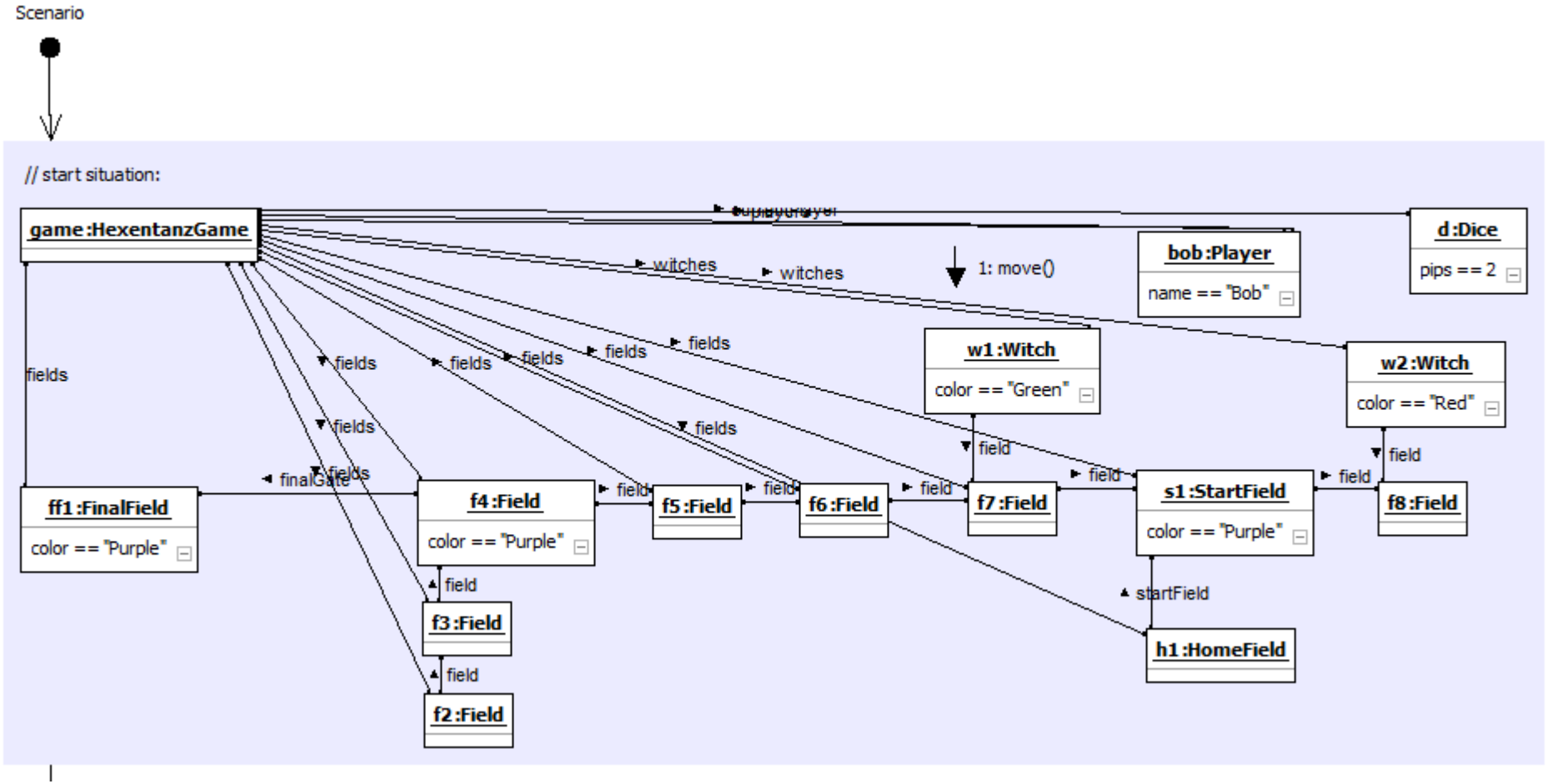
george@uni-kassel.de

Agenda

- **Besprechung HA 5**
- **Entwicklung von grafischen Oberflächen**
  - Mock-Ups
  - GUI Builder
  - Swing
  - SWT
  - Demo
- **Praktische Übung: Erstellen eines Hexentanz Login Screens**
- **Vorschau HA 6**

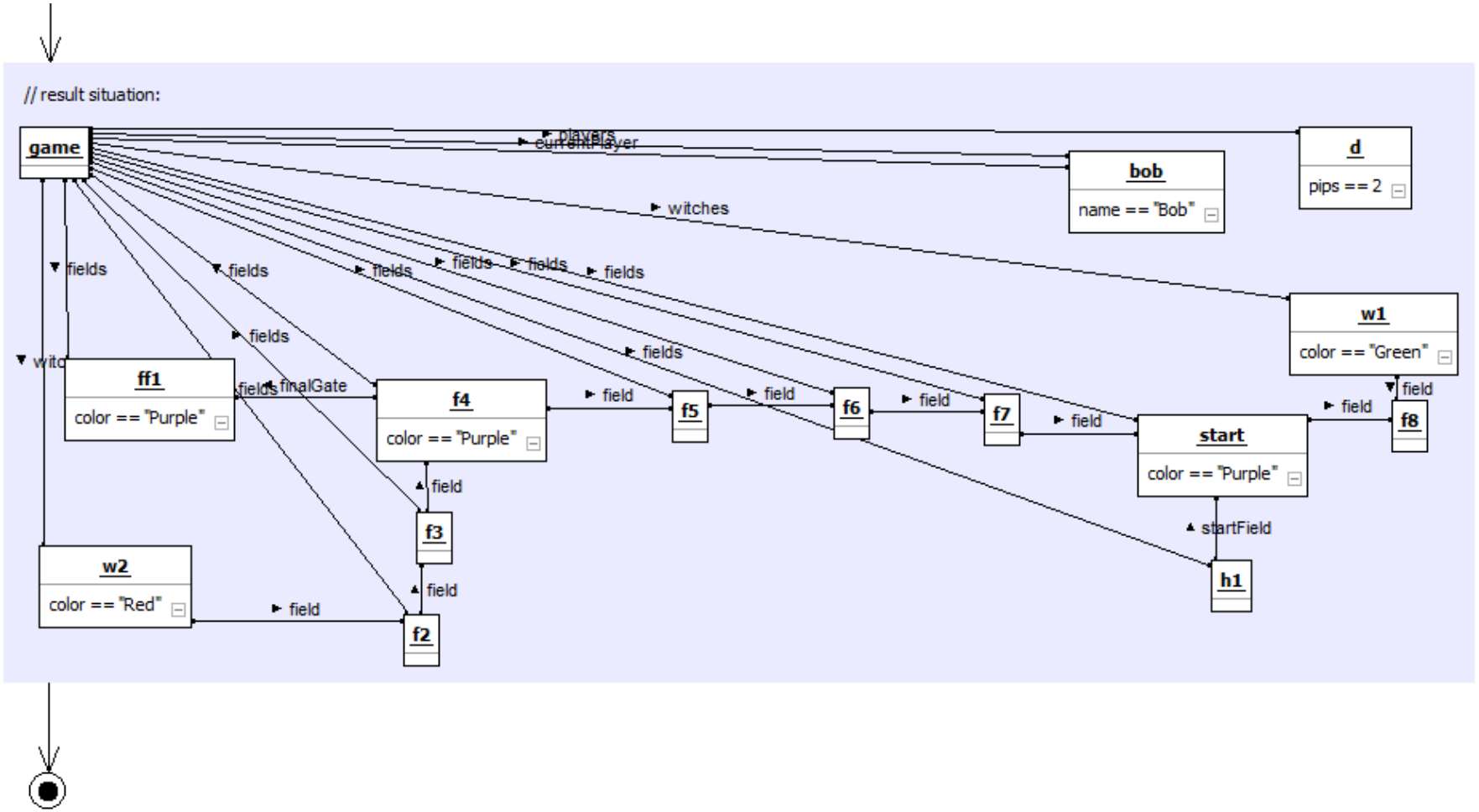
Besprechung HA5 I

- **Aufgabe 1.1:**



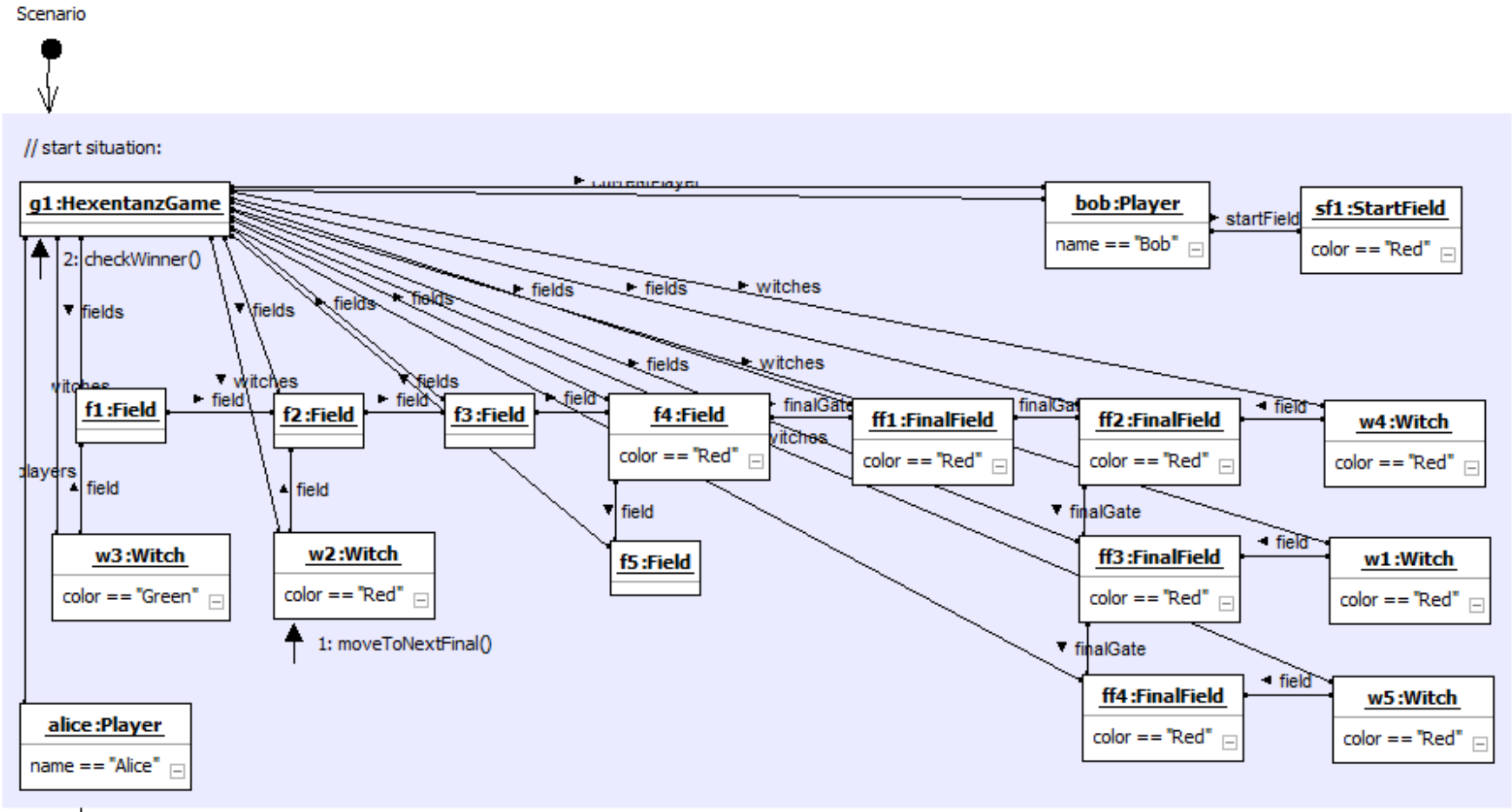
Besprechung HA5 II

- **Aufgabe 1.1:**



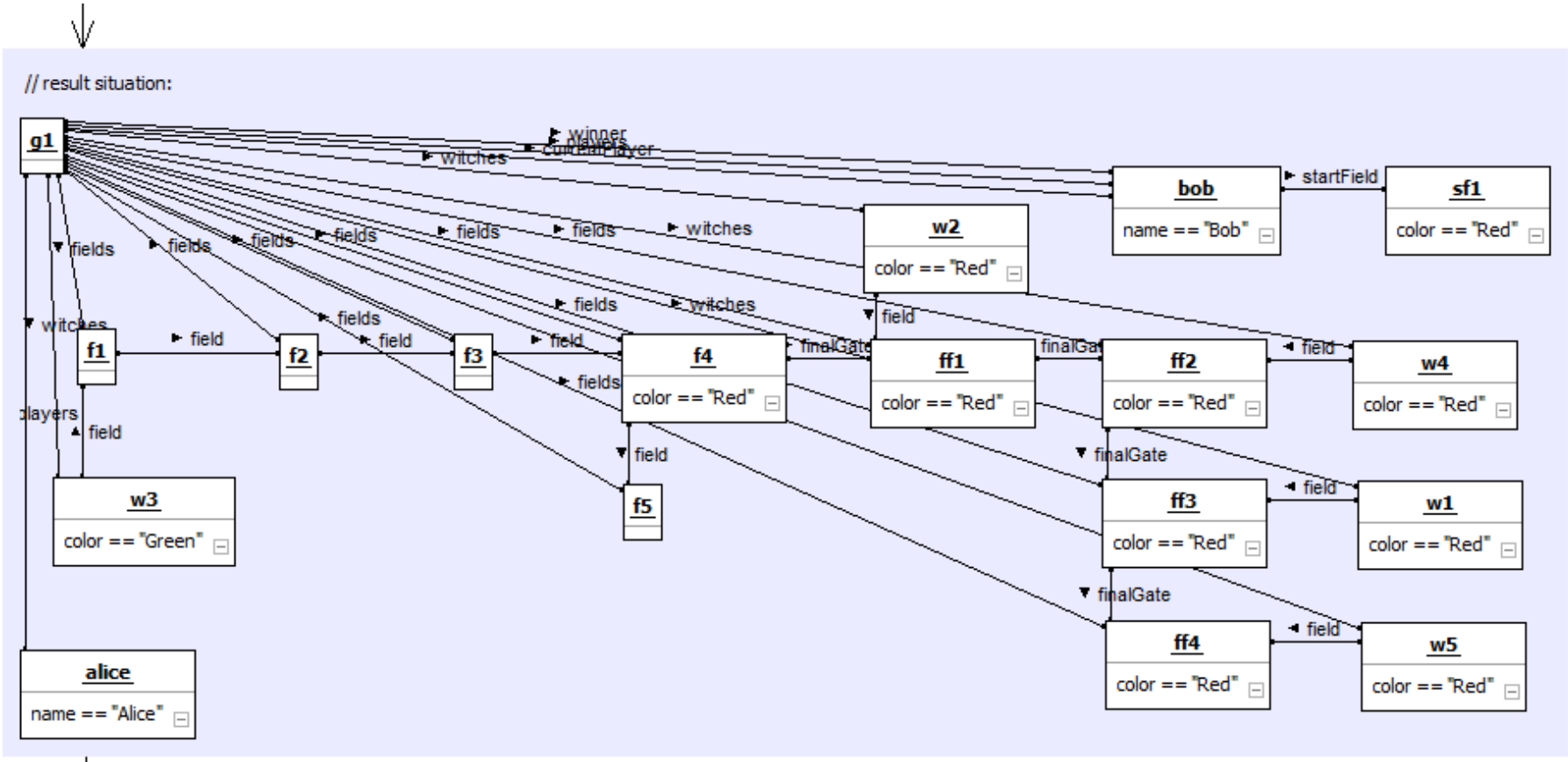
Besprechung HA5 III

- **Aufgabe 1.4:**



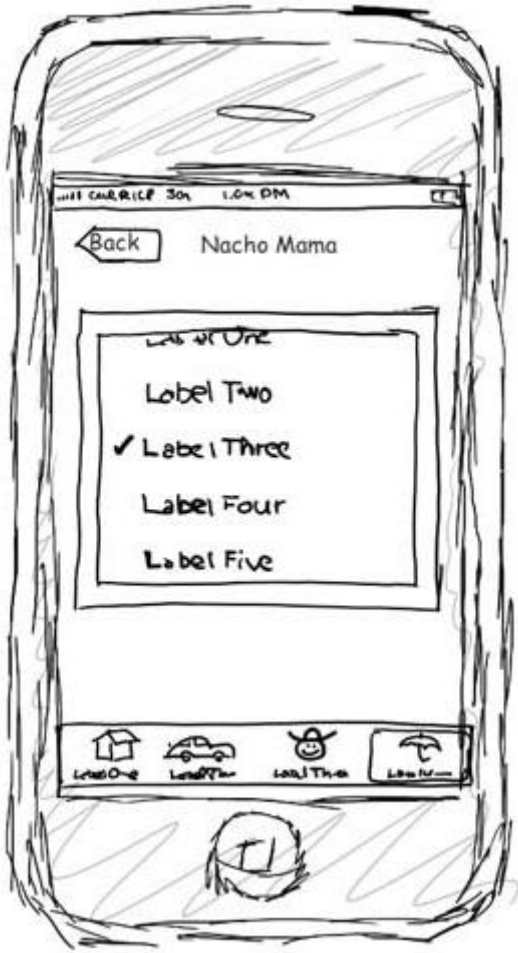
Besprechung HA5 IV

- **Aufgabe 1.4:**



Entwicklung von grafischen Oberflächen – Mock-Ups

- Bevor man anfängt zu coden: „Mock-Ups“
- In der Realität: Designer != Entwickler
- Erstellung von Mock-Ups kostet deutlich weniger Zeit
- Erleichtert die Implementierung, weil man schon weiß was herauskommen soll

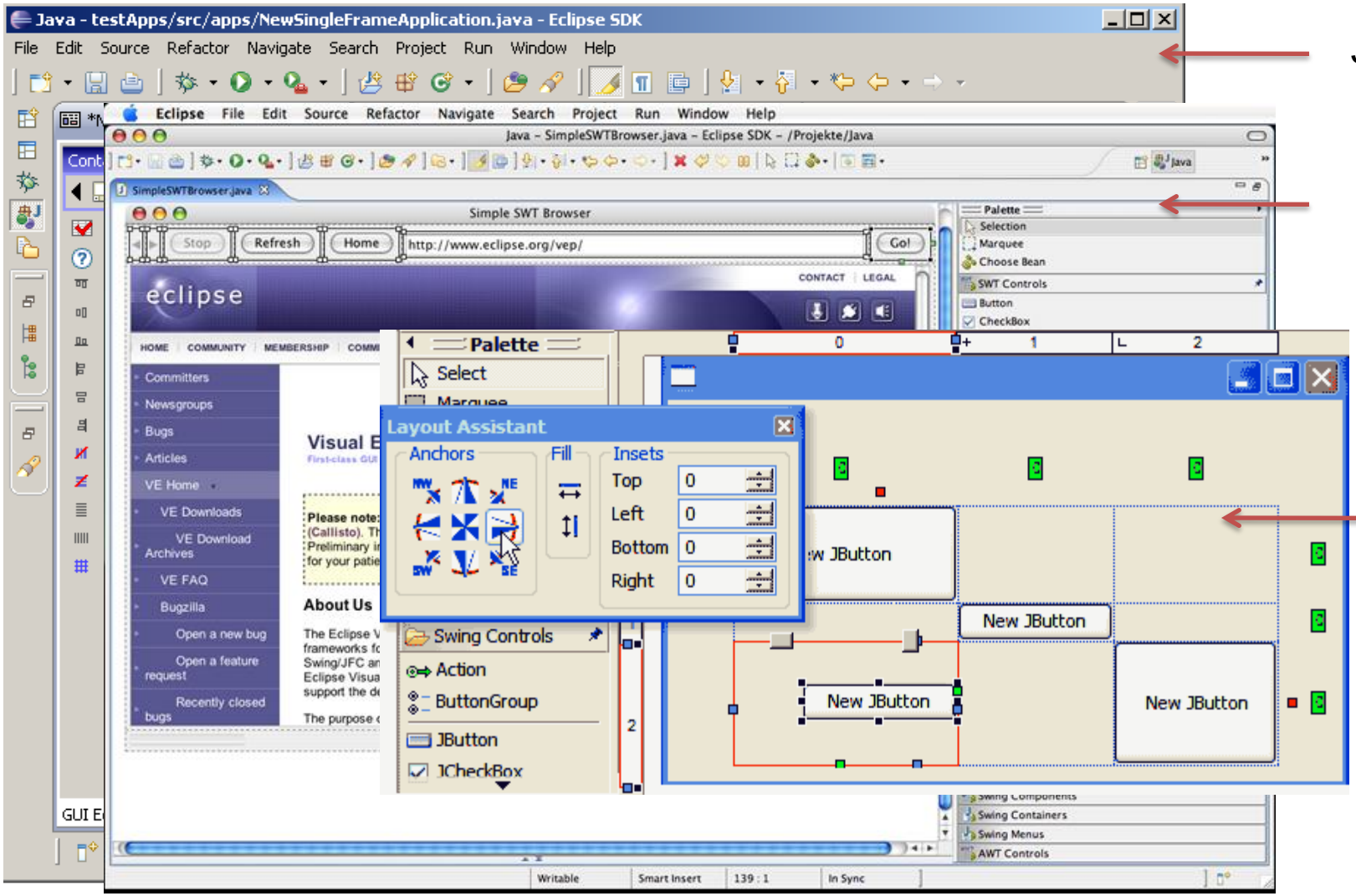


Quelle: http://media.konigi.com/notebook/iphone-mockup.jpg

Entwicklung von grafischen Oberflächen – GUI Builder I

- **Problem bei grafischen Oberflächen: Aufwendig in der Entwicklung**
- **Abhilfe sollen GUI-Builder schaffen (WYSIWYG Prinzip)**
- **Kommerzielle Tools (SWT/Swing):**
  - Jigloo, etc.
- **Freie Tools (SWT/Swing):**
  - Window Builder Pro
  - Visual Editor (Eclipse 3.2, veraltet, wird nicht mehr gepflegt)
- **Generieren (meist hässlichen) SWT/Swing Code**

Entwicklung von grafischen Oberflächen – GUI Builder II



Jigloo

Visual Editor

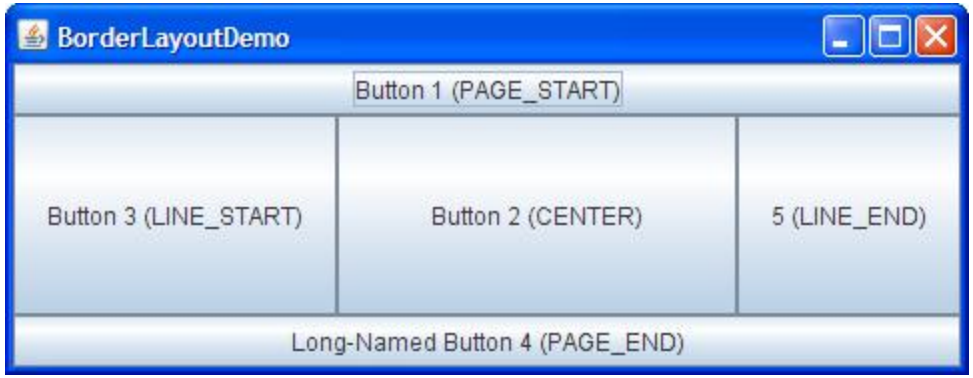
Window Builder Pro

Entwicklung von grafischen Oberflächen – Swing I

- **Swing ist seit Java 1.2 Bestandteil der Java Runtime**
- **Baut auf dem Abstract Window Toolkit (AWT) auf**
- **Swing Komponenten werden direkt von Java gerendert**
  - Funktioniert auf allen Plattformen
  - Sieht überall gleich aus
  - NICHT nativ
- **Verschiedene Look&Feels**
  - Windows
  - Linux
  - Mac
  - …

Entwicklung von grafischen Oberflächen – Swing II

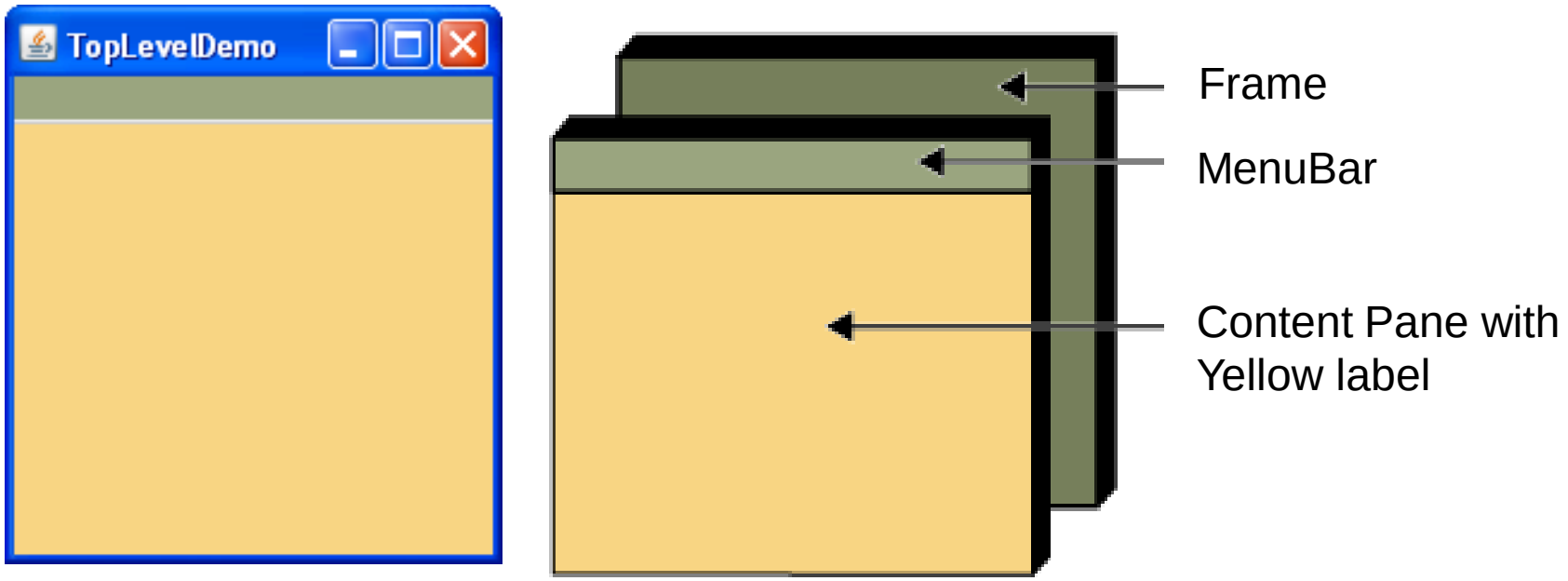
- **Referenz: http://java.sun.com/docs/books/tutorial/uiswing/**
- **Einstieg: JFrame (Top-Level Container)**
  - Repräsentiert ein Fenster
  - Kann Inhalt aufnehmen
  - Standard Layoutmanager: BorderLayout



  - jFrame.getContentPane() liefert Container

Entwicklung von grafischen Oberflächen – Swing III

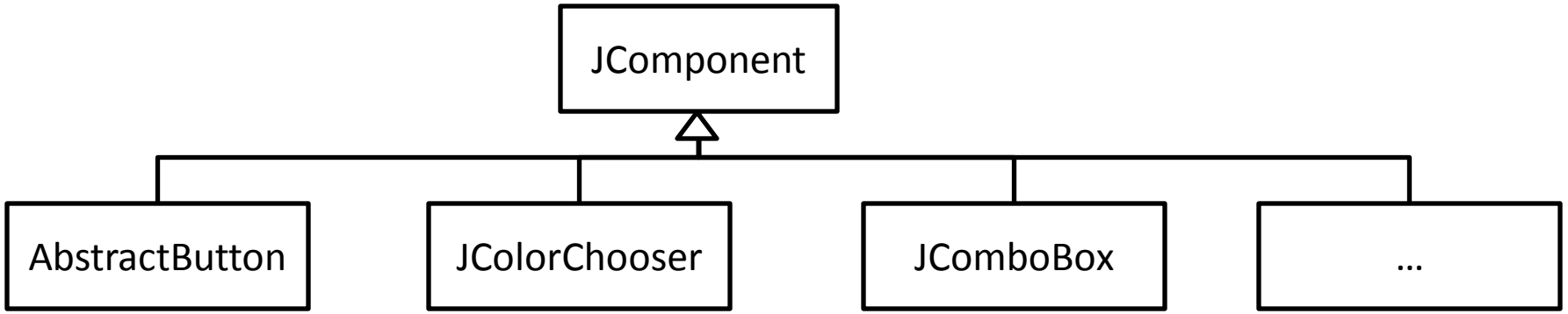
- **Aufbau eines JFrames**

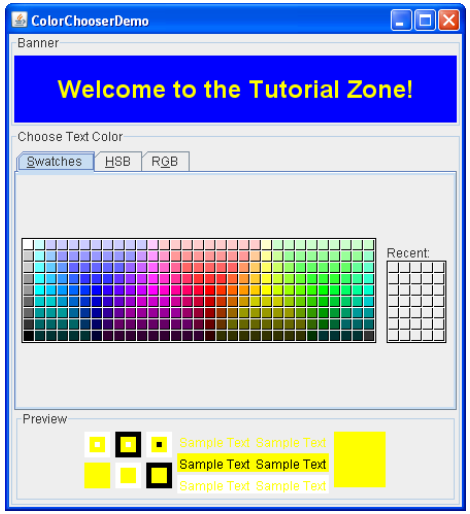


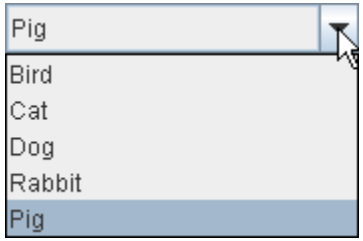
- **jFrame.getContentPane().add(yellowLabel, BorderLayout.CENTER)**

Entwicklung von grafischen Oberflächen – Swing IV

- **Alles (bis auf die Top-Level Container) leitet von JComponent ab**





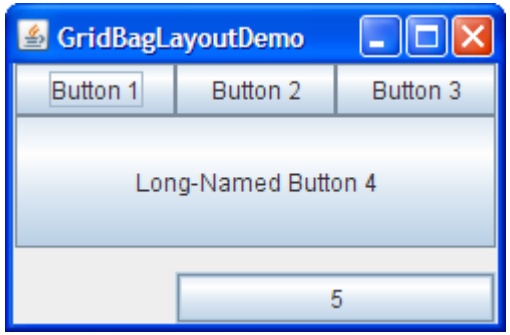


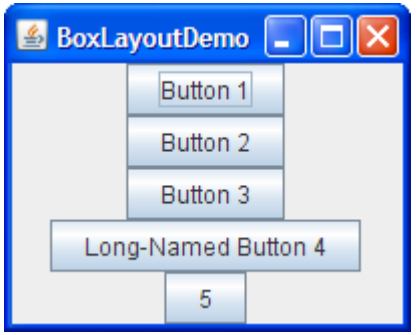
Entwicklung von grafischen Oberflächen – Swing V

- **JComponent stellt alle grundlegenden Features zur Verfügung**
  - Tooltips
  - Rahmen
  - Look&Feel
  - Layout
  - DnD
  - …

Entwicklung von grafischen Oberflächen – Swing VI

- **Absolute Positionierung vs. LayoutManager**
- **LayoutManager**
  - Positionierung
  - Verhalten bei Größenänderung des Fensters
- **Immer einen LayoutManager benutzen!**
- **Beispiele**
  - BorderLayout
  - BoxLayout
  - GridBagLayout





Entwicklung von grafischen Oberflächen – Swing VI

- **Verwendung von LayoutManagern**
  - Layout wählen (http://java.sun.com/docs/books/tutorial/uiswing/layout/visual.html)
    - BoxLayout boxLayout = new BoxLayout(jComponent, BoxLayout.PAGE_AXIS)
  - LayoutManager auf Container anwenden
    - jComponent.setLayout (boxLayout)
  - Kinder hinzufügen
    - jComponent.add(new Label(..));

Entwicklung von grafischen Oberflächen – SWT

- **Standard Widget Toolkit (SWT) http://www.eclipse.org/swt**
- **Wurde 2001 von IBM für Eclipse entwickelt**
- **NICHT Bestandteil der Java Runtime**
  - Bibliothek (inkl. nativen Bestandteilen) müssen mit ausgeliefert werden
- **Abstrahiert von nativer grafischer Benutzeroberfläche**
  - Einmal coden, überall nativ laufen lassen (theoretisch)
- **(Unter Windows) deutlich schneller als Swing**
- **Im Gegensatz zu Swing „schwergewichtig“, wegen der Verwendung von nativen Komponenten (statt sie selbst zu zeichnen)**

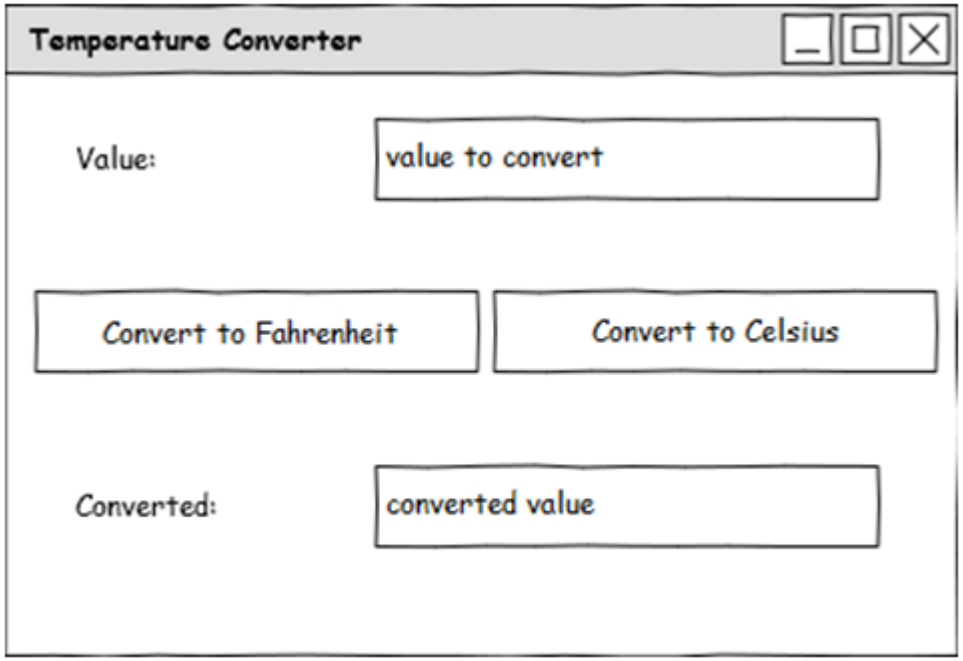
SWT – Tutorials und hilfreiche Links

- **SWT**
  - http://www.eclipse.org/articles/article.php?file=Article-Understanding-Layouts/index.html
  - http://www.eclipse.org/swt/widgets/
  - http://zetcode.com/tutorials/javaswttutorial/
  - http://www.vogella.de/articles/SWT/article.html

- **WindowBuilder Pro**
  - **Update Site: http://dl.google.com/eclipse/inst/d2wbpro/latest/3.7**
  - http://dl.google.com/eclipse/inst/d2wbpro/latest/3.6
  - http://code.google.com/intl/de-DE/javadevtools/download-wbpro.html

Entwicklung von grafischen Oberflächen

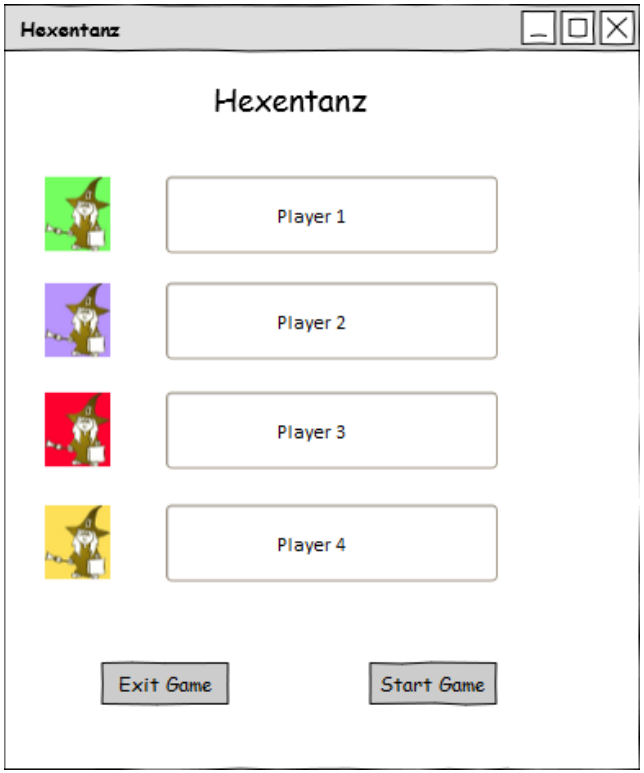
- **Demo: „Celsius – Fahrenheit Converter“**
- **Mock-Up:**



- **Umrechnungsformeln:**
  - Celsius in Fahrenheit = ((TCelsius × 9) / 5) + 32
  - Fahrenheit in Celsius = (TFahrenheit - 32) × 5 / 9

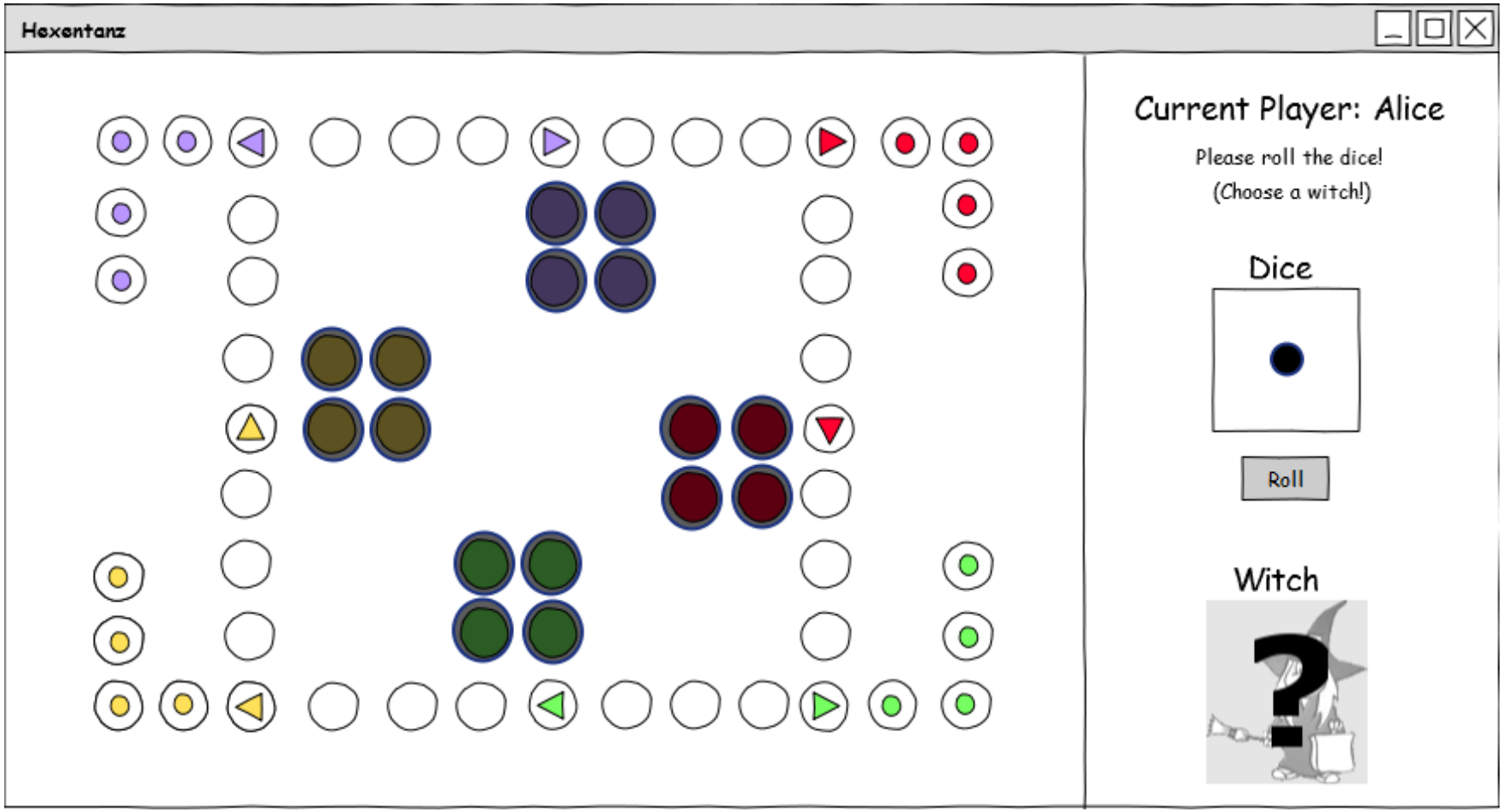
Vorschau HA6 I

- **Deadline: 15.12.2011, 23:59 Uhr**
- **Aufgabe 1.1: Hexentanz LoginScreen**



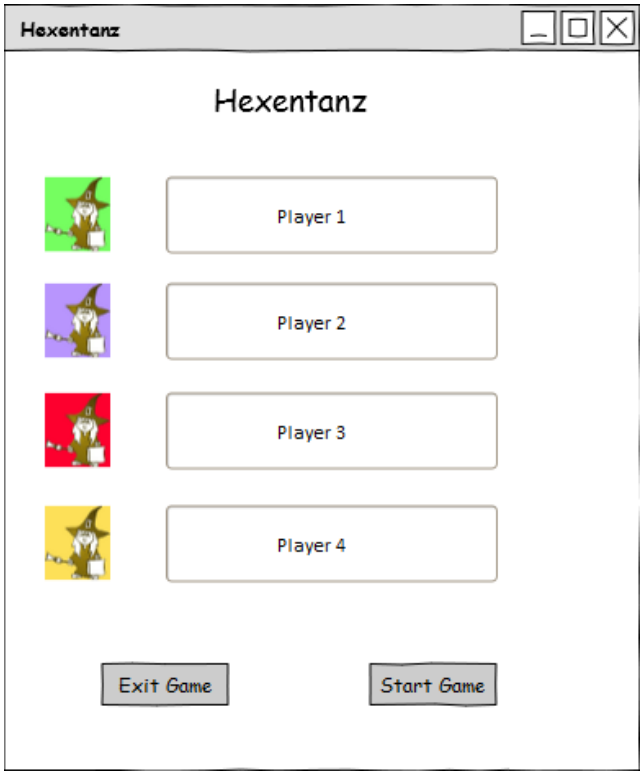
Vorschau HA6 2

- **Aufgabe 1.2: Hexentanz GameScreen**



Praktische Übung

- Entwicklung eines Hexentanz Login Screens
- Mock-Up:



- Zeigt das Ergebnis einem Betreuer

Ende

Jetzt: Betreutes Arbeiten

Ansonsten: Schönes WE!