System Design

Projekt: SprichWort

Referenznummer: 143376-LLP-1-2008-1-SI-KA2-KA2MP

Verantwortlicher Partner: IICM, TU Graz

Leiter: Univ. Doz. Dr. DI Denis Helic

1 Auswahl der technischen Infrastruktur

Aus der Analyse des Systems sind die folgenden allgemeinen Anforderungen identifiziert worden. Das System sollte:

1. frei verfügbar sein (OpenSource)
2. benutzerfreundlich sein
3. die Community als integralen Bestandteil ansehen
4. Suchmethoden Volltext-oder Keywordsuche unterstützen
5. einfache Methodiken zur Content Erstellung anbieten (z.B.: WYSIWYG Editor, simple Syntax, Eingabe-Templates und so weiter)
6. kollaboratives Arbeiten ermöglichen und damit einhergehende wichtige Methodiken unterstützen (Revisionskontrolle, Diff-Funktion)
7. ein User Management-und Control-Modul zur Verfügung stellen
8. eine flexible Art der Verlinkung anbieten.

Das neue System soll vor allem auch den Aspekt der Community mit abdecken. Die Softwarelösung kann also auch als Community-Web Content-Management-System klassifiziert werden. Der erste Schritt des Software Designs besteht damit in der Suche nach einem geeigneten Open Source Systems das den Anforderungen am besten entspricht.

Hinsichtlich dieses Kriteriums gestaltet sich die Suche schon ein wenig einfacher, denn fallen insgesamt nur mehr drei Systeme in diesen speziellen Klassifikationsbereich:

1. Weblog-Systeme
2. Community-Content-Collaboration-Management-Systeme (C3MS)
3. Wiki-Systeme

Anders als bei Sprichwort-Plattform, wo die Content-Erstellung auf einer gemeinschaftlichen Ebene stattfindet (kollaborativ), fördern Weblogs die individuelle Content-Erstellung. Weblogs arbeiten also nach dem Kommunikationsschema „One-to-Many“ wohingegen das Sprichwort-Projekt vorsieht, das Konzept „Many-to-Many“ zu unterstützen. Genau aus diesem Grund muss aber gesagt werden, dass der Weblog als geeignetes Konzept für die technische Realisierung des Sprichwort-Projektes auszuschließen ist.

Bei den Community-Content-Collaboration-Management-Systemen würde man annehmen, dass die Entscheidungsfindung abgeschlossen sein müsste, denn erfüllen diese Systeme im Grunde alle gewünschten Anforderungen. Doch musste festgestellt werden, dass es entweder nur teure, nicht frei verfügbare Lösungen in diesem Bereich gibt oder unausgereifte Portalsoftwarelösungen (siehe beispielsweise PHP-Nuke oder PostNuke) welche in technischer und sicherheitsrelevanter Hinsicht den Anforderungen nicht gerecht werden können.

Somit bleibt also die Frage offen, inwieweit Wiki-Systeme den Anforderungen gerecht werden können.

1.1 Wiki Systeme

Heutzutage existieren mehr als 300 Wiki Klone. Sie unterscheiden sich auf technischer Ebene zumeist in der zu Grunde liegenden Programmiersprache und dem verwendeten Backend für das Ablegen von Daten. Aber auch auf Anwenderebene sind nicht alle Wikis gleich, denn werden zumeist die unterschiedlichsten Zusatzfeatures mit implementiert (oder auch nicht).

Als typische Wiki-Merkmale bezeichnet man den kleinsten gemeinsamen Nenner aller Eigenschaften, Funktionen und Besonderheiten unterschiedlicher Wiki-Systeme.

Die typischen Merkmale eines Wikis können in folgenden Punkten zusammengefasst werden:
1. Wikis laden dazu ein von jedem User benutzt zu werden, sprich jeder hat die Möglichkeit Seiten frei zu erstellen und diese zu bearbeiten. Des Weiteren sollte das Editieren mit einem Standard Webbrowser möglich sein, ohne zusätzliche Software installieren zu müssen.
2. Wikis fördern eine aussagekräftige thematische Zuordnung zwischen verschiedenen Seiten durch die einfach und intuitive Art und Weise der Verlinkung.
3. Wikis sind keine sorgfältig zusammen gestellten Websites. Stattdessen zielen Wikis darauf ab, den Benutzer in einen stetigen Prozess der Seitenerstellung und Kollaboration zu involvieren, um das Aussehen und dessen Inhalte ständig zu verändern.

Eine etwas andere, aber ähnliche Zusammenfassung dessen, was die typischen Merkmale eines Wikis seien, wird im Buch „Wiki -Kooperation im Web“ von Anja Ebersbach, Markus Glaser, Richard Heigl und Alexander Warta, näher erläutert. Die Autoren beschreiben die Essenz eines Wikis hierbei zusammengefasst wie folgt:
1. *Nichtlineare Hypertextstruktur:* Wikis ermöglichen die Entstehung assoziativer Hypertexte mit nicht-linearen Navigationsstrukturen.
2. *Einfacher und weitgehender Zugriff:* Wiki-Systeme reduzieren das technische Vorwissen im Bezug auf deren Benutzung auf ein Minimum.
3. *Keine Client-Software:* Wiki-Systeme erlauben das Lesen, Schreiben, Navigieren und das Editieren einer Seite ohne Installation zusätzlicher Client-Software.
4. *Soziale Prozesse im Vordergrund:* In Wiki-Systemen steht das Diskutieren und Philosophieren über Beiträge mehr im Vordergrund als die Erstellung derer.

1.2 Wiki Technik

Auch wenn sich die einzelnen Wiki-Klone funktional teilweise beträchtlich voneinander unterscheiden, so unterliegen diese vom technischen Standpunkt aus betrachtet immer demselben Prinzip.

Wikis sind im Grunde nichts anderes als herkömmliche Client-Server Anwendungen, welche auf der Basis des World Wide Web agieren. So beruht jegliche Kommunikation zwischen Benutzer und System auf dem HTTP-Protokoll mit den dazugehörigen Request Methoden

GET und POST, wobei die Methode GET dafür verwendet wird, um Daten vom Server zu laden und die Methode POST, um Daten zum Server zu schicken.

Möchte man nun als herkömmlicher Benutzer einen Eintrag auf einem Wiki-System lesen, werden im Hintergrund folgende Operationen durchgeführt:

1. Im ersten Schritt wird über die Methode GET ein Request an den Server abgesetzt, welcher über die URL erkennt, um welche Seite der Benutzer gebeten hat.
2. Im zweiten Schritt auch Verarbeitungsschritt genannt, generiert das Wiki-System (jetzt serverseitig) aus dem so genannten Wiki-Text HTML Code.
3. Im dritten Schritt, wird der zuvor erzeugte HTML Code in eine vom Wiki-System zur Verfügung gestellten Vorlage, auch Template genannt, eingebettet um schlussendlich die eigentlich Wiki-Seite zu generieren.
4. Im vierten und letzten Schritt wird der zuvor produzierte Code zurück an den Client gesendet, welcher dort vom Browser interpretiert und schließlich angezeigt wird.

1.3 Charakteristische Wiki Funktionen

Wie die typischen Wiki-Merkmale unabhängig von der Implementierung sind, so verhält sich dies auch mit den so genannten charakteristischen Wiki-Funktionen. Beispiele hierfür der so genannte Wiki-Text, eine einfache Textaufzeichnungssprache um Wiki-Beiträge zu erstellen, die enthaltene „Edit“ Funktion, mit der Beiträge erstellt und bearbeitet werden können, die Überblicksseiten „History“ und „Recent Changes“ sowie die Suchfunktion oder die so genannte „Sandbox“, um nur einige zu nennen.

1.3.1 Wiki-Syntax

Wie schon in Anforderungsspezifikation beschrieben, sollte die Methodik des Beitragerstellens so einfach wie möglich sein. Wiki-Systeme implementieren deshalb häufig eine einfach zu lernende Sprache, gerne auch als „Wiki-Text“ bezeichnet, welche rudimentäre Elemente moderner Textaufzeichnungssprachen in ihrer einfachsten Form unterstützt.

Wiki-Text lehnt sich dabei sehr stark an die Möglichkeiten, die HTML bietet, an. Das hat den einfachen Grund, dass Wiki-Sprach-Elemente, die so genannten Wiki-Tags, vom Wiki-System, vor der Ausgabe auf das Browserfenster in HTML konvertiert werden. Das wiederum bringt natürlich den Vorteil mit sich, dass keine zusätzliche Software installiert werden muss um einen Wiki-Beitrag zu erstellen.

Die Editierfunktion, auch als „Editpage“ bekannt, beinhaltet eine Reihe von Elementen mit denen es möglich ist eine Seite einfach zu bearbeiten oder zu erstellen. So finden sich auf dieser Seite zumeist ein Eingabefeld und eine Menüleiste mit deren Funktionsknöpfen es möglich ist, Wiki-Code auch ohne syntaktisches Vorwissen zu erzeugen. Aber auch eine „Preview-Funktion“ ist zumeist vorhanden sowie ein „Save“-Button um Beiträge abzuspeichern. Desweiteren bieten viele Wiki Systeme einen WYSIWYG Editor an.

1.3.2 Suchfunktion

Ein wichtiges Feature, das in jedem Datenbanksystem zu finden ist, ist die Such-Funktionen. So ist diese wichtige Zugangsmethodik auch integraler Bestandteil eines jeden Wiki-Systems.

Die meisten Wiki-Systeme bieten zusätzlich zu herkömmlichen Methodiken, wie Volltext- und Schlüsselwortsuche noch die Möglichkeit an, direkt zu einem Beitrag zu „springen“, ähnlich einem Karteikartensystem.

1.3.3 Verlinkung

Neben der Möglichkeit Beiträge frei erstellen zu können und diese zu editieren ist wohl die Option Beiträge miteinander zu verlinken, eine der wichtigsten Funktionen, die ein Wiki zu bieten hat.

So kann in einem Wiki-System jeder Beitrag auf einen anderen verweisen und eine neue Hyperlinkstruktur bilden. Die Verlinkung eines Artikels erfolgt hierbei zumeist über dessen Namen und verlangt keine Referenzierung über dessen URL. Zudem unterstützen die meisten Wikis das Generieren von so genannten WikiWords, oder auch CamelCases genannt. Hierbei werden Phrasen, welche im eigentlichen Sprachgebrauch auseinander geschrieben werden, zu einem Wort zusammengefasst. Die Phrase „DasIstEinWikiBeitrag“ wäre hierfür ein gutes Beispiel, um zu veranschaulichen was konkret damit gemeint ist.

Darüber hinaus bieten die meisten Wikis so genannte Übersichtsfelder an, die es ermöglichen festzustellen, welche Links auf eine soeben betrachtete Seite verweisen beziehungsweise davon weg zeigen.

Hyperlinks, welche nicht existieren, werden zudem als solche erkennbar gemacht. Zumeist rot unterlegt, offerieren diese dem Benutzer die Möglichkeit, einen Beitrag zu diesem Link/Thema zu erstellen.

1.3.4 Recent Changes

Eine wichtige und beliebe Anlaufstelle für den Wiki Benutzer ist die Seite „Recent Changes“, auf der, wie der Name schon sagt, der Benutzer eine Liste der letzten Änderungen im Wiki-System vorfindet. Hierbei ist zu beachten, dass diese Liste sich nicht nur Änderungen einer Seite bezieht wie die Funktion „History“, sondern alle in einem Wiki verfügbaren Seiten. Da diese Liste unüberschaubare Ausmaße annehmen würde, wäre sie nicht beschränkt, wird zumeist ein definierter Zeitraum herangezogen, um eine obere Schranke einzuführen und diese übersichtlicher zu gestalten. Neben „normalen“ Wiki-Benutzern, welche diese Überblicksseite zumeist dafür verwenden um über die neuesten Geschehnisse informiert zu werden, ist diese außerdem ein unabdingbares Administratorenwerkzeug, wenn es darum geht Vandalismus entgegen zu wirken.

1.3.5 Versionskontrolle

Die wohl mit Abstand wichtigste Wiki-Funktion, wenn es darum geht Änderungen an einer Wiki-Seite zuerkennen, ist die so genannte „History“-Seite. Auf ihr werden, ähnlich der „Recent Changes“-Seite, Änderungen, chronologisch sortiert, dem Benutzer dargeboten.

Zudem stehen dem User zwei wichtige Funktionen zur Verfügung: Die Revisions-und die Diff-Funktion.

So ist es also dem Benutzer möglich, nicht nur Änderungen festzustellen, sondern die-se wenn gewünscht, auch rückgängig zu machen, wobei die Versionisierung der Beiträge zumeist bis zu deren Erstellungsdatum zurückreicht.

Auch aus dieser einfachen Analyse ist es sofort sichtbar dass ein Wiki System eine gute technische Infrastruktur für das Sprichwort Projekt darstellt.

1.4 Vergleich der Wiki Systeme

Im Rahmen dieses Projektes wurden fünf bekannteste Open Source Wiki Systeme untersucht und bezüglich der Anforderungen des Projektes überprüft. Die Tabelle unten zeigt die Kriterien die zur Auswahl des JSPWiki Systems verwendet worden waren.

Kriterien	*Media Wiki*	*Flex Wiki*	*JSP Wiki*	*TWiki*	*Doku Wiki*
Security Features	o	++	+	++	+
Userverwaltung	+	++	++	+	++
Strukturierung	++	o	+	++	++
Suche	++	+	++	-	+
Kommentare, Forumen, Blogs	o	-	+	+	o
XHTML,CSS, RSS, PDF	+	-	++	+	+
Media (Video, Audio, Flash)	++	-	++	++	o
Conflict Management	++	o	o	++	o
Hersteller Support	o	-	++	o	++
Datastorage	o	+	++	o	o
Erweiterbarkeit	o	++	++	+	o
Code&Programmiersprache	+	+	++	o	o
Printer Friendly	++	-	++	+	++
Syntax	+	-	+	+	+
Usability	++	-	+	-	+

2 Design des Gesamtsystems

2.1 JSPWiki 2.8.0

Das JSPWiki folgt dem Model-View-Controller (MVC) Architektur Muster.

Das Model wird durch die `WikiEngine` und zusätzlichen Hilfsobjekten verwaltet. Durch verschiedene Manager Objekte wird zum Beispiel der Page Speicher verwaltet. Jeglicher Zugriff auf Wiki Services erfolgt über die `WikiEngine`.

Die View wird durch mehrere JSP Seiten umgesetzt. Diese befinden sich im `/template` Ordner und können, je nach Bedürfnis, angepasst werden.

Der Controller basiert ebenso auf JSP Seiten. Diese befinden sich aber im Top Level Directory. Der Controller beinhaltet die Logik des Frameworks, wie zum Beispiel das Editieren oder Löschen von Beiträgen. Der Vorteil durch die Verwendung von JSP Seiten im Gegensatz zu Servlets ist die große Flexibilität. So können diese Seiten modifiziert werden, ohne den Code neu zu kompilieren. Noch dazu sind JSP Seiten eigentlich Servlets.

2.2 System Design

Das Design des Gesamtsystems besteht im Wesentlichen von der Installation und Konfiguration des JSPWiki Systems sowie dessen Anpassung an die Anforderungen der SprichWort-Plattform. Das Design der einzelnen Komponenten des Systems wird in darauffolgenden Kapiteln näher erläutert:

1. Kapitel 3 beschreibt das Design der SprichWort-Datenbank
2. Kapitel 4 beschreibt das Design der SprichWort-Übungen

In diesem Kapitel wird nur kurz das Design des Gesamtsystems beschrieben, das im Wesentlichen aus einer einfachen Verlinkung der anderen Komponenten besteht. Desweiteren wird nur kurz angedeutet wie das Gesamtsystem die Community Komponente abdeckt da ein Wiki System per Definition ein Community System ist.

Schlussendlich wird kurz die Benutzerverwaltung sowie Datenverwaltung inklusive Versionskontrolle beschrieben.

2.3 Verlinkung der Komponenten

Eine einfache Menü Struktur bietet einen schnellen Zugang zu allen Bereichen der Sprichwort-Plattform (siehe dazu Abbildung 1). Dabei werden in jeweiligen Teilen weitere Strukturierungen vorgenommen wie etwa die dreistufige Struktur der Sprichwörter oder Unterteilung der Sprichwörter nach Sprachen (siehe dazu Abbildung 2 bzw. Abbildung 3).

Abbildung 1: Menü Struktur

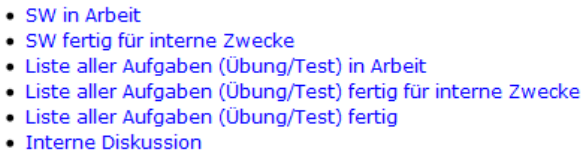


Abbildung 2: Dreistufige Strukturierung der Beiträge

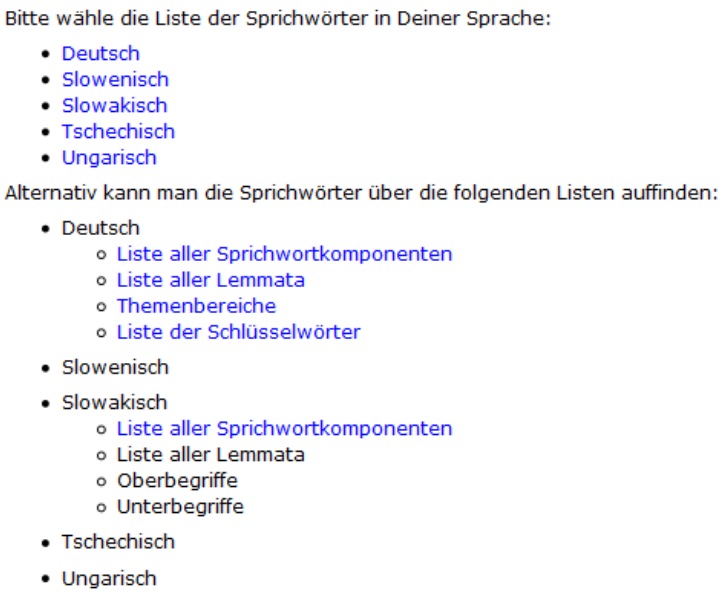


Abbildung 3 Strukturierung der Sprichwörter

2.4 Community Aspekte

Wie oben schon erwähnt ein Wiki System basiert auf der kollaborativen Arbeit mehrerer Benutzer bei der Erstellung der gemeinsamen Inhalte. Daher ist ein Wiki System per Definition ein Community System, das die Entstehung einer Community bestens unterstützt. Desweiteren werden von JSPWiki System Module wie Diskussionsforum, Annotieren und Kommentieren (siehe dazu Abbildung 4) sowie Möglichkeiten zur Erweiterung der Export-Formate von den Inhalten zur Verfügung gestellt. Der einzige Community-Aspekt der in der Anforderungsspezifikation definiert worden ist und nicht vom Haus aus im JSPWiki System

unterstützt wird ist das Tagging Modul. Deswegen sollen bei der Sprichwort-Plattform technische Gegebenheiten geschaffen werden, die es ermöglichen externe Tagging Systeme, wie zum Beispiel deliciuos.com einhzubinden. Nun dies geschieht durch einfache Modellierung von URLs von den Beiträgen im JSPWiki System. Jeder Beitrag erhält eine eindeutige URL, die dann in weiterer Folge mit Systemen wie delicious.com getaggt werden kann.

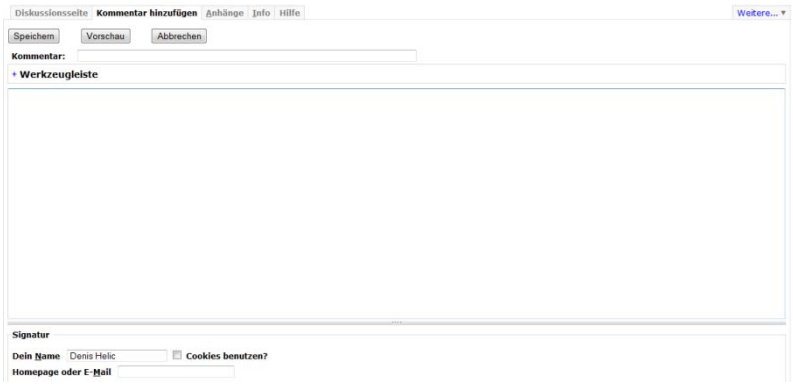


Abbildung 4: Kommentieren der Beiträge

2.5 Benutzerverwaltung

Die Benutzer gehören immer einer Benutzergruppe. Es gibt im System 4 Benutzergruppen, jeweils mit vordefinierten Datenzugangsrechten:

1. Administratoren, die alle Beiträge editieren können sowie andere Benutzergruppen modifizieren können
2. Editoren, die jeweils für einen Bereich mit dazugehörenden Inhalten zuständig sind, wie zum Beispiel die Gruppe der Editoren für die deutschen, slowenischen, usw. Sprichwörter.
3. Studenten, die alle öffentlichen Beiträge ansehen können, die Übungen lösen können und im Community Bereich alle Beiträge editieren können.
4. Anonyme Benutzer, die alle öffentlichen Beiträge ansehen können sowie alle Beiträge im Community Bereich editieren können.

Die Benutzerverwaltung soll mittels eines Benutzerverwaltungtools durchgeführt werden (siehe dazu Abbildung 5).

Abbildung 5: Benutzerverwaltung

2.6 Datenverwaltung und Versionskontrolle

Datenverwaltung passiert mittels Server File Systems. Da es sich bei den Inhalten des Sprichwort-Projektes hauptsächlich um Text-Dokumente handelt, die zwar eine gewisse interne Struktur aufweisen (wobei diese Struktur sehr stark von einem Beitrag zum anderen variieren kann), ist ein relationales Datenbanksystem für die Speicherung von solchen Daten nicht geeignet.

Die Versionskontrolle im File System erfolgt typischerweise mittels eines Revision Control System wie RCS oder Subversion.

3 SprichWort-Datenbank

Die SprichWort-Datenbank Komponente ermöglicht es Editoren die Inhalte mittels eines WYSIWYG Editors einzufügen. Dabei wird das Datenbeschreibungsmodel als Basis für eine Reihe von Vorlagen für verschieden Sprachen verwendet. Das Erfassen eines neuen Sprichwortes bedeutet deswegen immer zuerst eine Auswahl der Vorlage zu treffen und in weiterer Folge die Teile der Vorlage mit den aktuellen Daten zu befüllen (siehe dazu Abbildungen 6 und 7).

Abbildung 6: Bearbeiten der Vorlage

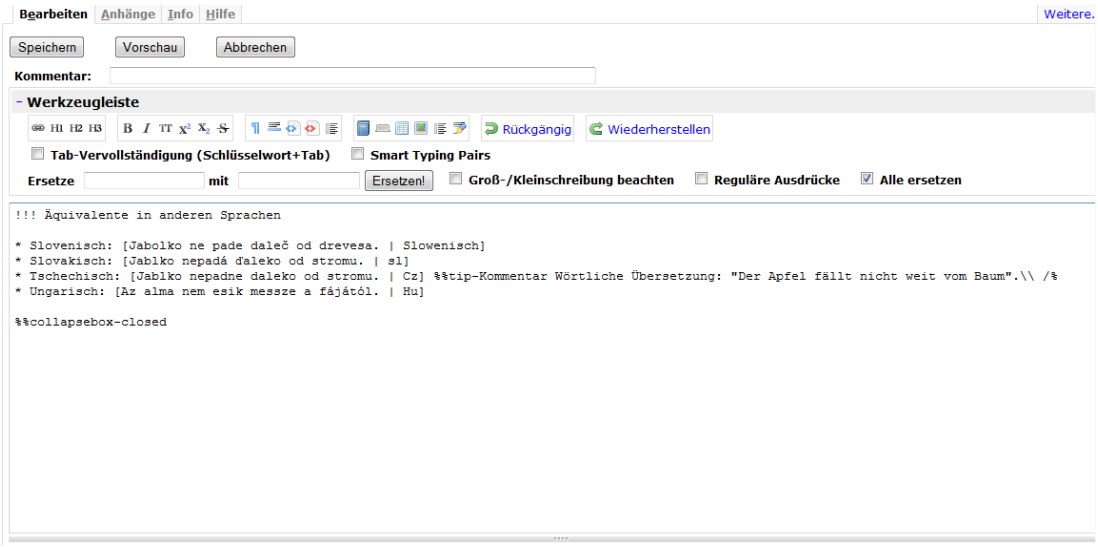


Abbildung 7: WYSIWYG Editor

Die weiteren Angaben zum Editieren und Erfassen der Daten befinden sich in verschiedenen Hilfe-Dateien und Anleitung die im System zur Verfügung gestellt werden sollen. Diese erklären in Details die Wiki Syntax, die Text-Formatierung Befehle sowie Möglichkeiten der automatischen Verlinkung. Da diese Dateien groß im Umfang sind werden sie den Rahmen dieses Dokumentes sprengen und werden daher nur online zur Verfügung gestellt.

4 SprichWort-Übungen

4.1 Plugins

Das JSPWiki unterstützt die Einbindung von Plugins in Wiki Seiten. Diese Plugins automatisieren Aktionen auf Wiki Seiten, wie zum Beispiel: das Search Plugin.
Die formale Definition eines Plugins zum Einfügen auf einer Seite lautet wie folgt:

```
[{INSERT <plugin class> WHERE <param1=value1>,<param2=value2>,...}]
```

Diese Definition kann aber auch verkürzt werden auf:

```
[{<plugin class> <param1=value1>}]
```

Ein Plugin setzt sich aus drei Elementen zusammen:

- die Plugin Klasse
- die Plugin Parameter
- der Plugin Body

Durch die Implementation des Interfaces *`com.ecyrd.jspwiki.plugin.WikiPlugin`* können eigene Plugins erzeugt werden. Wird ein Plugin in einer Seite eingebunden, so wird dessen „execute“-Methode aufgerufen und die Darstellung und Funktionalität der Wiki Seite entsprechend modifiziert.

4.2 Dynamic styles

Dynamische Styles ermöglichen ein erweitertes Design für Wiki Seiten. Diese Styles können in einer Wiki Seite ähnlich wie Plugins definiert werden und sind eine Kombination aus JavaScript und CSS. Ein möglicher Style ist zum Beispiel der slimbox Style. Durch ihn wird es ermöglicht ein Bild oder ein IFrame in Form der bekannten Lightbox anzuzeigen.

Die Anwendung eines solchen Styles wird mit folgender Syntax durchgeführt:

```
%%<style name>
```

Styles sollten nur die Präsentation der Seite beeinflussen.

4.3 Filters

Auch Filter werden durch das JSPWiki unterstützt. Diese dienen dazu um den Ein- und Ausgabeprozess zu manipulieren. Filter können zu vier verschiedenen Zeitpunkten aufgerufen werden:

- vor der Konvertierung von Wiki Syntax auf HTML Code (`preTranslate`)
- nach der Konvertierung von Wiki Syntax auf HTML Code (HTTP Request – SEND wurde noch nicht abgesetzt) (`postTranslate`)
- vor der Speicherung der Wiki Seite im Datastorage-Modul (`postSave`)
- nach der Speicherung der Wiki Seite im Datastorage-Modul (`postSave`)

Wird eine Methode des Filters zu einem der vier Zeitpunkte aufgerufen, so werden der `WikiContext` und der `Content` übergeben. Der Filter kann dadurch Informationen über den Kontext beziehen oder diesen befüllen. Der `Content` stellt den Inhalt der Wiki Seite zum entsprechenden Prozessschritt dar.

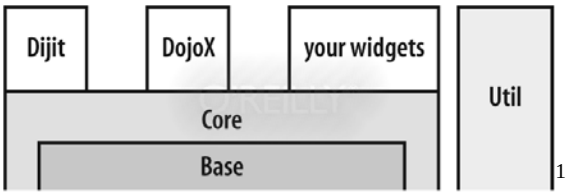
Durch die Implementation des Interfaces *`com.ecyrd.jspwiki.filters.PageFilter`* können auch hier wieder eigene Filter erzeugt werden.

4.4 Dojo Toolkit 1.3.1

Für den Einsatz von JavaScript wurde das Dojo Toolkit gewählt. Es ermöglicht nicht nur die dynamische Änderungen von Website Inhalten, sondern bietet auch die Möglichkeit zu Ajax. Dojo abstrahiert hierzu über einen Wrapper die XMLHttpRequest API und kapselt damit verschiedene Implementationen. Es erleichtert somit die Entwicklung von Ajax Code um die Cross-Browser Fähigkeit zu bewahren.

4.4.1 Architektur

Die Dojo Toolkit Architektur besteht aus mehreren Komponenten. Und kann durch eigene Widget Implementation erweitert werden.



[1]

Komponenten Base und Core

Die Komponente Base besteht aus lediglich aus der dojo.js Datei. Dieser Kernel beinhaltet in einer ultra-kompakten Form, die wichtigsten Features von Dojo. Zu diesen Funktionalitäten gehören die CSS-basierten Queries, Event Handling, Ajax oder Klassenbasierte Programmierunterstützung.
Die Core Komponente baut auf dem Kernel auf und stellt Funkionalitäten wie erweiterte Animationen, I/O und Drag and Drop zur Verfügung.

Komponente Dijit

Dijit repräsentiert eine große UI Bibliothek. Die Widgets von Dijit ermöglichen die Verwendung von Themes und Accesibility Standards und sind einfach zu verwenden. Ebenso besteht die Möglichkeit eigene Widgets zu erzeugen.

Komponente DojoX

[1] http://www.oreilly.de/catalog/9780596516482/toc.html

DojoX ist eine Sammlung von Subprojekten, die teilweise noch in einer experimentellen Phase sind. Darunter befinden sich Möglichkeiten für Grid-Widgets, bessere Animationen und Diagramme.

Komponente Util

Util beinhaltet Werkzeuge wie ein Unit-Test Framework oder verschiedene Build Tools.

4.4.2 Grund der Verwendung

JavaScript an sich ist essentiell um Interaktivität in die Aufgaben zu bringen. Dojo wurde deshalb gewählt, weil es eine weit verbreitete JavaScript Bibliothek ist und eine große Auswahl an Widgets liefert.
Ebenso unterstützt es hervorragend Drag und Drop von Objekten. Das ist ein sehr großer Vorteil, da viele Zuordnungsaufgaben angefordert werden.
Zusätzlich unterstützt das Toolkit eine Art objektorientiertes Programmieren.

4.5 Abstratkionsschichten

Als die Implementation begonnen wurde, war die Aufgabe ein Template System für Aufgaben in Form von Dojo Widgets zu erstellen. Instruktoren hätten über diese Abstraktionsschicht HTML Seiten erzeugt und diese Aufgaben Widgets eingebunden. Nach kurzer Zeit zeigte sich aber die Komplexität dieser Erzeugung. Instruktoren bräuchten dafür HTML und Javascript Hintergrundwissen, da viele Einzelkonfigurationen vorgenommen werden mussten.
Aus diesem Grund wurde eine weitere Abstraktionsschicht eingeführt. Diese Schicht beruht auf der Verwendung von den zuvor beschriebenen Mechanismen des Wiki Frameworks.

1. Abstraktionsschicht

```
<div dojoType="dispue.AssessmentController" id="c1"></div>

<p><b>Aufgabe: </b>Markieren sie die Fehler in den Sprichw&ouml;rtern. <br>
<hr>
<div class="content">
    Die Wasserballer schafften mit psychologischer Hilfe die Olympia-Qualifikation.
    Handball-Bundestrainer Heiner Brand gilt als Fan der Sportpsychologie.
    Als seine Mannschaft Europameister wurde, ließ er vor jedem Spiel im
    Besprechungszimmer Sprüche wie diesen aufhängen: » Wenn es
    eine<div dojoType="dispue.HotspotMarking" controller="c1"
        params="{   'params': [{
                    'content':'n',
                    'isCorrect':true,
                    'hint':'Hmm. Das passt doch oder?'},
                {   'content':'m',
                    'isCorrect':false,
                    'hint':'Komisch oder?'}
                ]}"
        isRandomized=true></div>&nbspGlauben gibt, der
        <div dojoType="dispue.HotspotMarking" controller="c1"
        params="{   'params': [{
                    'content':'Berge',
                    'isCorrect':true,
                    'hint':'Das passt hier her!'},
                {   'content':'Wolkenkratzer',
                    'isCorrect':false,
                    'hint':'Das stimmt nicht ganz so!'}
                ]}"
        isRandomized=true>
        </div>&nbspversetzen kann, dann ist es der Glaube an die eigene Stärke.
</div>
```

2. Abstraktionsschicht

In dieser Abstraktionsschicht kann der Instruktor auf einer Wiki Seite ein Aufgaben Plugin definieren. Durch diese Abstraktion wird die Verwendung von JavaScript und HTML zur Gänze gekapselt.

```
Ein alt bekanntes Sprichwort ist:
[{SpHotspotKennzeichnung
|
inhalt='Eigener Herd ist goldes wert.'
korrekt=true
|
inhalt='Eigener Ofen ist silber wert.'
korrekt=false
|
}]
```

4.6 Wiki Seite als Container für Test / Übungen und Aufgaben

Das Konzept dieser zweiten Abstraktionsschicht ist nicht nur für Aufgaben, sondern auch für Test und Übungen gedacht.
Eine Wiki Seite bildet hier, je nach Konfiguration, eine Aufgabe, einen Test oder eine Übung. Die Elemente dieser verschiedenen Typen werden über Plugins hinzugefügt.

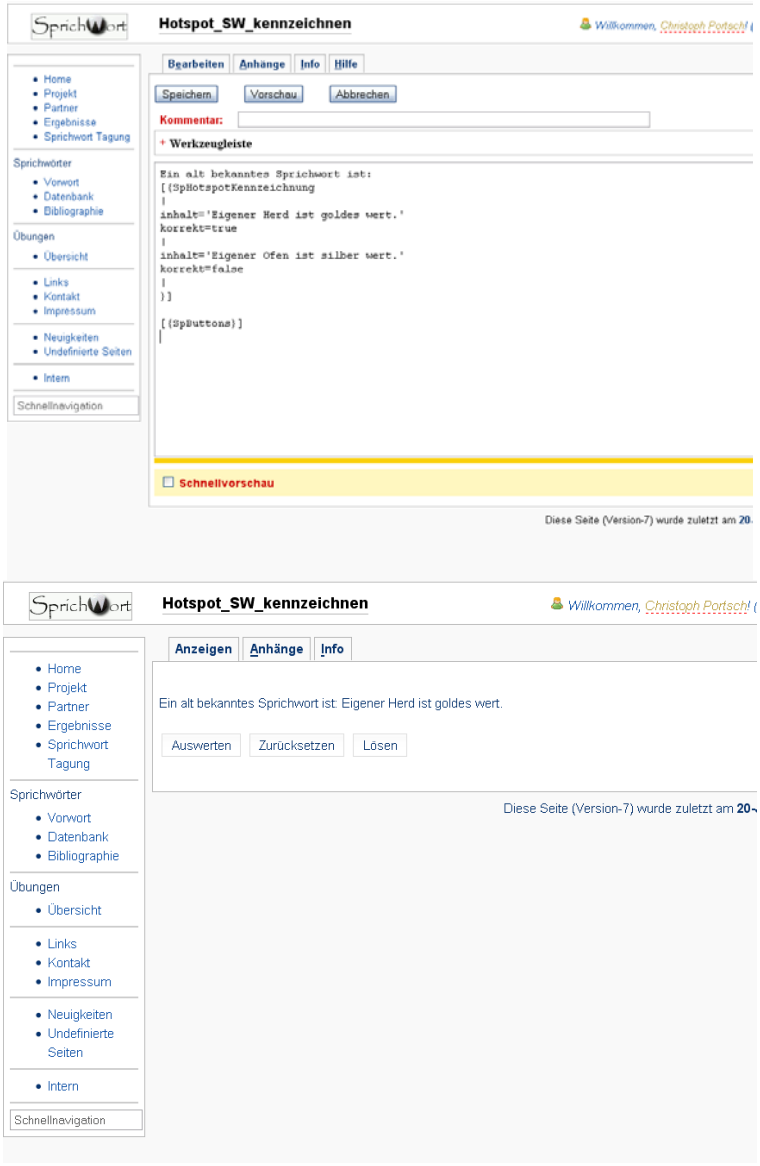
Bereits implementierte Plugins:

- `SpPhase`
- `SpAnleitung`
- `SpButtons`
- `SpHotspotKorrektur`
- `SpHotspotKennzeichnung`
- `SpHotspotMCQ`
- `SpHotspotLuecke`
- `SpMCQ`
- `SpMemory`
- `SpDndSalat`
- `SpDndListe`
- `SpDndQuelle`
- `SpDndZiel`
- `SpDnd`

4.6.1 Aufgaben erstellen

Der Instruktor öffnet den Editor einer Wiki Seite und wählt, aus der Reihe von Plugins, das gewünschte aus. Es kann somit separat die Anleitung / Instruktion, die Funktionsbuttons, die Phase und der Aufgabentyp eingefügt werden.
Zusätzlich wird implementiert, dass die Liste der Plugins über die Werkzeugleiste auswählbar ist. Somit muss sich der Instruktor nicht die ganzen Formen merken.
Bei der Erstellung der Plugins können auch Parametrisierungen vorgenommen werden. So zeigt die unten abgebildete Syntax zwei unterschiedliche Darstellungen an.

Die folgenden Abbildungen geben einen Einblick in die Verwendung von Plugins. Die erste Abbildung zeigt die Anwendung des Wiki Syntax. Die zweite Abbildung zeigt die übersetzte Form dieser Syntax.



4.6.2 Wiki Syntax auf HTML und JavaScript übersetzen

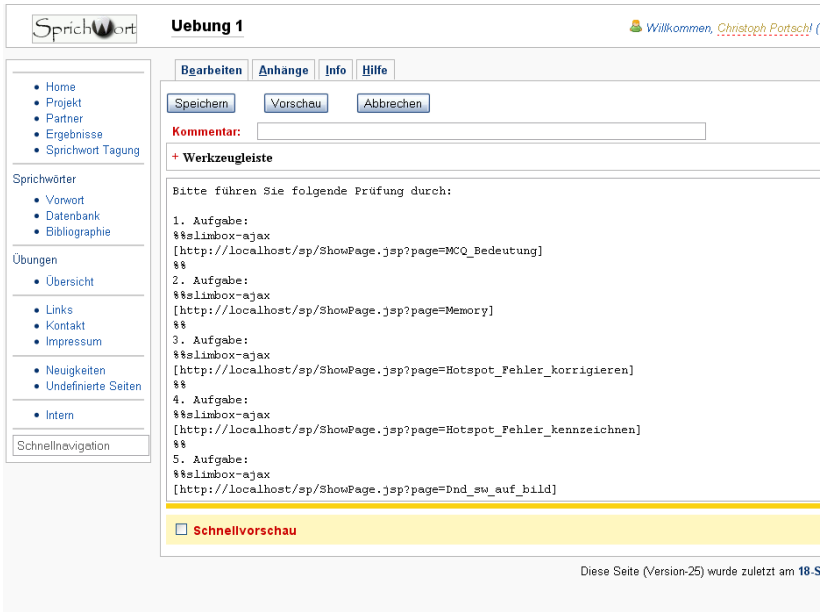
Die Plugin Erweiterungen verwenden eine spezielle Form der Syntax. Dieser Umstand beruht auf der Verwendung von mehreren Konfigurationen für ein Aufgaben Template. Mit der herkömmlichen Syntax für Plugins, können nur jeweils unterschiedliche Parameter definiert werden. Durch die Erweiterung in diesem System können aber auch Gruppen von Parametern mit übergeben werden. Zudem werden auch Arrays unterstützt, die zum Beispiel bei den Antworten für die Multiple Choice Fragen verwendet werden.
Um nicht in den Parsing Prozess für Plugins des Wiki Frameworks einzugreifen, werden Filter angewendet, die diese Aufgabe übernehmen. Ein Filter zerlegt vor der Erstellung des HTML Codes die Wiki Syntax und verarbeitet die übergebenen Parameter. Diese Parameter werden als Variablen in Wiki Kontext geschrieben. Die Wiki Syntax wird so verändert, dass lediglich eine spezielle generierte ID dem Plugin Prozess übergeben wird. Diese ID verhilft später dem Plugin die eigenen Parameter aus dem Kontext zu finden.

Der Plugin Prozess bindet, somit je nach Plugintyp und Parameter ein JavaScript in die generierte HTML Seite ein.

4.6.3 Test und Übungen erstellen

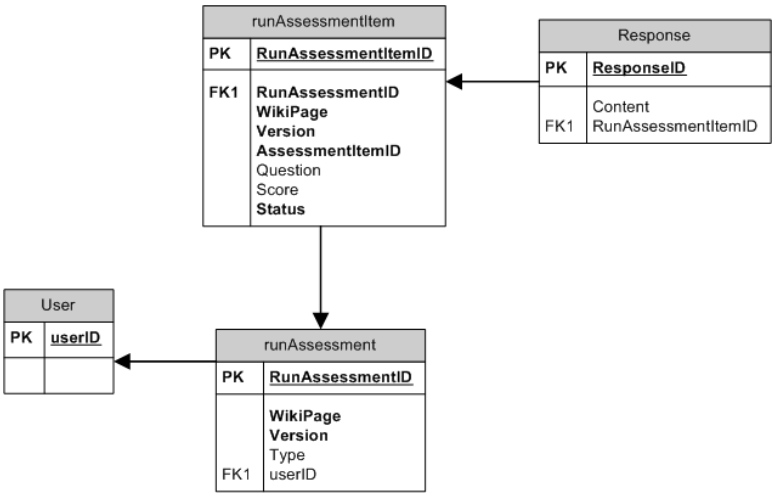
Die Erstellung der Tests und Übungen wurde bis jetzt nur theoretisch durchdacht. Ein Instruktor kann auch hier wieder über den Editor verschiedene Übungen und Tests erzeugen. Er bindet einfach die gewünschten Aufgaben in die Seite über eine `slimbox` ein. Die Aufgaben werden somit bei der Durchführung über ein IFrame angezeigt. Werden Änderungen im IFrame durchgeführt, so wird der Hauptframe benachrichtigt und die Modifikationen durchgeführt.
Problematisch ist jedoch die Verwendung von Wiki Styles, da diese nicht dafür vorgesehen sind. Aus diesem Grund wird wahrscheinlich ein weiteres Plugin implementiert, welches die Funktionalität einer Lightbox einbindet.

Die folgenden Abbildungen zeigen eine exemplarische Prüfung und deren HTML Darstellung.



4.7 Datenbank

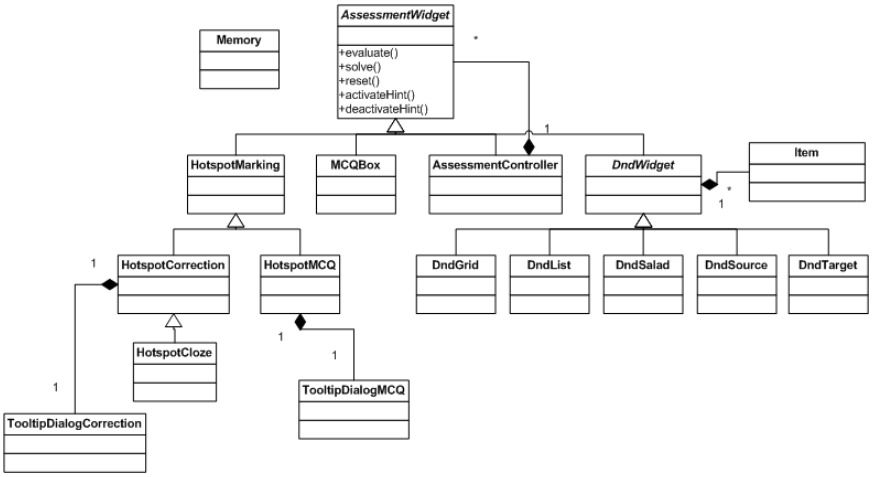
Die Datenbank verwaltet alle Interaktionen des Kandidaten und Bewertungen des Instruktors. Sie bildet somit den Grundstock für die Betrachtung der Ergebnisse und die Dokumentation derer.
In der Tabelle runAssessment werden alle durchgeführten Tests und Übungen abgebildet. Dazu wird die Wiki Seite und die Version der Seite gespeichert um eine exakte Zuordnung auf den Inhalt der Seite zu erzeugen. Zusätzlich wird auch die User ID abgelegt um die Übung oder den Test einen Kandidaten zuzuordnen. Der Typ repräsentiert die Unterscheidung zwischen Übung und Test.
Die Tabelle runAssessmentItem stellt alle gelösten Aufgaben mit deren Ergebnissen dar. Auch hier wird wieder eine exakte Zuordnung auf die Wiki Seite und die Version der Seite durchgeführt. Zusätzlich werden auch der Status und die Bewertung der Frage angezeigt. Der Status zeigt an ob die Prüfung schon bewertet wurde.
Die Tabelle Response lagert die Antworten des Kandidaten in eine eigene Tabelle aus. Ein Kandidat kann bei Multiple Choice zum Beispiel mehrere richtige / oder falsche Antworten auswählen.



Der Hauptfokus dieser Datenbank liegt auf der Prozessunterstützung der Bewertung und der Speicherung der verschiedenen Punkte. Die Implementation dieser Tabellen ist noch nicht erfolgt.

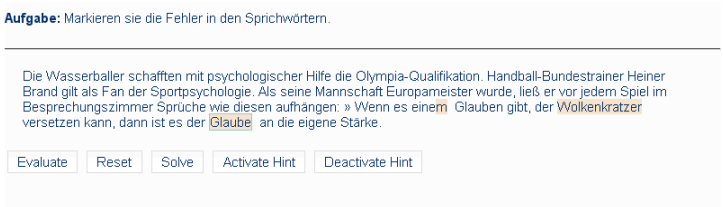
4.8 Aufgabentypen

Nach der Anforderungsanalyse der Fragentypologie, konnten verschiedene Aufgabetypen und hybride Formen identifiziert werden. Um den Anforderungen gerecht zu werden, wurden folgende Klassen erzeugt. Das Aufgabengebiet dieser Klassen ist die Interaktion zwischen Kandidat und Aufgabe zu ermöglichen.



4.8.1 Grundfunktionalitäten der Klassen

Jede Interaktionsklasse (außer Memory) wird von `AssessmentWidget` abgeleitet. Die Basis Klasse definiert fünf Grundfunktionen, die von den Ableitungen implementiert werden. Diese öffentlichen Funktionen setzen sich aus `evaluate`, `solve`, `reset`, `activateHint` und `deactivateHint` zusammen. Wird zum Beispiel ein Button für die Bewertung erzeugt, so ruft dieser die `evaluate` Funktion der Interaktionsklasse auf.
Es besteht aber die Möglichkeit, dass eine Aufgabe aus verschiedenen Interaktionen besteht, wie die folgende Abbildung zeigt:



Ein Button, darf hierbei nicht nur die Bewertungsfunktion eines einzelnen Hotspots aufrufen sondern muss jeden Hotspot Evaluieren. Abhilfe schafft hier der `AssessmentController`. Sobald eine Instanz der Interaktionsklassen erzeugt wird, wird dem Controller eine Referenz übergeben. Der Controller verwaltet diese Referenzen und kann dadurch gesammelte Evaluierungen erzeugen. Ein Button kommuniziert somit nur über einen Controller.

4.8.2 Hotspot

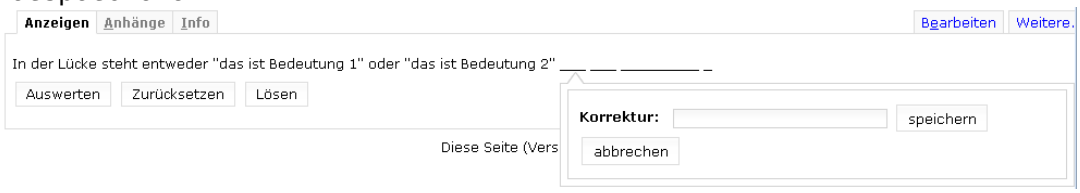
Diese Klassen ermöglichen die Interaktion zwischen einem Kandidaten und einer Hotspot Aufgabe. Die Basis bildet hier die Klasse `HotspotMarking` und implementiert damit die

rudimentären Mouse Events. Durch dieses Widget können Aufgaben wie „Sprichwort kennzeichnen“ durchgeführt werden.
`HotspotCorrection`, `HotspotCloze` und `HotspotMCQ` bilden eine Erweiterung der Basis, indem sie bei der Aktivierung eines Hotspots einen Dialog, mit einer erneuten Interaktionsmöglichkeit erscheinen lassen.
`HotspotCorrection` deckt die Aufgaben wie „Sprichwörter korrigieren“ ab. `HotspotCloze` baut auf dieser Grundfunktionalität der Korrektur auf und ermöglicht dazu eine Darstellung in Form einer Lücke.
`HotspotMCQ` gibt nach der Aktivierung eines Hotspots mehrere Antwortmöglichkeiten vor. Die folgenden Abbildungen zeigen die jeweiligen Darstellungen der Aufgaben.

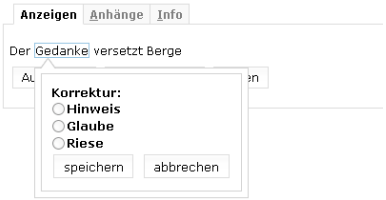
`HotspotMarking:`

`HotspotCorrection:`

`HotspotCloze:`

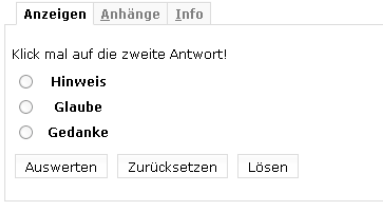


`HotspotMCQ:`



4.8.3 Multiple Choice Fragen

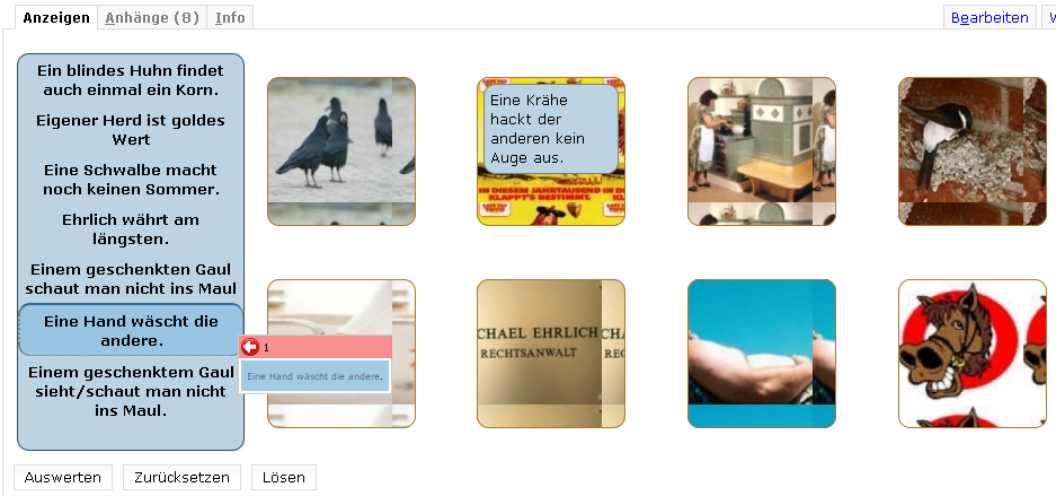
Diese Klasse `MCQBox` repräsentiert die Interaktion zwischen einem Kandidaten und einer Multiple Choice Aufgabe. Zurzeit wird nur Single Response unterstützt. Das heißt, dass ein Kandidat nur jeweils eine Antwort wählen kann. Ziel aber ist eine Multi Response Fähigkeit. Somit kann ein Kandidat mehrere Antworten auswählen. Folgende Abbildung zeigt die Darstellung einer Multiple Choice Aufgabe.

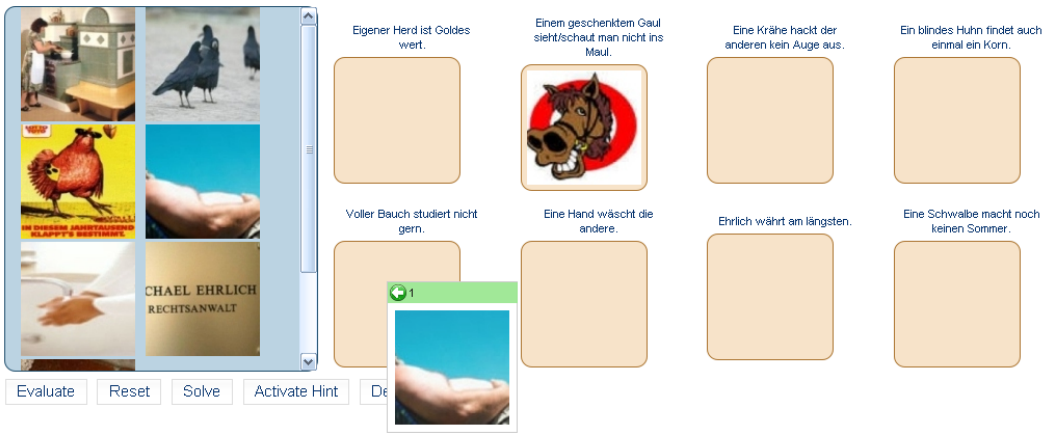


4.8.4 Drag and Drop

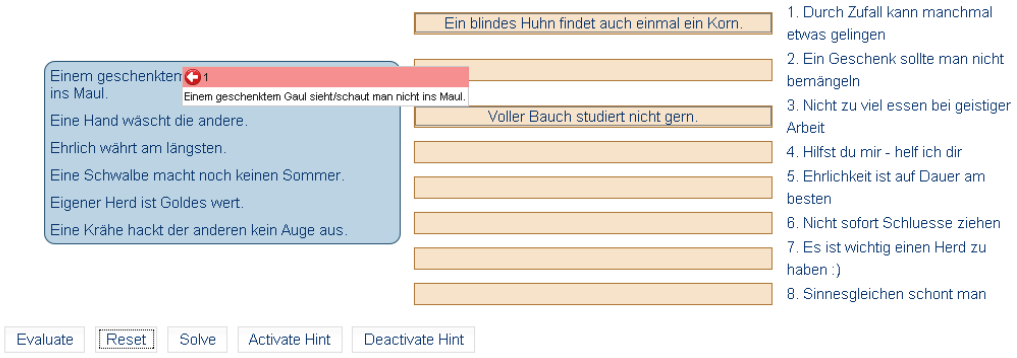
Drag und Drop Aufgaben stellen einen großen Anteil der Aufgabentypologie dar. Um die Grundfunktionalität dieser Interaktionen zu repräsentieren, wurde die Basisklasse `DndWidget` eingeführt. Die Klassen `DndGrid`, `DndList`, `DndSource`, `DndTarget` und `DndSalad` leiten von dieser Basisklasse ab. Die Architektur dieser Klassen ist zurzeit noch experimentell und vermischt teilweise Darstellung und Logik. Aufgrund der unterschiedlichen Anforderungen der verschiedenen Drag and Drop Aufgaben muss hier noch ein passendes Konzept gefunden werden. Die folgenden Abbildungen zeigen die jeweiligen Darstellungen der Aufgaben.

`DndGrid:`

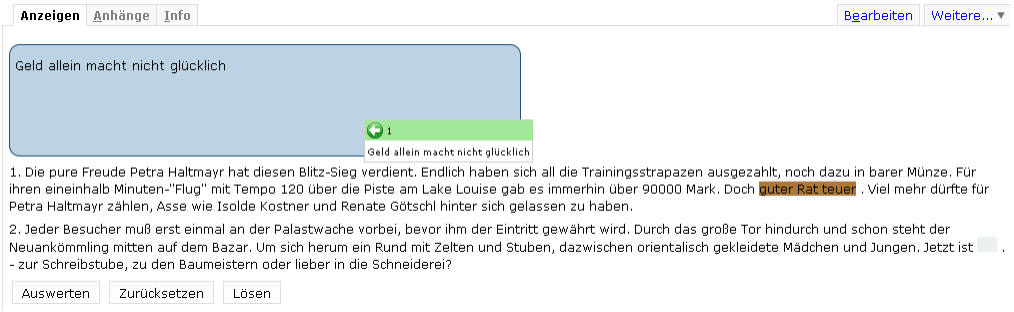




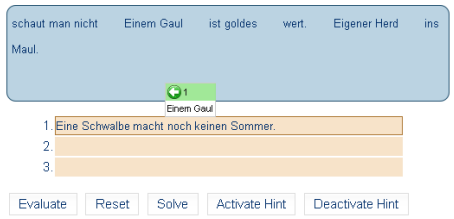
`DndList:`



`DndSource` und `DndTarget`:

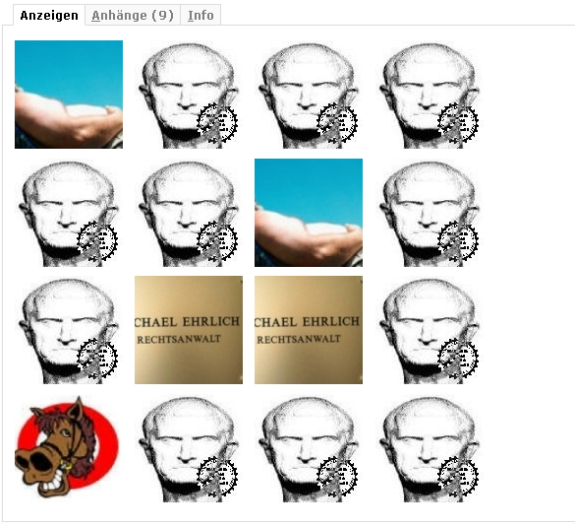


`DndSalad`:



4.8.5 Memory

Die `Memory` Klasse ist unabhängig von den anderen Aufgaben, da sie ausschließlich eine Interaktion darstellt, ohne Punkte zu vergeben. Ein Memory Objekt kann mit einer verschiedenen Anzahl von Feldern initialisiert werden. Somit kann der Schwierigkeitsgrad durch den Instruktor variiert werden. Die folgende Abbildung zeigt die Darstellung eines Memoryspiels.


Anzeigen
Anhänge (9)
Info
CHAEL EHRLICH
RECHTSANWALT
CHAEL EHRLICH
RECHTSANWALT