

Andreas Henrich
Stefanie Gooren-Sieber (Hrsg.)

Web- Technologien

**Grundlagen, ausgewählte Systeme und
Frameworks**

Beiträge des Seminars
im Sommersemester 2011



Lehrstuhl für Medieninformatik
Otto-Friedrich-Universität Bamberg

Inhaltsverzeichnis

Grundlagen

HTML 5 - Stand der Dinge	3
<i>Maximilian Neundorfer</i>	
Silverlight vs. Flash - zwei Namen für eine Technologie?	17
<i>Alexander Hofmann</i>	
Design von Webseiten	29
<i>Alexander Karg</i>	
Grundlagen von Webframeworks	45
<i>Björn Michelsen</i>	
Testen von Webanwendungen	61
<i>Steffen Witt</i>	

Content-Management-Systeme

CMS Typo3	73
<i>Anton Schreiber</i>	
CMS Joomla!	85
<i>Johannes Kupser</i>	
CMS Drupal	99
<i>Maximilian Geissler</i>	
Spezielle CMS: LMS Ilias	111
<i>Johannes Müller</i>	
Spezielle CMS: Shopsystem Interchange	127
<i>David Wagner</i>	

Webframeworks

Cake PHP	143
<i>Sebastian Boosz</i>	
Zend Framework	159
<i>Martin Schmalfuß</i>	
Django	175
<i>Thomas Heinz</i>	
ASP.NET	189
<i>Fabian Biberger</i>	
Google Web Toolkit	211
<i>Kathrin Massny</i>	

1 Einführung

1.1 Der Standard HTML5

HTML5 soll vor allem den längst überfälligen Anschluss an die neuen Technologien des Web 2.0 bringen. Der noch im letzten Jahrtausend veröffentlichte Standard HTML 4.01 hinkt diesen neuen Entwicklungen natürlich längst hinterher. Um dies zu erreichen wurden und werden auch Technologien implementiert, welche auf verschiedensten Wegen - etwa durch Plugins - längst ihre Verbreitung im Web gefunden haben. Zu den wohl interessantesten Neuerungen gehören unter anderem Geolocation, verbesserte Formulare (Web Forms 2.0), neue Möglichkeiten der Strukturierung, direkte Einbettung von Audio und Video, ein Offline-Modus sowie ein Canvas-Element. [VKP11] Diese (und noch weitere) sollen im Verlauf dieser Arbeit noch näher betrachtet werden. Zu beachten ist dass die Neuerungen (und vor allem die wirklich interessanten) nur geringfügig auf HTML-Syntax-Ebene basieren - die wahren Potentiale werden nur durch JavaScript-APIs ausgeschöpft.

Da HTML5 von den Browserherstellern (im Zusammenschluss WHATWG¹) als Gegenentwurf zum vom W3C² vorangetriebenen Standard XHTML 2 ins Rennen geschickt wurde, das W3C diesen Entwurf seit 2009 aber übernommen und seine eigenen Entwicklungen an XHTML eingestellt hat, existieren zwei parallele Spezifikationen. [Wik11]

Diese unterscheiden sich nur geringfügig [Aut11], es ist lediglich zu beachten dass in der Spezifikation des W3C einige Elemente ausgelagert sind, zum Beispiel Canvas oder auch die Drag&Drop-Schnittstelle, welche in der WHATWG-Spezifikation direkt enthalten sind. Dies begründet sich dadurch, dass das W3C einem komplexen Standardisierungsprozess folgen, die WHATWG sich auf der anderen Seite an keinerlei formale Vorgaben halten muss.

1.2 Die Dokument-Deklaration

Der erste signifikante Unterschied zwischen früheren HTML-Versionen und HTML5 besteht in der Deklaration des Doctype. Der einfachste Weg aus einem HTML 4.01- oder XHTML 1.0-Dokument ein HTML5-Dokument zu machen ist die Anpassung desselben. Dieser ist nicht nur kürzer, intuitiver und somit einfacher zu merken sondern auch Versionslos. Da HTML5 vollständig abwärtskompatibel zu bisherigen Standards ist kann er ab sofort eingesetzt werden. [W3C11]

<!DOCTYPE html>

¹ <http://www.whatwg.org>

² <http://www.w3.org>

2 Neuerungen in HTML5

2.1 Semantisches HTML

Während man zur Strukturierung einer Webseite bislang in der Regel auf `<div>`-Container zurückgriff, besteht nun die Möglichkeit spezifische Elemente zur Gliederung von Inhalten zu verwenden. Somit ist es dem Browser möglich, Inhalte zu interpretieren und Zusammenhänge beim Rendering entsprechend zu berücksichtigen. [Bra11, S. 140f.] Ebenfalls enthalten sind weitere Elemente die zu mehr Semantik im Quellcode beitragen, wie `<time>` und `<mark>`. [W3C11] Im `<time>`-Element werden - in einem standardisierten Format und somit maschinenlesbar - Daten gekennzeichnet, die einen zeitlichen Hintergrund darstellen, etwa der Veröffentlichungszeitpunkt einer Nachricht. Kommt es innerhalb des `<article>`-Elements vor so ist damit der Veröffentlichungszeitpunkt des Inhalts, falls außerhalb so der gesamten Dokuments gemeint. Das Element `<mark>` ist dazu gedacht, Ergebnisse von Nutzeraktionen hervorzuheben, etwa Suchanfragen - nicht jedoch um von Entwicklerseite Hervorhebungen zu tätigen. [Pil11]

Element	Beschreibung
<code><header></code>	Kopfbereich einer Seite oder einer Section (Titel, Logo bzw. <code><h></code> -oder <code><hgroup></code> -Elemente)
<code><nav></code>	Enthält Navigationslinks (In erster Linie Hauptnavigation)
<code><section></code>	Thematische Gruppierung von Inhalten
<code><article></code>	In sich geschlossene Inhalte (Zeitungsartikel, Foreneintrag)
<code><aside></code>	Inhalt, der nicht unmittelbar zum Hauptinhalt gehört (Sidebar, Werbung)
<code><footer></code>	Fußbereich einer Seite oder einer Section (Copyright, Querverweise)
<code><hgroup></code>	Kopfbereich einer Section (Gruppe von mehreren <code><h></code> -Elementen)
<code><time></code>	Kennzeichnung von Zeitangaben (Veröffentlichungszeitpunkte)
<code><mark></code>	Farbliche Unterlegung von Text (Suchergebnisse)

Tabelle 1: Neue semantische Elemente in HTML5

Zu beachten ist dass das `<div>`-Element weiterhin verwendet werden kann und auch soll, nämlich dann wenn ein Container ausschließlich als Gestaltungselement dienen soll oder er sich nicht einem der neuen *sectioning*-Elemente zuordnen lässt. So könnten die `<div>`-Container einer Webseite etwa wie folgt durch die neuen semantischen Elemente ersetzt werden: Neben den erwähnten Elementen wurde zahlreiche weitere abgeändert, hinzugefügt oder aus der Spezifikation entfernt, sodass es aus Platzgründen leider nicht möglich ist an dieser Stelle auf jedes einzugehen. [Krö10, S. 322 f.]

```

15  □ <div id="rear">
16  □   □ <div id="header">           <header>
18  □   □ <div id="menu">            <nav>
20  □   □ <div id="content">          <article>
22  □   □ <div id="sidebar">          <aside>
24  □   □ <div id="footer">           <footer>
26  □ </div>

```

Abbildung 1: Verwendung der Elemente zur Strukturierung

2.1.1 Der Outline-Algorithmus

Eine weitere Neuerung ist die Anwendung des Outline-Algorithmus. Dieser besagt dass die Zählung der Überschriftenelemente `<h1>` bis `<h6>` nach Öffnung eines neuen *sectioning*-Elements wieder von vorne beginnt, jedoch eine gesamte Hierarchieebene tiefer als im umschließenden Element. Somit stehen faktisch nun beliebig viele Überschriftenebenen zur Verfügung. [Krö10, S. 76]

2.2 Das Canvas-Element

Das Canvas-Element stellt - wie der Name schon suggeriert - eine 'Leinwand' (den sogenannten Rendering Context) bereit, die mithilfe von JavaScript-Funktionen bemalt werden kann. Im Ausgangszustand handelt es dabei um eine leere, transparente Fläche mit einer Höhe `height` und einer Breite `width`. Zwischen dem Öffnenden und dem Schließenden Tag kann sich Inhalt befinden, der zu Anzeige gebracht wird, wenn der Browser Canvas nicht unterstützt. [W3C11] Somit ergibt sich folgende, simple HTML-Syntax:

```

<canvas height="100" width="100" id="meineLeinwand">
    Canvas wird leider nicht unterstützt
</canvas>

```

2.2.1 Operationen

Die Liste der möglichen Zeichenfunktionen und geometrischen Operationen ähnelt der von SVG-Grafiken. Zwar können auch diese mittels des `<svg>`-Tags eingebunden werden, allerdings bestehen zwischen beiden Möglichkeiten Unterschiede, sodass je nach Verwendungszweck das eine oder das andere eingesetzt werden sollte. [Suc10] So können etwa Formen (Rechtecke, Pfade, Bézierkurven) gezeichnet, Farben und Schatten definiert sowie Transformationen (Rotieren, Skalieren, Verschieben, Transformieren) durchgeführt werden. Auch Animationen sind mithilfe von JavaScript-Zeitfunktionen möglich. Ebenfalls möglich sind komplexere Funktionen wie etwa Bild- und Videomanipulation. So können beispielsweise Videos Pixel für Pixel auf ein Canvas-Element übertragen und dort verändert wieder ausgegeben werden. [Hoe11] Beim Anlegen eines `canvas`-Elements werden sowohl der eventuell vorher vorhandene Rendering Context als auch das zugehörige Bitmap gelöscht. Der Reset geschieht dabei laut Spezifikation:

tion durch einen Reset des `width`- oder `height`-Attributs. [WHA11] Der grobe Ablauf des Zeichnens soll anhand eines simplen Beispiels skizziert werden:

```
// Referenz auf das Canvas-Objekt
var canvas = document.getElementById("meineLeinwand");
// Überprüfung, ob Browser Canvas unterstützt
if (canvas.getContext) {
    // Rendering Context setzen ("2d" oder "webgl")
    var ctx = canvas.getContext("2d");

    // Füllfarbe erstellen (hier: Rot)
    ctx.fillStyle = "rgb(200,0,0)";
    // Rechteck erstellen (10 Einheiten nach Rechts und
    // unten, 55x50 Einheiten groß
    ctx.fillRect (10, 10, 55, 50);

    // Änderung der Füllfarbe (hier: Blau, mit Transparenz
    // von 50%)
    ctx.fillStyle = "rgba(0,0,200,0.5)";
    // Drehung um 50°
    ctx.rotate(50);
    // Erstellung des 2. Rechtecks
    ctx.fillRect (30, 30, 55, 50);
}
```

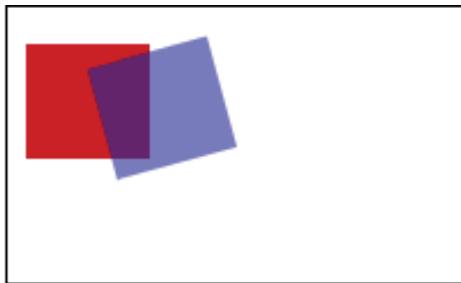


Abbildung 2: Mit JavaScript befülltes Canvas-Element

Der jeweils aktuelle Zustand des Rendering Context kann zusätzlich auf einem LIFO-Stack abgelegt und von dort wieder aufgerufen werden. Dies geschieht durch die Funktionen `save()` und `restore()`. [WHA11] Ebenfalls möglich ist eine globale Festlegung von Zeichenoptionen. So kann per `globalAlpha`-Attribut die Transparenz für alle folgenden Attribute festgelegt werden, das Attribut `globalCompositeOperation` kann 12 verschiedene Werte erhalten, durch welche definiert wird wie sie ein neu gezeichnetes Element zu einem vorher vorhandenen verhält (z.B.: Überlappung, Subtraktion, Addition, ...). [Krö10, S. 261f.]

2.2.2 Frameworks und Bibliotheken

Wie bereits erwähnt lassen sich durch den Einsatz von Timer-Funktionen 2D-Animationen bewerkstelligen. Auch in Richtung 3D-Animationen sind erste Funktionen bereits möglich, realisiert werden diese durch den Einsatz von WebGL. Für die Erstellung dieser Animationen, Browserspielen und die Visualisierung von Daten stehen zahlreiche kostenlose (und auch kostenpflichtige) Frameworks zur Verfügung, die sich je nach Verwendungszweck im Umfang und Komplexität unterscheiden. [BZ11, S. 152ff.]

2.3 Embedding

In HTML5 sind auch endlich native Tags zum direkten Einbetten von Audio- und Video-Dateien spezifiziert. Musste man vorher noch auf Flash-Elemente zurückgreifen, welche als Plugin explizit installiert werden mussten, so ist es nun möglich mithilfe weniger Zeilen HTML-Code Multimedia-Elemente einzufügen. Analog zum Canvas-Element befindet sich auch hier zwischen dem Öffnenden und Schließenden Tag der Fallback-Content, also der Inhalt der dann zum Einsatz kommt wenn `<audio>` bzw. `<video>` nicht unterstützt werden. [W3C11]

```
<audio src="#.mp3">
    <a href="#.mp3">Download #.mp3</a>
</audio>
```

2.3.1 Attribute der Multimedia-Elemente

Die für die Multimedia-Elemente spezifizierten Attribute sind in der folgenden Tabelle dargestellt, für die ersten drei sind dabei keine Attributwerte definiert. [W3S11]

Attribut	Beschreibung
autoplay	Startet das Element automatisch nach laden der Seite
controls	Zeigt die browserspezifischen Steuerbuttons an
loop	Aktiviert die automatische Wiederholung
preload	Lädt das Element mitsamt der Seite vor
src	URL zum Quellelement
height/width	Definiert die Größe des Players (nur <code><video></code>)
poster	URL zu einem Vorschaubild (nur <code><video></code>)

Tabelle 2: Auswahl von Attributen der Multimedia-Elemente

2.3.2 Die Steuerungselemente

Wer unabhängig von den unterschiedlichen Designs der verschiedenen Browser sein möchte, hat die Möglichkeit die Elemente zur Steuerung mithilfe des Multimedia-API nachzubauen und mit JavaScript und CSS dem vorhandenen Design anzupassen. Vor allem für die Buttons stellt dies keine große Herausforderung dar, lediglich die Zeitleiste benötigt einen höheren Arbeitsaufwand. Dafür sind der Phantasie quasi keine Grenzen gesetzt. Die dafür hauptsächlich benötigten Methoden sind: [W3S11]

Methode	Funktion
<code>play()</code>	Startet das Element an der aktuellen Wiedergabeposition, setzt <code>paused</code> auf <code>false</code>
<code>pause()</code>	Pausiert das Element, setzt <code>paused</code> auf <code>true</code>
<code>volume</code>	Zum setzen bzw. auslesen der Lautstärke, wobei 0.0 für stumm und 1.0 für die maximale Lautstärke steht (<code>volume:double</code>)
<code>muted</code>	Wenn <code>true</code> , dann stumm, ansonsten Wiedergabe der aktuell in <code>volume</code> gesetzten Lautstärke (<code>muted:bool</code>)
<code>paused</code>	Enthält den aktuellen Status des Elements, pausiert (<code>true</code>) bzw. laufend (<code>false</code>) (<code>paused:bool</code>)
<code>currentTime</code>	Rückgabewert ist die aktuelle Position der Wiedergabe in Sekunden, kann auch auf einen bestimmten Wert gesetzt werden (<code>currentTime:double</code>)

Tabelle 3: Einige Methoden des Multimedia-API

Ein kleines Anwendungsbeispiel, welches einen Start/Pause-Button erzeugt, wird im Anschluss gezeigt. Wem dieses Vorgehen zu umständlich und aufwändig ist, der hat die Möglichkeit sich zwischen diversen, meist kostenlosen, vorgefertigten und frei konfigurierbaren HTML5-Playern zu entscheiden. Eine große Übersicht über diese findet sich auf den Seiten von video-flash.de.³

```
<video src="#" id="video"></video>
<button id="start" onClick="start ()">Start / Pause</button>

function start (){
    var video = document.getElementById("video");
    if (video.paused){
        video.play();
    }
    else{
        video.pause();
    }
}
```

³ <http://www.video-flash.de/index/html5-videoplayer-uebersicht2>

2.3.3 Der Streit um die Multimedia-Formate

Eine große Problematik besteht im Format der jeweiligen Elemente. Durch die Eigeninteressen der Browserhersteller existiert immer noch kein einheitliches Format, welches von allen Browsern unterstützt wird.

		12+	5+	6-8	9+	10+	5+
Audio-Codecs	MP3	Green	Red	Red	Green	Red	Green
	Vorbis	Green	Green	Red	***	Green	**
	AAC	*	Red	Red	Green	Red	**
Video-Codecs	Theora	Green	Green	Red	Red	Green	**
	VP8	Green	Green	Red	***	Green	**
	H.264	*	Red	Red	Green	Red	
Video-Container	MPEG (.mp4)	*	Red	Red	Green	Red	
	OGG (.ogg)	Green	Green	Red	Red	Green	**
	WebM (.webm)			Red	***	Green	**

Abbildung 3: Codec-Unterstützung nach Browsern

Aus dieser Tabelle⁴ wird klar ersichtlich, dass es unmöglich ist alle potentiellen Nutzer mit nur einem Format zu erreichen. Da MPEG-4-Container (bzw. deren Inhalte) patentrechtlich geschützt sind lehnen es die OpenSource-Browser ab diesen Codec zu implementieren. Auf der anderen Seite sind Microsoft und Apple von diesem Standard überzeugt, sodass auch mittelfristig keine Einigung abzusehen ist. Bis es doch soweit ist bleibt keine andere Wahl als alle Formate anzubieten und dem Browser die Auswahl des für ihr korrekten Codecs über den MIME-Type selbst zu überlassen - auch dafür sieht HTML5 eine Möglichkeit vor.

```
<video controls>
    <source src="#.ogg" type="video/ogg">
    <source src="#.webm" type="video/webm">
    <source src="#.mp4" type="video/mp4">
</video>
```

Durch Angabe des MIME-Types wird unnötiger Traffic vermieden, da der Browser gar nicht erst versucht das Video zu laden (und danach erst festzustellen dass es nicht gelesen werden kann) wenn er mit dem MIME-Type nicht vertraut ist. [Pil11]

⁴ * Support wird eingestellt: Chromium ** Nur mit PlugIns für QuickTime möglich
*** Nur mit nachinstalliertem VP8-Codec

2.4 Formulare

Die neuen 'Web Forms 2.0' unterscheiden sich zwar syntaktisch nur unwesentlich von den bisher bekannten, sind aber doch ein großer Fortschritt - vor allem für Nutzer von Smartphones und sonstigen mobilen Geräten. So werden viele neue Werte für das Attribut `type` von `input`-Elementen spezifiziert und somit Funktionen ermöglicht, die bislang nur durch mehr oder minder aufwändige Skripte möglich waren. [Krö10, S. 112] Und: sämtliche Neuerungen können ohne weitere Anpassungen vornehmen zu müssen ab sofort benutzt werden, denn sie sind vollständig abwärtskompatibel. So kann der etwa der Internet Explorer vor Version 9 nichts mit den neuen Elementen anfangen, dieses Problem wird aber einfach dadurch umgangen dass der althergebrachte `input-type="text"` genutzt, die neuen Typen also einfach ignoriert werden. So werden zwar wahrscheinlich die wenigsten Nutzer die Anpassungen überhaupt bemerken, die bei denen sie schon funktionieren werden aber einen Mehrwert daraus erhalten. [Pil11]

Ein weiterer Vorteil besteht darin, dass Geräte nun kontextspezifisch auf bestimmte `input-types` reagieren können. So werden schon jetzt im iPhone für verschiedene Arten von `type` die jeweils angepassten Tastaturlayouts angezeigt, die vor allem Zeichen hervorheben welche für entsprechende Felder besonders genutzt werden, etwa das "@" bei eMail-Adressen, wobei in diesem Fall die Leer-taste verkleinert wird) oder ein reines Ziffernlayout im Falle von Telefonnummer-Feldern.



Abbildung 4: Angepasste Tastaturlayouts im iPhone (<http://www.elated.com>)

2.4.1 Neue type-Werte und integrierte Validerung

Sämtliche schon in vorherigen HTML-Versionen spezifizierten Typen sind nach wie vor zulässig, in einigen Fällen - etwa beim `file`-Attribut - auch in verbesserter Form. Dort können nun auch mehrere Uploads zeitgleich angenommen sowie akzeptierte Dateiformate angegeben werden. Zusätzlich sind die in der nachfolgenden Tabelle angegebenen Werte neu in die Spezifikation aufgenommen. Neben der Tatsache dass Endgeräte nun also auf den Wert entsprechend reagieren können enthalten einige auch noch eingebaute Validierungsregeln, welche auch automatisch geprüft werden - ohne dass auf Nutzerseite JavaScript

aktiviert sein muss. Ebenfalls bieten manche, etwa `date` und `color` integrierte Auswahlboxen. [Ham06]

<code>type="..."</code>	Anwendung	Regel
<code>search</code>	Ignoriert Formatierungen, um dasselbe Aussehen wie Suchfelder im Betriebssystem zu erzielen	Nein
<code>telephone</code>	Für Telefonnummern	Nein
<code>url</code>	Für URLs	Ja
<code>email</code>	Für eMail-Adressen	Ja
<code>number</code>	Für Zahlen, inklusive Auf-/Ab-Schalter	Ja
<code>range</code>	Auswahl eines Wertes durch Schieberegler	Ja
<code>color</code>	Auswahl von Farbwerten in Farbpaletten	Nein
<code>date</code>	Auswahl eines Datums aus Dropdown-Menü (analog: <code>datetime</code> , <code>month</code> , <code>week</code> , <code>time</code> , <code>datetime-local</code>)	Nein

Tabelle 4: Neue Werte des `type`-Elements

2.4.2 Neue Attribute für `input`-Elemente

Neu sind neben den `type`-Werten auch noch einige Attribute für das `input`-Element. So lassen sich beispielsweise Wertebereiche mithilfe der Attribute `min` (kleinstmöglicher Wert), `max` (größtmöglicher Wert) und `step` (Schrittweite) definieren, welche dann auch automatisch bei Eingabe überprüft werden. [Krö10, S. 230ff.] Wem die vordefinierten Validierungsregeln nicht genügen, der kann durch den Einsatz von regulären Ausdrücken auch eigene Regeln festlegen oder per JavaScript direkt in die Validierungs-API eingreifen und Anpassungen vornehmen. [Krö10, S. 130]

Attribut	Beschreibung
<code>novalidate</code>	Verhindert die Validierung, auch für <code>form</code> -Elemente
<code>placeholder</code>	Zeigt einen 'Platzhalter-Text' im Feld an, der bei Anklicken des Feldes verschwindet
<code>autofocus</code>	Der Cursor springt das mit <code>autofocus</code> deklarierte Feld direkt an
<code>required</code>	Kennzeichnet benötigte Felder, Aussehen des Feldes wird hervorgehoben
<code>autocomplete</code>	Kennzeichnet benötigte Felder, Aussehen des Feldes wird hervorgehoben, auch für <code>form</code> -Elemente
<code>list</code>	Referenziert eine <code>datalist</code> die Vorschläge enthält
<code>pattern</code>	Enthält einen regulären Ausdruck für die erlaubten Zeichen

Tabelle 5: Neue Attribute des `input`-Elements

3 Ausblick / Kritik / Fazit

3.1 HTML5-Support nach Browsern

Die folgende Übersicht zeigt den Support der in dieser Arbeit vorgestellten HTML5-Features der gängigsten Browser in ihren aktuellsten Versionen sowie den prozentuellen Anteil der unterstützten Funktionen am gesamten HTML5-Umfang.

	9	5	5	12	011
Semantische Elemente					
Multimedia-Elemente					
Web Forms 2.0	Red	Yellow	Yellow	Yellow	Green
Form Validation	Red	Yellow	Yellow	Green	Green
Canvas (Basics)	Green	Green	Green	Green	Green
WebGL	Red	Yellow	Red	Yellow	Red
Drag&Drop / Web Storage / Geolocation /Offline-Anwendungen uvm...
Σ Unterstützung HTML5-Features	52%	78%	63%	88%	64%

Abbildung 5: HTML5-Support nach Browsern

3.2 Kritik an Inhalten und Arbeitsprozessen

So notwendig die Weiterentwicklung des ursprünglichen HTML auch ist und war - es gibt immer noch teils heftige Kritik, die zwar in erster Linie auf den Arbeitsprozess der WHATWG bzw. des W3C abzielt, aber auch mit den Inhalten von HTML5 zu tun hat.

Ein wesentliches Problem besteht nach Meinung vieler darin dass eine vernünftige Standardisierung ohnehin nicht machbar ist: Die Macht liegt so oder so bei den Browserherstellern, spezifiziert wurde nur was schon längst implementiert war, verwendet (und von den Browsern dann auch irgendwie verarbeitet) wurde ohnehin schon immer noch viel mehr. Das diese Browserhersteller nun mit der Errichtung der WHATWG dem W3C die Macht über die Standardisierung mehr oder weniger entrissen haben ist allerdings auch differenziert zu sehen: Offiziell kann jeder Nutzer Anregungen und Vorschläge äußern und auch über den aktuellen Entwicklungsstand jederzeit informieren. Dass mit Ian Hickson letztlich aber ausschließlich dem Initiator der WHATWG und zudem einem Angestellten Googles wirkliche Änderungen erlaubt sind scheint doch recht undemokratisch. Was die Entwickler, und damit das 'Volk' des World Wide Web gerne hätten bleibt unter Umständen aufgrund dieses Prozederes außen vor. [Krö10, S. 307f.]

Ein weiterer oft angemerkerter Kritikpunkt ist die Rückkehr zu alter, unsauberer HTML-Syntax. Ein Dokument welches erfolgreich als HTML5 validiert wird muss noch längst nicht XHTML-konform sein. Ein gutes Gegenargument dazu ist sicherlich dass die W3C mit ihrem Versuch XML-konformes HTML unter dem Namen XHTML 2 zu etablieren grandios gescheitert ist und letztlich sogar den HTML5-Entwurf des konkurrierenden WHATWG übernehmen musste.

HTML5 besteht kaum aus HTML, in erster Linie werden schließlich neue JavaScript APIs eingeführt und nur nebensächlich kommt das ein oder andere neue HTML-Tag zum tragen. Diese Tatsache erklärt sich sicherlich auch dadurch dass der erste Arbeitentwurf von HTML5 noch unter dem Namen 'Web Applications 1.0' lief und der Fokus somit von vorneherein klar auf Applikationen, nicht auf statischen Web-Dokumenten lag. Dass die Mitglieder der WHATWG ihr Geld in erster Linie mit Web-Applikationen verdienen sei dabei nur eine Anmerkung am Rande, schließlich werden diese auch immer wichtiger. [Wik11]

Ein generelles Problem liegt in der zum jetzigen Zeitpunkt immer noch mangelhaften und vor allem unausgeglichenen Umsetzung der Neuerungen. Trotz der Tatsache dass von HTML5 in seinen Ursprüngen bereits 2004 zu hören war existieren viele nach wie vor nur auf dem Papier. Und während das W3C von einem 'Last Call', d.h. der letzten Einreichung von Ideen, für Mai 2011 und von 2014 als angestrebtem Jahr für den endgültigen Release spricht [W3C11] will die WHATWG ohnehin keinen festen Standard mehr erheben, HTML5 soll als 'Living Standard' konzipiert werden und versionslos unter ständiger Weiterentwicklung stehen, weswegen in WHATWG-Kreisen nur noch von 'HTML' gesprochen wird. [WHA11] Doch selbst wenn der eine oder der andere Fall eintreten sollte - abhängig ist die Webgemeinde nach wie vor von der Unterstützung durch die Browser.

Weitere Kritikpunkte, die sich auf die konkrete Implementierung beziehen sind etwa die nur sehr geringfügigen Unterschiede zwischen dem `<section>`- und dem `<article>`-Element, sowie den angeblich mangelhafte Nutzen des `<aside>`-Elements. Diese und noch mehr wurden von den 'HTML5 Super Friends' veröffentlicht, einer Gruppe von bekannten Webentwicklern die sich mit dem HTML5 auseinandersetzt. Ebenfalls als signifikantes, in absehbarer Zeit aber wohl kaum lösbares Problem kann der Streit um die Codecs für die Audio- bzw. Videounderstützung gesehen werden. An dieser Stelle sind die Differenzen zwischen den Browserherstellern einfach zu groß, sodass wohl auch weiterhin Daten in allen Formaten vorgehalten werden müssen, um alle Nutzer bedienen zu können. Zwar wurde seitens youtube bereits mit den neuen HTML5-Elementen experimentiert, der bisherige Player ist jedoch nur als Beta-Version verfügbar und unterstützt lediglich den H.264-Codec. [Har10]

3.3 Ausblick / HTML6

Umsetzbar sind sehr viele Inhalte von HTML5 schon jetzt. Der Nutzen zum aktuellen Zeitpunkt darf allerdings als fraglich gesehen werden. Gerade für Webentwickler die ein breites Publikum ansprechen wollen ist es faktisch unmöglich Webseiten in HTML5 ganz oder teilweise umzusetzen. Selbst die neuesten Versionen der gängigsten Webbrower unterstützen längst nicht alle Inhalte, es gibt nur wenige Überschneidungen bei bereits implementierten Features und der Support für den immer noch weit verbreiteten, aber HTML5-unkonformen Internet Explorer 6 wurde von Microsoft bis 2014 verlängert. [Pro10]

Einige Elemente, beispielsweise die neuen Eingabefeldtypen sind zwar dennoch nutzbar, da sie manchen Nutzern schon Vorteile durch erhöhten Komfort bieten können, von älteren Browsern jedoch wie ein normaler `text`-Input behandelt und somit trotzdem korrekt angezeigt werden. Ebenso ist der Doctype vollständig abwärtskompatibel zu älteren HTML-Versionen und kann somit ebenfalls eingesetzt werden. [Pil11]

An weiteren Funktionen wird bereits gearbeitet, im Internet können W3C-Autoren Vorschläge für den nächsten Standard unter dem W3C-Pseudonym HTML.next einbringen. Dem aktuellen Stand nach wird es sich dabei in erster Linie um weitere APIs handeln, die Webentwicklern das Leben erleichtern und viele Spielereien einfacher möglich machen sollen.

Zusätzlich zu den in dieser Arbeit erläuterten Neuerungen stehen auch noch weitere bereit, auf welche es aus Platzgründen nicht möglich war einzugehen: Ein API für eine Drag & Drop-Anwendung, welche sogar Browserübergreifend funktionieren soll. Microformate als Gegenentwurf zu RDFa. Die nur kurz erwähnte Web Storage API, die Möglichkeit Web-Anwendungen Offline-tauglich zu machen und viele weitere kleine geänderte oder neue Elemente sowie die Streichung einige Elemente die es aus HTML 4.01 nicht in die HTML5-Spezifikationen geschafft haben. [Vkp11] Übersichten bzw. Einführungen in die APIs finden sich in den HTML5-Spezifikationen des W3C und der WHATWG.

Literatur

- [Aut11] AUTOR, Ohne: *Gegenüberstellung der W3C- sowie WHATWG-HTML5-Spezifikation.* 2011. – <https://spreadsheets.google.com/pub?key=0Aj29LJkEL8LydFpxSnR6WGtQb1NEUTZTVU1ZQmFU0Gc&hl=de&output=html>; Stand: 02.06.2011
- [Bra11] BRAUN, H.: Nummer 5 bebt. Warum HTML5 ein großer Wurf ist. In: *c't 11* (2011), S. 140f.
- [BZ11] BRAUN, H. ; ZOTA, Dr. V.: Web-Werker. Frameworks und Bibliotheken erleichtern den Umgang mit HTML5. In: *c't 11* (2011), S. 152ff.
- [Ham06] HAMMERSCHMIDT, F.: *Web Forms 2.0.* 2006. – <http://www.malleus.de/docs/tutorials/web-forms-2-0.html>; Stand: 02.06.2011
- [Har10] HARDING, J.: *Flash and the HTML5 video tag.* 2010. – <http://apiblog.youtube.com/2010/06/flash-and-html5-tag.html>; Stand: 02.06.2011
- [Hoe11] HOELLEIN, M.: *Canvas.* 2011. – <http://canvas.quaese.de>; Stand: 02.06.2011
- [Krö10] KRÖNER, P.: *HTML5. Webseiten innovativ und zukunftssicher.* München : Open Source Press, 2010
- [Pil11] PILGRIM, M.: *Dive into HTML5.* 2011. – <http://diveintohtml5.org>; Stand: 02.06.2011
- [Pro10] PROTALINSKI, E.: *Microsoft: dropping support for IE6 is not an option.* 2010. – <http://arstechnica.com/microsoft/news/2009/08/microsoft-dropping-support-for-ie6-is-not-an-option.ars>; Stand: 13.06.2011
- [Suc10] SUCAN, M.: *SVG or Canvas? Choosing between the two.* 2010. – <http://dev.opera.com/articles/view/svg-or-canvas-choosing-between-the-two>; Stand: 02.06.2011
- [VKP11] VAN KESTEREN, A. ; PIETERS, S.: *HTML5 differences from HTML4.* 2011. – <http://www.w3.org/TR/html5-diff>; Stand: 02.06.2011
- [W3C11] W3C: *HTML5. Specification.* 2011. – <http://www.w3.org/TR/html5/>; Stand: 02.06.2011
- [W3S11] W3SCHOOLS: *Learn HTML5.* 2011. – <http://www.w3schools.com/html5>; Stand: 02.06.2011
- [WHA11] WHATWG: *HTML5. Specification.* 2011. – <http://www.whatwg.org/specs/web-apps/current-work/multipage>; Stand: 02.06.2011
- [Wik11] WIKIPEDIA: *HTML5.* 2011. – <http://de.wikipedia.org/w/index.php?title=HTML5&oldid=89560805>; Stand: 02.06.2011

Silverlight vs. Flash – zwei Namen für eine Technologie?

Alexander Hofmann

Abstract. Bei der Entwicklung hin zum WEB 2.0 wird der Nutzer aktiver in die Gestaltung von dynamischen und interaktiven Inhalten mit eingebunden. Darüber hinaus gewinnt die optische Aufbereitung von Inhalten und das simulieren von Desktopanwendungen in Form von Rich Internet Applications eine immer größere Bedeutung. Die Umsetzung dieser Trends basiert meist auf Browser Plug-ins, wie dem Adobe Flash Player oder Microsoft Silverlight. Ob es sich bei diesen beiden Technologien um ebenbürtige Konkurrenzprodukte handelt oder ob es lediglich zwei Namen für eine Technologie sind, soll im Folgenden näher untersucht werden.

Keywords: Silverlight, Flash, Browser Plug-in, WEB 2.0, RIA

1 Einführung

Im Internet sind sogenannte Browser Plug-Ins weit verbreitet. Neben dem sehr bekannten und weit verbreiteten Adobe Flash gibt es seit dem Jahr 2005 auch ein Browser Plug-In von Microsoft, das sogenannte Silverlight. Gerade im Hinblick auf die Entwicklung des Internets zum sogenannten WEB 2.0 haben Rich Internet Applications in Kombination mit Plug-In Technologien an Bedeutung gewonnen. Daher wird im Folgenden zunächst kurz auf den allgemeinen Kontext eingegangen und die grundlegenden Bestandteile erklärt.

1.1 WEB 2.0 und Rich Internet Applications

Das WEB 2.0 ist laut Gabler.de (2011) keine grundlegende neue Entwicklung einer vorhandenen Technologie oder Anwendung. Vielmehr wird darunter eine neue soziotechnische Auffassung bzw. Evolutionsstufe des Internets verstanden. Das Internet wird nicht mehr als statische Plattform zur Informationsverbreitung gesehen, sondern die Nutzer werden aktiv in die Gestaltung und Generierung von dynamischen Inhalten mit einbezogen. Konkrete Beispiele sind Wikis, das Social Tagging¹ und Blogs. Das Internet wird dadurch nach Gabler (2011) "eine von allen produktiv nutzbare Plattform" in der Browser Plug-ins und die damit dynamisch erzeugbaren Inhalte eine zentrale Rolle spielen.

¹ Im Rahmen des Social Tagging werden Informationsobjekte (Texte, Bilder, Videos etc.) von einer Gruppe von Nutzern mit Schlagwörtern (Tags) versehen, ohne dass hierfür genaue Regeln oder ein feststehendes Vokabular spezifiziert werden müsse (Gabler, 2011)

Eine weitere Entwicklung die mit dem WEB 2.0 einhergeht, sind sogenannte Rich Internet Applications (kurz RIA). Laut Huber (2010, S. 42) handelt es sich bei RIAs um Webanwendungen, die Merkmale und Funktionen von Desktopanwendungen besitzen und zwar in Browsern ablaufen, allerdings durch die Bereitstellung von Drag&Drop oder Animationen einen klaren Mehrwert gegenüber statischen Internetseiten liefern. Aus technischer Sicht handelt es sich um eine Client-Server Architektur, bei der Programmcode auch auf dem Client ausgeführt wird. Die Kommunikation zwischen Server und Client beschränkt sich dabei auf das Senden und Empfangen von Daten.

1.2 Browser Plug-in

Browserhersteller haben spezielle Schnittstellen definiert, mit deren Hilfe Erweiterungen für die jeweiligen Webbrower erstellt werden können, diese Erweiterungen werden Browser Plug-ins genannt. Mit ihrer Hilfe ist es Dritten möglich, den Funktionsumfang des Browsers zu erweitern. Zu den bekanntesten Browser Plug-ins zählt der Adobe Flash Player und Microsoft Silverlight. Diese Technologien verfolgen alle einen grundlegenden Aufbau (siehe Fig. 1), welcher im Folgenden näher erklärt wird.

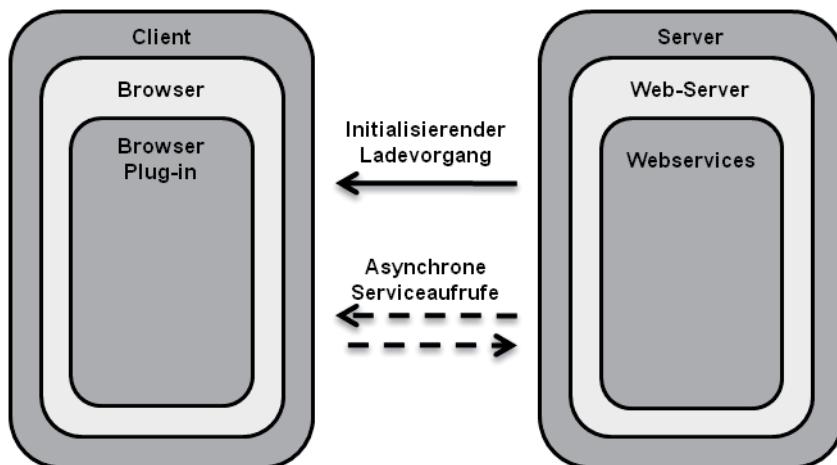


Fig. 1. Interaktion zwischen Client und Server bei Verwendung eines Browser Plug-ins
(In Anlehnung an Huber, 2010, S. 50)

Grundlegend handelt es sich um eine Client-Server Architektur. Auf der Clientseite läuft der Webbrower, welcher wiederum Browser Plug-ins enthalten kann. Auf der Serverseite wird die Webseite auf einem Web-Server gehostet. Zusätzlich kann Businesslogik und andere Funktionalitäten in Form von Webservices abgebildet werden.

Beim initialisierenden Ladevorgang einer Webseite, die eine Plug-in-Technologie verwendet, wird eine gepackte Anwendungsdatei vom Web-Server geladen. Diese gepackte Datei wird nun durch das Browser Plug-in entpackt und der darin enthaltene Anwendungscode geladen. Somit ist es z.B. möglich eine RIA direkt im Browser des Clients ablaufen zu lassen. Da durch das Plug-in eine standardisierte Systemumgebung vorliegt, in der die Anwendungsdatei ausgeführt wird, ist das einheitliche Aussehen der Anwendung, selbst in unterschiedlichen Browsern, gewährleistet. Wird bei der Arbeit mit der Webanwendung z.B. eine Berechnung oder Businesslogik benötigt, so werden die jeweiligen Webservices, die auf dem Web-Server bereitgestellt sind, aufgerufen. Da diese Aufrufe abhängig von den Benutzereingaben erfolgen und die im Browser Plug-in geladene Anwendung nicht auf die Serviceantworten warten muss, spricht man von asynchronen Serviceaufrufen.

2 Silverlight

Wie bereits in Kapitel 1.2 erwähnt, ist Silverlight eines der bekanntesten Browser Plug-ins. Laut Wikipedia.de (2011a) wird Silverlight wie folgt definiert:

"Silverlight ist als proprietäres, programmierbares Plug-in für Windows und Apple Macintosh. Silverlight basiert auf einer reduzierten Version des .NET Frameworks"

Es handelt sich also um ein nicht freies Webbrowser Plug-in von Microsoft, das inzwischen in der Version 4.0 erhältlich ist. Laut Microsoft.com (2011a) erscheint bereits in diesem Jahr die finalen Version 5.0 von Silverlight. Diese Version wird insbesondere als Entwicklungsplattform für Windows Mobile konzipiert sein.

2.1 Entwicklungsumgebung

Als Standardentwicklungsumgebung ist Microsoft VisualStudio 2010 vorgesehen, wobei sich diese Umgebung vor allem an .NET-Entwickler und das Erstellen von Businesslogik richtet. Für Designer und Oberflächengestalter gibt es das Programm Microsoft Expression Blend + SketchFlow. Um die entwickelten Silverlight-anwendungen testen zu können ist darüber hinaus noch die Installation des Silverlight Plug-ins erforderlich.

Um die Entwicklungsumgebung installieren zu können, sind laut Microsoft.com (2011b) mindestens folgende Systemvoraussetzungen notwendig. Neben einem Einkernprozessor mit mindestens 1,6 GHz wird für 32-bit Systeme ein GB Arbeitsspeicher und für 64-bit Systeme zwei GB Arbeitsspeicher benötigt. Da es sich um ein Microsoftprodukt handelt wird ausschließlich Windows ab XP mit dem Servicepack 3 unterstützt. Zusätzlich zur Installation (variable Größe je nach Edition) werden nochmals drei GB freier Festplattenspeicher für Auslagerungsdateien benötigt.

2.2 Funktionsweise

Wie bereits in Kapitel 1.2 angedeutet, handelt es sich bei Silverlight um eine Browser Plug-In Technologie der Firma Microsoft. Dabei besteht Silverlight aus einer speziell für das Internet angepassten Untermenge der Windows Presentation Foundation (WPF) und basiert auf einer verschlankten Version des .NET-Frameworks. Die für den Nutzer sichtbare Oberfläche wird mit Hilfe einer eigens von Microsoft entwickelten XML-Variante der Extensible Application Markup Language (XAML) beschrieben. Das Besondere dabei ist, dass alle zur Ausführung von Silverlight Webseiten benötigten Laufzeitumgebungen im Plug-in selbst bereits enthalten sind. D.h. der Nutzer muss lediglich das Silverlight Plug-in herunterladen und installieren und ist dann bereits in der Lage Silverlightanwendungen zu starten.

Das Arbeitsformat zur Entwicklung einer Silverlightanwendung ist eine SLN-Datei. Dabei handelt es sich um ein Microsoft Expression Blend bzw. Visual Studio Projekt, welches in beiden Programmen geöffnet und bearbeitet werden kann. Nachdem die Entwicklung an der Silverlightanwendung abgeschlossen und das Projekt kompiliert worden ist, entsteht als Ausgabeformat eine XAP-Datei. Eine XAP-Datei oder auch Silverlight Application Package genannt, ist ein ZIP-Ordner, der alle zur Ausführung des Programms benötigten Daten enthält.

2.3 Besonderheiten

Mit Silverlight hat Microsoft einige technologische Besonderheiten eingeführt, dazu zählen u.a. Transformation, Animation, DataBinding und Deep Zoom. Laut Fischer (2009, S. 190ff.) lassen sich Menschen im Bereich Medien vor allem durch Bewegung und Animation begeistern. Dazu gibt es in Silverlight sogenannte Transformationen mit deren Hilfe Größe, Lage, Position und Gestalt von Objekten verändert werden kann. Mehrfach hintereinander geschaltete, sich wiederholende Transformationen eines Objekts ergeben schließlich eine Animation. In Silverlight gibt es drei Haupttransformationen. Objekte können rotiert, skaliert und geschert werden und es kann eine beliebige Kombination aus diesen Grundtransformationen stattfinden.

Gerade in Form einer Animation bieten Transformationen die Möglichkeit die Aufmerksamkeit des Betrachters gezielt zu lenken. Animationen werden in Silverlight nach Fischer (2009, S. 200) in die zwei Klassen "einfache Animation" und "Keyframe Animation" unterteilt. Bei einer einfachen Animation wird nur ein sich ändernder Wert betrachtet. Es gibt einen klar definierten Startwert und einen Endwert, der am Ende der Animation erreicht wird. Keyframe Animationen sind deutlich flexibler, denn sie enthalten auch Zwischenwerte der gesamten Animation. Diese Schlüsselwerte (eng. Keyframes) dienen als Stützstelle für die Interpolation der übrigen Werte und ermöglichen es so, komplexe Animationen zu erstellen. Alle Animationsformen liegen eine Zeitleiste (eng. Timeline) zu Grunde, welche die Dauer der Animation bestimmt.

Ein weiteres Kernelement von Silverlight ist nach Fischer (2009, S. 244ff.) das sogenannte DataBinding. Beim DataBinding wird eine Beziehung zwischen Daten

und Oberfläche hergestellt. Ziel einer solchen Datenbindung ist die Trennung von Oberfläche und Daten, die Bindung besteht immer aus zwei Elementen, dem Bindungsziel und der Bindungsquelle. Die Bindungsquelle ist das Property eines CLR² Objekts, die Dependency Property eines FrameworkElements ist das Bindungsziel (siehe Fig. 2). Das Bindungsobjekt vermittelt zwischen beiden Elementen und führt ggf. eine Wertkonvertierung durch. Beim DataBinding können Daten in beide Richtungen übertragen werden, in der Regel ist die Datenquelle eine Objektinstanz.

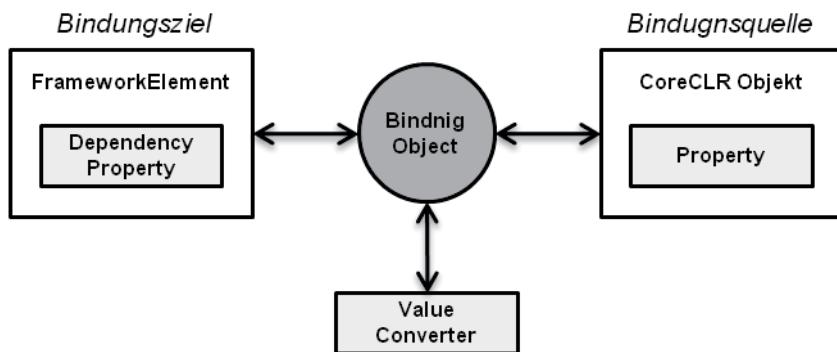


Fig. 3. Datenbindung zwischen einem Oberflächenelement und Daten
(vgl. Fischer 2009, S. 245)

Eine herausragende Eigenschaft von Silverlight ist laut Fischer (2009, S.214) die DeepZoom-Technologie. Mit DeepZoom ist es möglich sehr große Bilder in wenigen Sekunden auf dem Client anzuzeigen. Dabei erscheint das Bild zunächst etwas unklar und sehr grobkörnig und wird dann immer klarer. Der Anwender hat darüber hinaus die Möglichkeit in das Bild weiter hinein zu zoomen um sich so weitere Einzelheiten anzeigen zu lassen. Das Geheimnis von DeepZoom ist, dass die Technologie nur die Bildinformationen lädt, welche für die aktuelle Zoomstufe benötigt werden. Erst wenn die Zoomstufe und damit der Detailgrad erhöht wird, werden weitere Bildinformationen nachgeladen. Nicht sichtbare Teile eines Bildes werden nicht angefordert und daher genügt bereits eine normale Internetverbindung um DeepZoom flüssig verwenden zu können.

Eine weitere Besonderheit von Silverlight sind die unterschiedlichen Entwicklerwerkzeuge für Silverlight. Microsoft stellt sowohl für Entwickler als auch für Designer von Silverlightanwendungen eine eigene Design- bzw. Entwicklungsumgebung zur Verfügung. Mit Expression Blend steht vor allem den Designern und Oberflächenentwicklern ein mächtiges Werkzeug zur Gestaltung von Silverlightanwendungen zur Verfügung. Darüber hinaus ist es mit Expression Blend ebenfalls möglich C# oder VB.NET Sourcecode zu editieren. Den Entwicklern insbesondere den .NET-Entwicklern wird mit VisualStudio 2010 ein sehr mächtiges

² Common Language Runtime (CLR) ist die Laufzeitumgebung des Microsoft .NET-Frameworks.

Entwicklerwerkzeug bereitgestellt, bei dem der Fokus auf die Programmierung von Anwendungen gelegt wurde und welches dem Entwickler ggf. bereits aus der Anwendungsentwicklung von Windowsprogrammen bekannt ist. Analog zum Expression Blend ist es auch in VisualStudio möglich Oberflächen zu gestalten.

Da je nach Anwendungsszenario eines der beiden Programme zu bevorzugen ist und beide Programme auf Grundlage der selben SLN-Datei arbeiten, empfiehlt Wolter (2011) das parallele Arbeiten mit beiden Systemen. Wird am Sourcecode gearbeitet sollte das mit VisualStudio erfolgen und falls die Oberfläche angepasst wird, sollte auf Expression Blend umgestiegen werden. Bei der parallelen Arbeit ist es dabei problemlos möglich eine Silverlightanwendung in beiden Programme gleichzeitig geöffnet zu haben, denn sobald eine Änderung registriert wird, lädt das jeweilige andere Programm diese nach.

3 Flash

Das weltweit häufigste Browser Plug-in ist der Adobe Flash Player. Durch die nahezu flächendeckende Verbreitung des Flash Players (siehe Kapitel 4.2) wird von einem quasi Monopol gesprochen. Er wird benötigt um Flashinhalte im Browser anzuzeigen und ist laut Wikipedia (2011b) definiert als:

"Adobe Flash (kurz Flash, ehemals Macromedia Flash) ist eine proprietäre integrierte Entwicklungsumgebung von Adobe Systems zur Erstellung multimedialer, interaktiver Inhalte."

Der klare Fokus von Flash liegt auf Multimedia und laut Kersken (2009, S. 1f.) ist Flash eine „gelungene Synthese aus Multimedia-Autorensystem und professioneller Entwicklungsumgebung geworden[...]“. Die derzeit aktuelle Version ist Adobe Flash CS5, welche im April 2010 veröffentlicht wurde. Adobe arbeitet bereits an Version CS6, ein Veröffentlichungstermin ist noch nicht bekannt.

3.1 Entwicklungsumgebung

Zur Entwicklung von Flashanwendungen wird Adobe Flash Professional CS5 als Standardentwicklungsumgebung eingesetzt. Flash CS5 richtet sich durch seinen flexiblen Aufbau sowohl an Entwickler als auch an Designer, da sich die Oberfläche an die jeweiligen Bedürfnisse anpassen lässt. Um eine Flashanwendung testen zu können, wird außerdem das Browser Plug-in Adobe Flashplayer benötigt.

Die Entwicklungsumgebung gibt es sowohl für Microsoft Windows als auch für Macintosh. Um Flash CS5 auf einem Windowsrechner zu installieren werden laut Adobe.com (2010) folgende Systemvoraussetzungen benötigt: ein Prozessor mit 1,0 GHz oder mehr dazu mindestens 1 GB Arbeitsspeicher, wobei 2 GB empfehlenswert sind. Alle Windowsversionen ab Windows XP mit Servicepack 2 werden unterstützt. Zusätzlich zur Installation sind 3,5 GB freier Festplattenspeicher erforderlich. Die Systemanforderungen für Mac OS sind ähnlich denen von Windows. Es wird ein Intel

Mehrkerンprozessor und mindestens 1 GB Arbeitsspeicher benötigt. Flash CS5 ist lauffähig ab Mac Os X in der Version 10.5.8 und ist neben der Installation auf weitere 4,0 GB Festplattenspeicher angewiesen. Für Multimedia-Funktionen ist QuickTime 7.6.2 sowohl bei Windows als auch bei Mac OS erforderlich.

3.2 Funktionsweise

Auch Adobe Flash liegt das Konzept eines Browser Plug-Ins zu Grunde. Die Oberfläche wird hauptsächlich mit Hilfe von Vektorgrafiken erzeugt und ist demnach sehr gut skalierbar. Als Programmiersprache kommt ActionScript zum Einsatz. Bei Action Script handelt es sich um eine (inzwischen) objektorientierte Programmiersprache mit Java ähnlicher Syntax. Das Arbeitsformat von Flash sind sogenannte FLA-Dateien, welche nach dem Kompilieren einen Flash Film in Form einer SWF-Datei ergeben. Diese Datei wird analog zu den XAP-Dateien bei Silverlight auf dem Server bereitgestellt und dann im Browser Plug-in des Clients, dem Flash Player, ausgeführt.

3.3 Besonderheiten

Flash bietet einige produktsspezifische Besonderheiten, zu denen laut Fischer (2010, S.16ff.) neben Animation, Multimedia und Sound vor allem die Zeitleiste mit Schlüsselbildern, das Ebenenkonzept und Vektorgrafiken zählen. Die Zeitleiste ist das zentrale Element von Flash, sie steuert den zeitlichen Ablauf eines Films und wird zur Erstellung von Animationen verwendet. Die Zeitleiste ist dabei horizontal in Frames unterteilt, welche einer Abfolge von Einzelbildern entsprechen.

Aus vertikaler Sicht besteht die Zeitleiste aus mehreren Ebenen. Ebenen können als übereinander gestapelte, transparente Folien verstanden werden. Wie bei Folien sind Ebenen nur sichtbar, wenn sie Inhalt enthalten, wobei eine weiter oben liegende Ebene die darunter liegenden Ebenen verdecken kann.

Grafiken werden in Flash in der Regel als Vektoren erstellt. Es gibt bereits vorgefertigte Grundkonturen wie z.B. Rechteck, Polygon und Ellipse aber auch die Möglichkeit einer Freihandzeichnung. Die Kombination aus verschiedenen, beschrifteten bzw. bemalten Ebenen und die Möglichkeit diese Ebenen im Verlauf der Zeitleiste zu manipulieren führt zur Animation. Animationen werden mit sogenannten Schlüsselbildern (eng. Keyframes) erzeugt.

Ein Keyframe repräsentiert dabei einen besonderen Zustand, z.B. die Ausgangsposition einer Vektorgrafik. Ein weiterer Keyframe wird nun als Endposition der selben Vektorgrafik definiert und die über die Zeitleiste stattfindende Transformation von Ausgangsposition zur Endposition als Animation dargestellt. Die Entwicklung einer Flashanwendung ist mit der Erstellung eines Zeichentrickfilms vergleichbar, wobei die Zeichenfläche als Bühne und das kompilierte Programm als Film bezeichnet wird. Wie bei einem Film können neben Grafiken auch Multimediaelemente wie Videos oder Musik eingefügt werden.

Weitere Besonderheiten, gerade für die Entwicklung von Flashanwendungen sind z.B. Arbeitsbereiche der Programmoberfläche oder die Programmiersprache

ActionScript. In Flash CS5 gibt es vordefinierte Arbeitsbereiche für verschiedene Anwendergruppen wie z.B. Designer oder Entwickler und darüber hinaus die Möglichkeit sich Arbeitsbereiche selbst zu definieren und so auf die individuellen Bedürfnisse anzupassen. Eine weitere Besonderheit ist die Programmiersprache ActionScript, die seit der Version 3.0 objektorientiert ist und eine Java ähnliche Syntax besitzt, was gerade Entwicklern aus der Javawelt den Einstieg mit Flash erleichtert.

4 Technologievergleich

Nachdem die Kernkonzepte und der Grundlegende Aufbau der beiden Technologien Silverlight und Flash nun bekannt sind, ist es möglich einen Technologievergleich durchzuführen, an dessen Ende die Frage "Silverlight vs. Flash – zwei Namen für eine Technologie?" beantwortet werden kann.

4.1 Ursprünge

Um die Unterschiede beider Technologien zu verstehen, ist es wichtig, deren Ursprung zu betrachten und die Weiterentwicklung beider Ansätze zu verfolgen. Ein Zitat von Entwickler.de (2007)

"Während Microsoft dem klassischen .NET-Entwickler das Thema Design näherbringt, hat sich Flash vom Animations-Tool immer mehr zu einer Programmiersprache entwickelt."

Flash hat seinen Ursprung im Design und sich Laufe der Zeit, u.a. durch wachsende Anforderungen an den Funktionsumfang, immer mehr zur Entwicklung hin ausgerichtet und ist damit sehr erfolgreich geworden. Seimert (2010, S. 15) bezeichnet Flash als Grafikprogramm mit Animationstendenzen und spricht von einem Quasistandard, den Flash zumindest aktuell unumstritten besitzt.

Silverlight hingegen hat den entgegengesetzten Weg eingeschlagen. Mit seinem Ursprung im .NET-Umfeld richtet sich Silverlight zunächst an Entwickler. Um sich allerdings als Konkurrenzprodukt mit Flash messen zu können, musste der Fokus in Richtung Design verlagert werden, was unter anderem durch das Programm Expression Blend sehr deutlich wird. Laut Fischer (2009, S. 12) erleichtert Silverlight gerade Windows-Entwicklern den Einstieg in die Webentwicklung, da die Grundkonzepte von WPF und dem .NET-Framework in Silverlight als Untermenge enthalten sind.

4.2 Verbreitung

In Fig. 3 ist die Verbreitung der unterschiedlichen Versionen von Silverlight und Flash zu sehen. Dabei ist im oberen Bereich die weltweite und im unteren Bereich die

deutschlandweite Verbreitung dargestellt. Die zugrundeliegenden Daten basieren auf den Beobachtungen der Website RIAStats.com (2011) und erstrecken sich über einen Zeitraum von 30 Tagen. In die weltweite Auswertung sind über 4,5 Millionen individuelle Browser auf 115 Webseiten mit eingeflossen. Die Auswertung für Deutschland basiert auf über 60.000 Browsern und einer Beobachtung von 67 Webseiten.

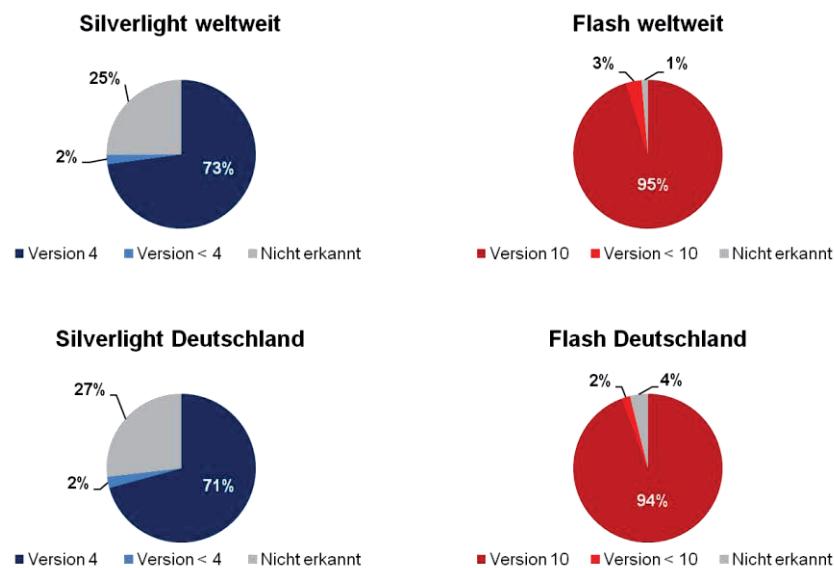


Fig. 3. Verbreitung von Silverlight und Flash in Deutschland und weltweit³

Im weltweiten Vergleich schneidet Flash mit einer Gesamtverbreitung von 98% deutlich besser ab als Silverlight mit einer Gesamtverbreitung von 75%. Ebenfalls wird deutlich, dass bei beiden Technologien die jeweils aktuellste Version am häufigsten vertreten ist und Vorgängerversionen mit 2% bzw. 3% zu vernachlässigen sind. In Deutschland sind die Verbreitungszahlen auf beiden Seiten marginal geringer (jeweils 2%) und decken sich somit mit den weltweiten Verbreitungszahlen.

Für die Entwicklung von Webanwendungen lassen sich daher folgende Schlüsse ziehen. Zum einen kann auf Abwärtskompatibilität zu Vorgängerversionen bei beiden Technologien verzichtet werden. Zum anderen ist beim Einsatz von Flash durch seine flächendeckende Verbreitung gewährleistet, dass die Anwendung auf nahezu allen Rechnern lauffähig ist. Bei der Entwicklung mit Silverlight hingegen muss beachtet werden, dass die Anwendung bei einem Viertel der Anwender nicht ausgeführt werden kann bzw. erst eine Installation des Browser Plug-ins erforderlich wäre.

³ Eigene Auswertung von Daten der Webseite RIAStats.com (2011), Stand. 28. Juni 2011

4.3 Möglichkeiten

Auf der Webseite Shinedraw.com (2009) werden die beiden Technologien Silverlight und Flash, mit Hilfe von konkreten Anwendungsszenarien, direkt miteinander verglichen. Das besondere an der Webseite ist die Bereitstellung aller Anwendungsszenarien als downloadbare Quellcodepakete. Sehr interessant ist der rekursive Baum Shinedraw.com (2009). Sowohl für Silverlight als auch für Flash wurde eine Animation erstellt, bei der mit Hilfe von C#- bzw. ActionScript-Code ein rekursiver Baum mit Blättern berechnet wird. Der schrittweise Aufbau dieses Baumes kann parallel für beide Technologien betrachtet werden und demonstriert sehr anschaulich deren Ebenbürtigkeit. Weitere interessante Anwendungen sind ein sich drehender 3-D Würfel mit der Bezeichnung „Rotating Cube“ oder der Ladevorgang eines sehr großen, externen Bildes. Bei allen verfügbaren Beispielen wird deutlich, dass nahezu alle Anforderungen mit beiden Technologien umgesetzt werden können.

4.4 Einsatzgebiete und Zielgruppe

Bei der Entscheidung über den Einsatz von Silverlight oder Flash gibt es einige Grundlegende Überlegungen, welche vorab getroffen werden sollten. So ist zunächst zu klären welche Erfahrungen bereits mit der jeweiligen Technologie gemacht worden sind. Gerade für (.NET-)Entwickler bietet sich Silverlight als Entwicklungsumgebung an, da in der gewohnten VisualStudio Umgebung gearbeitet werden kann. Für Designer hingegen bietet sich Flash an, da der grundlegende Aufbau dem bekannten Adobe Layout entspricht und auch die Historie der Technologie im Designbereich angesiedelt ist.

Als nächstes muss geklärt werden, für welches Einsatzgebiet die jeweilige Technologie verwendet werden soll. Ein klares Ziel von Silverlight ist dessen Einsatz für seriöse Geschäftsanwendungen. Durch die gegebenen Möglichkeiten und den Fokus auf RIAs ist es relativ einfach möglich komplexe Geschäftsprozesse in Form einer Silverlightanwendung umzusetzen. Ein weiteres großes Einsatzgebiet von Silverlight ist die Entwicklung von Anwendungen für Smartphones, denn Silverlight ist die Entwicklungsplattform für Windows Phone 7.

Flash hingegen legt seinen Hauptfokus auf die Erstellung von Rich-Media Webseiten. Eines der größten Einsatzgebiete ist die Verwendung von Flash für Streaming und Webvideoangebote. Ein anderes großes Einsatzgebiet ist die Bannerwerbung. Durch die Möglichkeit Werbung mit Multimediaelementen wie etwa Animationen oder Musik zu kombinieren, können umfangreiche und optisch ansprechende Werbekampagnen gestartet werden. Neben diesen eher klassischen Einsatzgebieten wird Flash auch im Bereich Zeichentrick und Kunst verwendet, insbesondere bei der Entwicklung von Spielen für Handys und Smartphones. Mit Flash ist es ebenfalls möglich RIAs zu erstellen, allerdings ist die Akzeptanz und Verbreitung solcher Geschäftsanwendungen eher gering. Flash RIAs werden hauptsächlich als PR-Maßnahme von Designagenturen eingesetzt, um das interne Technologie Know-how zu demonstrieren.

5 Fazit

Gerade in der jüngeren Vergangenheit wurde das Thema Silverlight vs. Flash häufig aufgegriffen und in der Öffentlichkeit diskutiert. Microsoft-Manager Bob Muglia gab bei einem Interview unter Golem.de (2010a, 2010b) und ZDNet.de (2010) zu, „Our Silverlight strategy [...] has shifted.“ und machte mit der Aussage „[...] we'll continue to invest in Silverlight and enable developers to build great apps [...] Silverlight is our development platform for Windows Phone“ deutlich, dass Silverlight nicht mehr primär als Konkurrenzprodukt von Flash, sondern als Entwicklungsplattform für Windows Phone weiterentwickelt wird.

Auch bei der öffentlichen Wahrnehmung von Flash werden Veränderungen deutlich. Im Artikel Spiegel.de (2010) wird behauptet „Die Giganten wollen Flash verschwinden lassen“. Als Gigant wird in diesem Kontext unter anderem YouTube gezählt. Die Videoplattform testet bereits HTML5 als Streamingalternative zu Flash. Aber auch Google stellt sich der Monopolsituation von Flash entgegen. Auf Golem.de (2011) wird berichtet, dass Google an einem Konverter für Flash-Dateien arbeitet, um sie in HTML5 zu konvertieren.

Schließlich kommt Netzwerlt.de (2010) beim Technologievergleich von Flash, Silverlight und JavaFX zu einem überraschenden Fazit. So heißt es auf nach einer mehrseitigen Betrachtung der drei Technologien „Fazit: HTML 5 gehört die Zukunft“.

Es wird deutlich, die Frage „Silverlight vs. Flash - zwei Namen für eine Technologie?“ lässt sich nicht pauschal beantworten. Theoretisch lassen sich nahezu alle Anwendungsszenarien mit beiden Technologien umsetzen. Durch einen jahrelangen Entwicklungsprozess besitzen beide Ansätze eine breite gemeinsame Schnittmenge. Aus diesem Gesichtspunkt müsste man die Frage mit ja beantworten. Betrachtet man allerdings die Zielgruppen, Anwendungsgebiete und unterschiedlichen Firmenstrategien, so sind deutliche Unterschiede sichtbar. Während Flash mit seinem quasi Monopol vor allem im Multimediacbereich zum Einsatz kommt und gerade bei Designer sehr beliebt ist, steht bei Silverlight der Einsatz als RIA und der .NET-Entwickler als Zielgruppe im Vordergrund. Durch externe Einflussfaktoren wie etwa HTML5 wird die Konkurrenz der Technologien weiter angefacht. Daher muss die Antwort nein lauten. Es handelt sich um zwei unterschiedliche Technologien, welche zwar eine gemeinsame Schnittmenge besitzen, allerdings einen anderen Fokus verfolgen und somit ihren individuellen Namen verdient haben.

Literaturverzeichnis

1. Huber, T. C., 2010, Silverlight 4: Das umfassende Handbuch, Verlag: Galileo Press GmbH
2. Fischer, O., 2009, Silverlight 3: Crashkurs, Verlag: Microsoft GmbH
3. Seimert, W., 2010, Adobe Flash CS5, Verlag: Hüthig Jehle Rehm
4. Kersken, S., 2009, Praxiswissen Flash CS4, Verlag: O'Reilly Media Inc.
5. Wolter, S., 2011, User Experience Expression Blend, Verlag: DATEV Weiterbildung

Internetquellen

1. Entwickler.de, 2007, <http://entwickler.de/zonen/portale/psecom,id,99,news,37978,.html>, Abgerufen: 30.07.2011
2. Gabler.de, 2011, <http://wirtschaftslexikon.gabler.de/Archiv/80667/web-2-0-v7.html>, Abgerufen: 30.07.2011
3. Wikipedia.org, 2011a, http://de.wikipedia.org/wiki/Microsoft_Silverlight, Abgerufen: 28.06.2011
4. Wikipedia.org, 2011b, http://de.wikipedia.org/wiki/Adobe_Flash, Abgerufen: 28.06.2011
5. Microsoft.com, 2011a, <http://www.microsoft.com/silverlight/future>, Abgerufen: 31.07.2011
6. Microsoft.com, 2011b, <http://www.microsoft.com/visualstudio/en-us/products/2010-editions/ultimate/system-requirements>, Abgerufen: 28.07.2011
7. Adobe.com, 2010, <http://www.adobe.com/de/products/flashplayer/productinfo/features/>, Abgerufen: 29.06.2011
8. Golem.de, 2011, <http://www.golem.de/1106/84546.html>, Abgerufen: 28.06.2011
9. Golem.de, 2010a, <http://www.golem.de/1011/79034.html>, Abgerufen: 30.07.2011
10. Golem.de, 2010b, <http://www.golem.de/1011/79060.html>, Abgerufen: 30.07.2011
11. Netzwelt.de, 2010, <http://www.netzwelt.de/news/82540-netzwelt-wissen-flash-silverlight-javafx.html>, Abgerufen: 30.07.2011
12. ZDNet.de, 2010, <http://www.zdnet.com/blog/microsoft/microsoft-our-strategy-with-silverlight-has-shifted/7834>, Abgerufen: 30.07.2011
13. Spiegel.de, 2010, <http://www.spiegel.de/netzwelt/web/0,1518,680353,00.html>, Abgerufen: 30.07.2011
14. RIAStats.com, 2011, <http://www.riastats.com/>, Abgerufen: 28.06.2011
15. Shinedraw.com, 2009, <http://www.shinedraw.com/flash-vs-silverlight-gallery/>, Abgerufen: 30.07.2011

Design von Webseiten

Alexander Karg

Abstract. In diesem Paper werden die grundlegenden Fragestellungen zum gelungenen Design von Webseiten erörtert. Themen von der Konzeption über die Wahl des richtigen Layouts bis hin zur richtigen Farb- und Bilderauswahl werden behandelt und anhand von Beispielen erläutert. Ziel ist es, dem Leser einen Gesamtüberblick über die Vorgehensweisen im Webdesign zu geben, um selbst in der Lage zu sein Internetauftritten ein interessantes und zeitloses Design zu geben.

Keywords: Webdesign, Konzept, Layout, Bilder, Farben, Texturen, Texte, Typografie

1 Einleitung

Die Schwierigkeit am Design von Webseiten (im Folgenden auch Website oder Webpage genannt) ist der richtige Start und die schier endlose Anzahl an Möglichkeiten ein erfolgreiches Design zu kreieren. Entscheidend ist hierbei, dass es aufs „Ganze“ ankommt. Es ist nicht zielführend nur Teilstücke zu entwerfen oder zu verändern. Wichtiger ist es die einzelnen Teilbereiche geschickt aufeinander abzustimmen, zu kombinieren und somit ein ganzheitliches Bild zu schaffen. Denn nicht der Inhalt und die Funktionalität entscheiden über den ersten Eindruck des Webauftritts, sondern eine ansprechende Gestaltung. Inhalte werden über eine gezielte Auswahl von Farben und Formen oder auch eine passende Wahl von Bildern unterstützt und noch verstärkt.

Es gibt vielerlei verschiedene Anforderungen an ein gutes und zielführendes Design, jedoch ist es immer die subjektive Wahrnehmung eines jeden Einzelnen, die über „Gut“ oder „Schlecht“ entscheidet.

2 Grundlegende Schritte und Zieldefinition

Um ein attraktives Design für eine Webseite zu erstellen sollte man sich im Voraus mit der Zielsetzung und den Zielgruppen auseinander setzen.

Soll meine Webseite als Visitenkarte dienen, eine Verkaufsplattform darstellen oder ist es einfach ein Hobbyprojekt, mit dem sich der Autor in seiner Freizeit beschäftigt? Will ich neue Kunden gewinnen, informieren oder wird mein Internetauf-

tritt zu Marketingzwecken betrieben? Wenn will ich eigentlich erreichen? Gibt es demografische Gemeinsamkeiten? Welche Technologien werden eingesetzt?

3 Layout und Komposition

3.1 Was ist gutes Design?

Gutes Design beruht nach Jason Beaird [1] auf verschiedenen Punkten, über die sich der Gestalter im Klaren sein muss. Ein erfolgreiches Design darf in keinem Falle die Funktionalität des Online-Angebotes beeinflussen. Durch große, aufwändige Animationen, welche unter Umständen nicht abgebrochen werden können, ergeben sich unter anderem lange Ladezeiten. Dasselbe gilt für hochauflösende Bilder. Das Design soll dem Benutzer gefallen und auf Inhalte aufmerksam machen und darf dabei nicht hinderlich sein. Des Weiteren wird durch gelungenes Design die Übersichtlichkeit und Lesbarkeit des Angebots verbessert. Zur Unterstützung einer guten Ästhetik sollte außerdem auf hohe Genauigkeit, Aktualität und Fehlerfreiheit geachtet werden. Ein weiterer Punkt ist die Konzentration auf das wesentliche Ziel des Webauftrittes. Google.com¹ (in Abbildung 1 zu sehen) ist hierfür ein hervorragend geeignetes Beispiel. Es wird auf dieser Seite zentral das bedeutsamste Element, die Suche, dargestellt. Unwichtigere Elemente werden an den Rändern eingeblendet. Durch den farbigen Schriftzug und das Suchfeld in der Mitte der Seite wird sofort die Aufmerksamkeit in diesen Bereich gelenkt. Mit diesen einfachen Designmitteln wird die Funktionalität nicht beeinträchtigt und der User wird zu den wesentlichen Inhalten geführt.



Fig. 1. Google.com

Für die Besucher ist es überdies wichtig, sich mit Hilfe einer intuitiven Navigation durch die Internetseite zu bewegen. Die Hauptnavigation sollte gut erkennbar sein und die Linkbeschreibungen aussagekräftige Bezeichnungen besitzen. Navigationselemente sollten sich auch optisch in das Gesamtbild einfügen und bestenfalls dem

¹ www.google.com

User zeigen, wo er sich gerade befindet. Ein gelungenes Beispiel für eine intuitive und passende Navigation ist in Abbildung 2 zu sehen, welche die Hauptnavigation des Internetauftritts Racket.net.au² darstellt.



Fig. 2. Racket.net.au

Konsistenz und Kontinuität sind weitere fundamentale Designrichtlinien. Ein Benutzer sollte jede einzelne Unterseite als Teil des Ganzen wiedererkennen. Visuelle Indikatoren lassen den Benutzer jederzeit wissen, dass alle Seiten zur selben Webseite gehören. Dies kann am einfachsten durch eine konstante Verwendung von Logos und Navigationselementen erreicht werden. Eine konsequente Farbnutzung trägt ebenfalls dazu bei.

3.2 Layout

Der wohl einfachste Weg ein ansprechendes Layout zu entwerfen ist der Weg über Design-Raster. Ein Design-Raster ist wie eine Art „Bauplan“ für den gesamten Aufbau des Internetauftritts. In Abbildung 3³ [2] werden die wichtigsten Bestandteile dargestellt, welche auf einer gewöhnlichen Internetseite zu finden sind. Mit diesen Hilfsmitteln und unter Anwendung der Drittel-Regel lassen sich die einzelnen Blöcke gut strukturieren und verschiedene Layoutvarianten erstellen. Somit kann ein immer feiner werdendes Raster erstellt und die Anordnung von Elementen im Gesamten und in einzelnen Teilbereichen frei geplant werden bis das gewünschte Ergebnis erreicht ist. Bei der Drittel-Regel handelt es sich um eine Vereinfachung des goldenen Schnittes [1]. Dabei wird beispielsweise ein Blatt Papier immer wieder in gleichgroße Rechtecke gedrittelt. Zusätzlich sollte man sich bei der Planung des Layouts Gedanken bezüglich der Erweiterbarkeit des Internetauftrittes machen, da die Seite sowohl in der Breite und Länge, als auch in der Tiefe wachsen kann und das gewählte Layout leicht auf die neuen Gegebenheiten anpassbar sein bzw. bereits Platz für Erweiterungen vorgesehen sein sollte.

² <http://www.racket.net.au/>

³ <http://www.perfekte-website.de/design/webdesign-anordnung-raster.php>

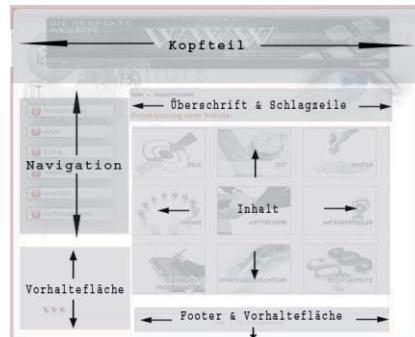


Fig. 3. Beispiel für die Grundbausteine einer Webseite und deren Anordnung

Moderne Webseiten werden hauptsächlich über Cascading Style Sheets (= CSS) gelayoutet. Für kreative Gestaltung ist es empfehlenswert sich von anderen Seiten inspirieren und von Internetauftritten wie CSS Zen Garden⁴ zu lassen, welche viele verschiedene Ideen zur Darstellung bieten. Aber es finden sich durchaus noch Anwendungsfelder für HTML Tabellen. Wissenswert ist, dass es bei beiden Arten der Strukturierung zu Problemen in Hinblick auf Barrierefreiheit führen kann. Hauptsächlich beim Vorlesen von Bildschirmhalten über Screenreader kommt es bei komplexeren Tabellen nicht selten zu Schwierigkeiten, diese im korrekten Sinn wiederzugeben. Als praktisches Beispiel für ein gelungenes und gut strukturiertes Layout kann die Seite der Süddeutschen Zeitung⁵ genannt werden. Hier sind alle Berichte nach einem strikten Bauplan aufgebaut und die Positionierung und Formatierung von Bildern und Texten vorgegeben und konsistent angewandt wie in Abbildung 4 zu sehen.



Fig. 4. Gut strukturierter Aufbau von Elementen in Webseiten

⁴ <http://www.csszengarden.com/>

⁵ <http://www.sueddeutsche.de/>

3.3 Richtige Wahl der Seitenlänge und -breite

Ein zentraler Aspekt bei der Erstellung von Webseiten ist die Wahl der passenden Seitenlänge und -breite. Wie in Abbildung 5 nach den Statistiken der w3schools [3] zu sehen ist geht der Trend aktuell weg von niedrigeren Auflösungen wie 800x600, die noch zur Jahrtausendwende dominierten, hin zu großen Breitbildschirmen mit hoher Auflösung, welche im Laufe der letzten Jahre durch attraktive Marktpreise an Beliebtheit gewonnen haben. Dies ist ein Trend, welcher bei der Gestaltung von Webseiten beachtenswert ist. Mit zunehmender Bildschirmgröße kann davon ausgegangen werden, dass sich User mehrere Programme gleichzeitig auf einem Bildschirm nebeneinander öffnen, was den Raum der Darstellung wieder in den Bereich von 800x600 Pixel bringt. Andererseits ist zu beachten, dass eine große Fläche einfach ungenutzt verstreicht wenn die Browseranwendung in Vollbild angezeigt wird oder die gewählte Schriftgröße verschwindend klein wirkt.

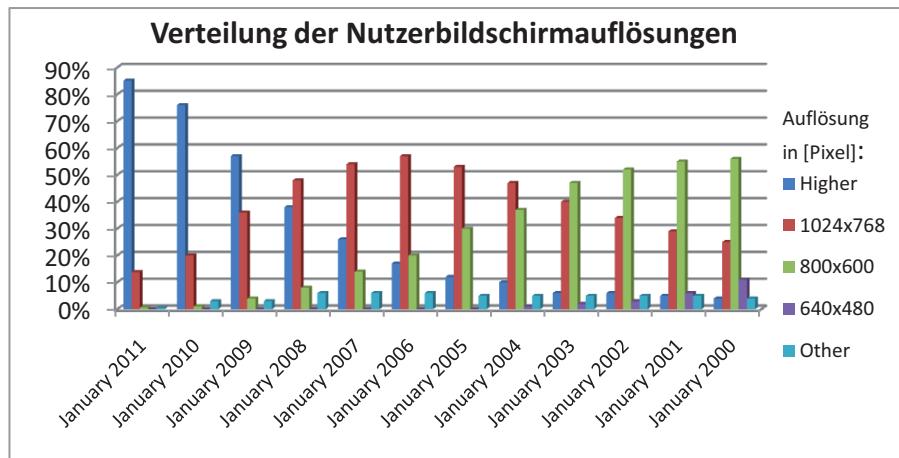


Fig. 5. Verteilung der Bildschirmauflösung von Usern (nach w3schools.com⁶)

Insgesamt ist folglich darauf zu achten die Seitenbreite so zu wählen, dass möglichst viele Nutzer die Seite problemlos, ohne horizontales scrollen, verwenden können. Daher empfiehlt es sich eine maximale Seitenbreite von ca. 750 Pixeln anzustreben [1]. Falls der Inhalt überwiegend aus Texten besteht, ist es sogar ratsam mit nur 560 Pixeln zu arbeiten, da davon ausgegangen werden kann, dass der Benutzer diese Webseite druckt und der Inhalt somit an die Breite eines Din-A4 Blattes angepasst ist.

Die Positionierung elementarer Inhalte sollte auf den oberen ca. 400 Pixel der Seite erfolgen, damit dieser auch ohne scrollen problemlos wahrgenommen werden kann. Dabei verhält es sich ähnlich wie bei Printmedien. Alles Wichtige wird auf der Titelseite über der sogenannten Falz angezeigt, damit der Leser ohne blättern und aufschlagen auf die Schlagzeilen aufmerksam wird. Eine einzelne Seite sollte hinsichtlich der Benutzerfreundlichkeit und Usability nicht zu viel Inhalt enthalten, um zu

⁶ http://www.w3schools.com/browsers/browsers_display.asp

langes vertikales Scrollen zu vermeiden. Als Alternative können Seitenlinks verwendet werden, um sich beim Inhalt in etwa am Format eines Din-A4 Blattes zu orientieren. Bei Grafik- und Datenintensiven Webaufritten ist zusätzlich eine kleinere Seitenhöhe mit weniger Inhalt anzustreben. So ermöglicht man Nutzern mit geringer Anschlussbandbreite kurze Ladezeiten.

4 Farben und Texturen

Farben und Texturen spielen bei der Nutzerführung und der Hervorhebung von Elementen beim Webdesign eine große Rolle. Durch sie können Abgrenzungen und Gruppierungen ohne aufwändige Konfiguration am Design erfolgen. Im Folgenden werden die Bedeutung von Farben und die Farbauswahl näher erläutert, sowie auf die unterschiedlichen Einsatzmöglichkeiten von Texturen eingegangen.

4.1 Einfluss von Farben auf Webseiten

Farben haben eine nicht zu unterschätzende Wirkung auf den Benutzer. Wenn die falschen Farben gewählt werden kann der Inhalt noch so gut sein, die Seite wird die Benutzer immer abschrecken. Mit Farben unterstützt man die Benutzerfreundlichkeit und das Corporate Design von Unternehmen, also die Identifizierung der Webseite mit dem Logo und der Firma selbst. Farben geben einem Webauftakt somit eine Identität. Es ist beispielsweise nur schwer vorstellbar einen Internetauftritt der deutschen Post AG in anderen Farben als Gelb und Rot zu gestalten. Dadurch würden sich Nutzer fehlgeleitet fühlen. Ebenso sollte die Farbe dem Geschlecht angepasst werden, welches der Zielgruppe entspricht. Eine Seite vorzüglich für Männer sollte nicht in lila oder rosa Farbtönen gestaltet werden.

Wie in Abbildung 6 zu sehen wird beim Internetauftritt der Freiwilligen Feuerwehr der Stadt Rödental⁷ auf Farben als Designmittel zurückgegriffen. Durch die Farbwahl rot und weiß kann der User sofort Feuerwehr assoziieren. Eine zusätzliche Farbe, die hier zum Einsatz kommen könnte, wäre blau. Weiterhin ist in der Abbildung zu erkennen, dass im unteren Bereich die Farben rot und grau zur Gruppierung von Elementen und zur Hervorhebung wichtiger Punkte verwendet werden. Im oberen Teil, der Navigation, werden durch den Kontrast Navigationselemente hervorgehoben und diese somit auf den ersten Blick erkennbar.

Wenn es um die richtige Farbe geht, sollte man immer an die folgenden Worte denken [4]:

There's nothing like the right choice of colors to make a Web page work. And nothing like the wrong colors to spoil your content.⁸

⁷ <http://www.feuerwehr-roedental.de/>

⁸ <http://www.coolhomepages.com/cda/color/>

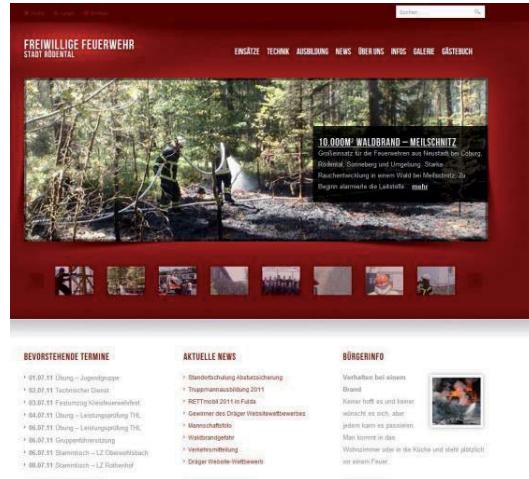


Fig. 6. Freiwillige Feuerwehr Stadt Rödental

4.2 Die Psychologie von Farben

Mit der Psychologie von Farben befassen sich ganze Studienfächer. Sie untersuchen die unterbewusste Beeinflussung durch Farben und Farbkombinationen auf emotionale und verhaltenspsychologische Reaktionen. Verschiedene Farben rufen auch unterschiedliche Wirkungen hervor, abhängig vom jeweiligen Kulturreis. Zudem lassen sich neben den positiven Effekten jeder Farbe auch negative Konnotationen zuschreiben.

Die Farbwahl sollte somit auf das entsprechende Umfeld, die Zielgruppe und das allgemeine Ziel der Webseite angepasst sein. Es ist zum Beispiel empfehlenswert für eine Seite für Gartenbau die Farbe Grün zu wählen oder für einen Bootsverleih eher auf Blautöne zurückzugreifen.

Im Folgenden sind einige Beispiele aufgeführt, welche Wirkungen eine bestimmte Farbe assoziieren kann [1]:

- Rot:
 - Farbe des Blutes, der Energie und Leidenschaft
 - aber auch: Aggression, Gefahr, Gewalt
- Grün:
 - Natur, Leben, Stabilität
 - aber auch: Künstlichkeit, Giftigkeit, Krankheit
- Blau:
 - Beruhigung, Eleganz, Spiritualität
 - aber auch: Trostlosigkeit, Depression

Detailliertere Informationen lassen sich aus diversen Büchern zu diesem Thema entnehmen. Bei Farben sollte dem Designer bewusst sein, dass diese eine gewisse Grundstimmung erzeugen und durch eine geschickte Wahl eine Botschaft klarer übermittelt wird.

4.3 Farbschemata

Bisher wurden nur Informationen über einzelne Farben beschrieben. Eine Webseite besteht aber im Normalfall aus einer Kombination von verschiedenen Farben. In diesem Abschnitt werden verschiedene Möglichkeiten erörtert, mit denen sich harmonische Farbeschemata kombinieren lassen. Im Grunde existieren sechs unterschiedliche Arten [1], welche auf der Anordnung der Farben im Farbkreis basieren: Monochrom, Analog, Komplementär, Teilkomplementär, Triadisch und Tetradisch.



Fig. 7. Die Anordnung der Farben im Farbkreis [6]

Alle in den nachfolgenden Kapiteln verwendeten Beispielbilder wurden mithilfe eines Farbschemagenerators⁹ erstellt. Jene sind eine gute Hilfe bei der richtigen Wahl von individuellen Farbkombinationen. Vielfältige Farben ergeben eine farbenfrohe Gestaltung, allerdings zulasten der Einheitlichkeit und Lesbarkeit.

4.3.1 Monochrom

Bei einem monochromen Schema wird hauptsächlich eine Grundfarbe verwendet. Durch Abstufungen werden hier Kontraste erzeugt.



Fig. 8. Beispiel eines monochromen Farbschemas [6]

⁹ <http://colorschemedesigner.com/>

4.3.2 Analog

Ein analoges Farbschema wird aus verschiedenen Farben gebildet, welche sich im Farbkreis nebeneinander befinden. Zumeist wird auf Farben zurückgegriffen, die nicht mehr als 30° zu beiden Seiten von der gewählten Grundfarbe aus entfernt liegen.



Fig. 9. Beispiel für ein analoges Farbschema [6]

4.3.3 Komplementär

Ausschlaggebend für ein komplementäres Farbschema sind sich im Farbkreis gegenüber liegende Farben. Durch die Wahl ungeeigneter Kombinationen kann es hier aber zu Disharmonie führen. Umgangssprachlich spricht man auch von sich „beißenden“ Farben. Eine solche Kombination sollte man vermeiden.



Fig. 10. Beispiel für ein komplementäres Farbschema [6]

4.3.4 Teilkomplementär, triadisch und tetradisch

Teilkomplementär, triadisch und tetradisch sind im Grunde genommen Variationen des komplementären Farbschemas. Ein teilkomplementäres Schema wird erstellt, indem zu einer Grundfarbe die benachbarten Farben der Komplementärfarbe gewählt werden. Triadische Farbschemata zeichnen sich außerdem dadurch aus, dass alle drei verwendeten Farben denselben Abstand im Farbkreis zueinander besitzen. Von einem

tetradischen Farbschema ist hingegen die Rede, wenn ein Farbschema aus zwei komplementären Farbschemata besteht.

4.4 Texturen

Texturen gehören zu den Gestaltungsmöglichkeiten, mit denen Details ausgearbeitet werden können. CSS bietet weitreichende Optionen, das Aussehen von Webseiten zu optimieren. Unter Texturen versteht man, Objekten ein unvergessliches und einzigartiges Aussehen zu verleihen. Sie sind ein unverzichtbares Designelement, wenn es darum geht ein Design zu perfektionieren. Vielfältige Arten von Texturen sind denkbar [1]:

- Durch Punkte können beispielsweise ganze Bilder dargestellt werden oder Hintergründe aufgewertet werden.
- Linien wiederum eignen sich hervorragend als Designelement zur Unterstreichung von Überschriften oder Links, sowie zur Begrenzung von Elementen, was häufig über horizontale und vertikale Linien erzeugt wird. Geschwungene Linien hingegen ermöglichen ein eher ungezwungenes, freies Design.
- Formen kommen häufig zum Einsatz um Kreativität, Qualität und Professionalität auszustrahlen. Durch beispielsweise abgerundete Ecken erfahren Designs eine sehr viel ansprechendere und weichere Wirkung.

In Abbildung 11 ist der Internetauftritt des Kukral Getränkehandels¹⁰ zu sehen [2]. Hier wird deutlich, wie professionell eine Webseite erscheinen kann, wenn Texturen und Perspektiven richtig aufeinander abgestimmt sind.



Fig. 11. Beispielhafter Einsatz von Texturen

¹⁰ <http://www.getraenke-kukral.de/>

5 Typografie

Auf Typografie kann bei der Gestaltung von Webseiten nicht verzichtet werden. Ohne Fonts ist es nicht möglich Inhalte zu vermitteln. Allerdings gibt es im Internet viele Hindernisse, die dazu beitragen können, dass das gewünschte Schriftergebnis nicht richtig angezeigt wird oder durch die Wahl der falschen Typografie die Vermittlung von Informationen erschwert wird. Eine häufige Fehlerquelle ergibt sich, wenn dem Schriftbild neben dem Layout nicht der gleiche Grad an Aufmerksamkeit gewidmet wird. Allgemein gilt es bei der Gestaltung von Texten darauf zu achten, die Möglichkeiten von CSS zu nutzen und Hierarchien zu verwenden. Texte sollten möglichst kurz gefasst werden und mit ausreichend Kontrasten angezeigt werden. In Puncto Lesefreundlichkeit ist eine gewisse Konstanz auf den einzelnen Seiten und Unterseiten förderlich.

5.1 Schriftarten

Bei der Websiteerstellung spielt die Wahl von geeigneten Schriftarten eine wichtige Rolle. In den 90er Jahren gab es einen kurzen Trend, in dem viele Designer ihrer Webseiten mit individuellen Schriftarten gestalteten[1]. Dies führte jedoch zu Problemen bei der korrekten Anzeige, wenn die Schriftart nicht installiert war. Es musste dann etwa eigens die Schriftart installiert werden oder auf eine optimale Anzeige verzichtet werden. Diese Hindernisse können über die Verwendung von „websicheren“ Schriftarten gelöst werden, die standardmäßig auf den beiden wichtigsten Betriebssystemen installiert sind. Zu dieser Gruppe gehören die in Abbildung 12 gezeigten Font-Typen.

```
ARIAL
ARIAL BLACK
COMIC SANS MS
COURIER NEW
GEORGIA
IMPACT
TIMES NEW ROMAN
TREBUCHET MS
VERDANA
```

Fig. 12. Auflistung der „websicheren Schriftarten“ [1]

Zur Definition von geeigneten Fonts sollte bei der Erstellung einer Webseite auf die Möglichkeiten von CSS zurückgegriffen werden. Dort kann Texten eine Reihenfolge von zu verwendenden Schriftarten zugewiesen werden, beispielhaft mit der CSS-Eigenschaft [1]:

```
font-family: 'Calisto MT', Georgia, 'Times New Roman',
serif;
```

In diesem Beispiel werden der Reihenfolge nach die gewünschten Schriftarten verwendet. Wenn zum Beispiel die Schrift ‚Calisto MT‘ nicht zur Verfügung steht, wird auf die nächste, in diesem Falle ‚Georgia‘ usw., zurückgegriffen.

Als weitere Alternative zur gewünschten Darstellung von Texten bietet sich die Option an, Texte als Bilder zu hinterlegen. Dies ist jedoch nur für statische Texte sinnvoll, an denen nur selten Änderungen vorgenommen werden. Weiterhin denkbar ist es, Texte mit sIFR, also mit Scalable Inman Flash Replacement, zu ersetzen. Hierbei werden hauptsächlich Überschriften mit Hilfe von Flash und JavaScript dargestellt.

5.2 Schrifttyp und Schriftgröße

Um einen möglichst passenden Schrifttyp zu wählen ist das Thema der Webseite von entscheidender Bedeutung. Wenn der Inhalt eher technisch und informativ ist, sollte auf serif- und zeitlose Schriftarten wie zum Beispiel Arial, Times New Roman, Trebuchet MS und Verdana zurückgegriffen werden. Schriften mit Serifen dagegen dienen entweder zur Hervorhebung oder finden bei passenden Stilen Verwendung. Bei beiden Schriften gilt jedoch, dass sie immer unleserlicher werden, je kleiner die Schriftgröße gewählt wird. In einem Designprojekt gilt als weitere Richtlinie eine Verwendung von mehr als 3 unterschiedlichen Schriftarten zu vermeiden [2].

Mit unterschiedlichen Schriftgrößen kann die Wirkung von Textstellen und Wörtern auf den Leser beeinflusst werden. Für Überschriften ersten und zweiten Grades sollte in der Regel eine große Schriftgröße gewählt werden, die zwischen 14 und 20 Pixeln liegt. Texte sollten in Schriftgrößen von 10 bis 13 Pixeln (px) dargestellt werden [2]. Durch die heutzutage immer größer werdenden Bildschirmauflösungen ist jedoch darauf zu achten, dass diese Schriftgrößen unter Umständen sehr klein dargestellt werden. Um diese Probleme zu umgehen empfiehlt es sich die font-size in ‚em‘ anzugeben, was eine relative Größe ist, und nicht in px. In den aktuellen Browsern ist eine stufenlose Anpassung der Schriftgröße mittels ‚em‘ problemlos möglich. Dabei wird die Schriftgröße relativ zur Größe des Elternelements berechnet beziehungsweise der Standardschriftgröße, die im Browser definiert ist.

5.3 Textausrichtung

Um eine gute Lesbarkeit zu erzielen, ist die Textausrichtung enorm wichtig. Eine Textausrichtung kann prinzipiell auf vier verschiedene Arten erfolgen, die aus der Textverarbeitung bekannt sind: linksbündig, rechtsbündig, zentriert und Blocksatz. Bei Blocksatz orientiert sich die Ausrichtung des Textes am linken und rechten Spaltenrand und der Text wirkt rechteckig und stimmig. Allerdings birgt die Blocksatzausrichtung auch Nachteile. Bei bestimmten Spaltenbreiten und Wörtern kann es dazu führen, dass Wörter isoliert jeweils am linken und rechten Spaltenende stehen. Weiterhin kann es zu sehr langgezogenen Wörtern kommen, wenn die Spalten zu schmal sind. Ein dritter Nachteil ist das Auftreten von sogenannten Flüssen wie in Abbildung 13 zusehen. Solche Flüsse entstehen durch Wortzwischenräume, welche zufällig in untereinanderliegenden Zeilen bündig sind. Wenn sich dies nun über mehrere Zeilen

fortsetzt, entsteht ein Fluss der den Leser verwirren kann. Im Printbereich lässt sich dieses Problem leichter umgehen, da man dort auf Silbentrennung oder die Anpassung von Wortabständen zurückgreifen kann. Moderne Browser unterstützen allerdings keine automatische Silbentrennung und bei einer Anpassung der Wortabstände für jede einzelne Zeile kann es zur inkonsistenten Darstellungen von Texten führen.



Fig. 13. Das Auftreten von Flüssen in Blocksatztexten [5]

Zentriert oder rechtsbündige Texte unterbrechen in der Regel ebenfalls den Lese- fluss, da der Leser keine besonderen Orientierungspunkte für den Anfangspunkt der nächsten Zeile hat. Deshalb sollten Texte vornehmlich linksbündig oder als Blocksatz ausgerichtet sein, da dies zu den geringsten Problemen führt und die Übersichtlichkeit wahrt.

Ein Hilfsmittel bei der Erstellung von Texten kann die Seite Typetester.org¹¹ sein. Hier können Texte mit verschiedenen Einstellungen dargestellt und verglichen werden. Anschließend lässt sich der entsprechende CSS-Code für die gewünschte Einstellung generieren.

6 Bilder

Die Metapher „Bilder sagen mehr als tausend Worte“ trifft auch im Webdesign zu. Bilder haben die Eigenschaft, Gesagtes zu unterstreichen oder auf bestimmte Inhalte aufmerksam zu machen. Beachtenswert ist jedoch, dass ein und dasselbe Bild bei Nutzern unterschiedliche Reaktionen hervorrufen kann. Daher gilt es, bei der Auswahl von Bildern die nachstehenden Eigenschaften nach Beard [1] zu beachten.

6.1 Eigenschaften von Grafiken

- Relevanz der Grafik:

Durch Bilder kann das gesamte Layout der Seite interessanter und ansprechender gestaltet werden. Ein Benutzer kann durch richtig gewählte Bilder dazu

¹¹ <http://www.typetester.org/>

animiert werden sich näher mit dem Angebot zu befassen beziehungsweise davon abgeschreckt werden, wenn die Relevanz nicht geboten ist. Des Weiteren dienen Bilder als eine Art optisches Lesezeichen, mit denen sich ein Interessent an den Webauftritt erinnert. Weiterhin dienen sie als Orientierungshilfen um sich auf der Seite zurechtzufinden und um später auf Inhalte zurückzukommen.

- Wirkung der Grafik:

Eingesetzte Bilder sollten neben der Relevanz auch das Interesse des Users wecken. Es sollte eine Verbindung zwischen dem Nutzer und dem Inhalt entstehen. Bilder sind ein Blickfang auf der Seite und sie wecken die Neugierde von Benutzern.

- Anspruch der Grafik:

Um die volle Wirkung, also Interesse und Relevanz zu erzeugen, müssen Bilder einen gewissen Anspruch erfüllen. Dies geschieht in der Regel durch die Verwendung von qualitativ hochwertigen Bildern.

6.2 Herkunft von Bildern

Bei der Herkunft von Grafiken gibt es viele Punkte zu beachten. Am wichtigsten ist die Beschaffung aus legitimen Bildquellen.

Die einfachste Möglichkeit ist die eigenhändige Erstellung von Bildwerken. Vorteile für eigenerstellte Bilder liegen klar auf der Hand. Sie können nach Belieben verändert und vervielfältigt werden und sind frei von Lizenzgebühren. Nachteilig wirkt sich der hohe Zeitaufwand aus, der zur Erstellung hochwertiger Bilder aufgewandt werden muss. Vor allem bei einer Vielzahl von Projekten ist dies keine adäquate Option.

Eine weitere Alternative neben der Eigenanfertigung ist die Beauftragung von Dritten. Allerdings ist diese Variante nur für professionelle Webdesigner eine Option, da die Kostensätze für Fotografen und Designer für kleine Webmaster oder Webdesigner schnell das Budget überschreiten. Der Vorteil ist allerdings die Erstellung von aktuellen und sehr ansprechenden Bildern auf spezielle Wünsche.

Die letzte Variante um an legitime Bildquellen zu gelangen ist der Kauf oder der anderweitige Erwerb. Hierbei werden Bilder über Bilddatenbanken vertrieben. Es gilt dabei eventuell auftretende Nutzungsgebühren zu beachten sowie bei kostenlosen Angeboten mögliche Qualitätseinbußen. Ein Sonderfall des anderweitigen Erwerbs von Bildern ist der Tausch über spezielle Netzwerke. Beim entgeltlichen Erwerb gilt es zudem zwischen verschiedenen Begriffen und Definitionen zu unterscheiden [1]:

- Freie Bilder:

Wie der Name ‚frei‘ schon vermuten lässt, handelt es sich hierbei um kostenlose Bilder. Diese können meist beliebig für private und kommerzielle Zwecke benutzt werden. Beispiele für eine Datenbank mit freien Bildern ist Stock.XCHNG¹² oder MorgueFile¹³. Es sollte aber auch bei diesen Quellen auf Einschränkungen geachtet werden.

¹² <http://www.sxc.hu/>

¹³ <http://www.morguefile.com/>

— Gebührenfreie Bilder:

Bei gebührenfreien Bildern handelt es sich nicht um kostenlose Bilder. Je nach Nutzungsvereinbarung fallen entweder einmalige Nutzungsgebühren an oder es sind anderweitige Bedingungen gegeben. Danach können die Grafiken beliebig oft wiederverwendet werden. Als Quelle für gebührenfreie Bilder kann zum Beispiel iStockphoto¹⁴ verwendet werden. Hier gibt es qualitativ hochwertigere Bilder und eine größere Menge, da die Ersteller bei solchen Plattformen entlohnt werden.

— Bilder mit eingeschränktem Nutzungsrecht:

Bilder mit eingeschränktem Nutzungsrecht werden häufig zur alleinigen Nutzung erworben. Kostenmäßig liegt der Preis sehr hoch, abhängig von der Anzahl der erwarteten Betrachter, der Nutzungszeitspanne und der Größe der beauftragenden Firma. Qualitativ sind solche Grafiken sehr hochwertig.

7 Abschlussfazit

Abschließend lässt sich zusammenfassen, dass es unabdingbar ist sich mit verschiedenen Bereichen der Webseitengestaltung auseinanderzusetzen, wenn man ein überzeugendes Ergebnis erzielen möchte.

Mit dieser Arbeit wurde ein grundlegender Umriss dieser Bereiche angeführt, um die Anfänge zum gelungenen Webdesign zu erleichtern. Zudem tragen zahlreiche praktische Beispiele sowie Bilder zum besseren Verständnis der Einführung bei, vor allem für Diejenigen, die sich zum ersten Mal mit dieser Thematik auseinandersetzen. Insgesamt ist es mit dieser Einführung somit Jedem möglich, sich in die Grundzüge des Designs einzuarbeiten und so dafür zu sorgen, dass die Umsetzung der eigenen Webseite zu einer rundum gelungenen Präsentation wird.

References

1. Beard, J.: Gelungenes Webdesign. Eine praktische Einführung in die Prinzipien der Webseitengestaltung. dpunkt.Verlag GmbH, Heidelberg (2008)
2. Webmaster Tutorials: Wie erstellt man die perfekte Website? Schwerpunkte sind Projektplanung, Technik, Webdesign, Existenzgründung und Online Marketing., <http://www.perfekte-website.de/>
3. Browser Display Statistics, http://www.w3schools.com/browsers/browsers_display.asp
4. Cool Home Pages – Web Site Design Examples, Gallery and Templates, <http://www.coolhomepages.com/cda/color/>
5. Design: im Kleinen - Webseite, <http://vsr.informatik.tu-chemnitz.de/proseminare/www05/doku/design/#prinzip>
6. Color Scheme Designer 3, <http://colorschemedesigner.com/>

¹⁴ www.istockphoto.com

Grundlagen von Webframeworks

Björn Michelsen

Seminararbeit Sommersemester 2011

Otto-Friedrich Universität Bamberg, Lehrstuhl für Medieninformatik

Feldkirchenstraße 21, 96052 Bamberg

bjoern-peter.michelsen@stud.uni-bamberg.de

Abstrakt. Der Trend von statischen zu dynamischen Webseiten hält bis heute ungebrochen an. Dabei bieten dynamische Webseiten durch moderne Webtechnologien und Webstandards stetig neue und anspruchsvollere Funktionen, welche beispielsweise die Grenzen zwischen Desktop- und Webanwendungen immer mehr verschwimmen lassen. Die damit verbundene steigende Komplexität neuer Anwendungen stellt vor allem Webentwickler vor immer neue Herausforderungen. Webframeworks spielen dabei eine zentrale und zunehmend wichtige Rolle, da sie versprechen dem Entwickler redundante Aufgaben abzunehmen, damit sich dieser auf die Lösung spezifischer Probleme seiner Entwicklungsaufgaben konzentrieren kann. Was Webframeworks leisten können, wie sie aufgebaut sind und wie man aus der Vielzahl von Angeboten ein passendes auswählen kann, soll im Folgenden dargestellt werden.

1 Einführung

1.1 Begriffsklärung

In dieser Arbeit soll der Fokus auf Web Application Frameworks gelegt werden, die im Weiteren der Einfachheit halber als Webframeworks bezeichnet werden. Im Begriff drückt „Framework“ aus, dass es sich um ein Software Framework handelt, also eine Menge von kooperierenden Klassen, die ein wiederverwendbares Design für eine bestimmte Klasse von Software ausmachen [1]. Diese Klasse von Software wird durch die Bezeichnung „Application“ beschrieben, welches eine Ausprägung der sieben Frameworktypen „Application“, „Conceptual“, „Domain“, „Platform“, „Component“, „Service“ und „Development“ darstellt [2]. Es wird also ein Framework zur Erstellung von Anwendungen bereitgestellt. Zuletzt beschreibt der Begriff „Web“, dass es sich um ein Framework für den Webkontext handelt. Eine umfassende Definition speziell für Web Application Frameworks bieten Tony C. Shan und Winnie W. Hua: „A Web Application Framework (WAF) is a reusable, skeletal, semi-complete modular platform that can be specialized to produce custom web applications, which commonly serve the web browsers via the Http(s) protocol“ [2].

1.2 Motivation

Die grundlegende Motivation hinter der Verwendung eines Webframeworks ist die signifikante Reduzierung des Zeit- und Ressourcenaufwands, der benötigt wird, um eine Webanwendung zu entwickeln und zu warten. Dies wird vor allem durch die Förderung der Codewiederverwendung erreicht, die wiederum unter anderem darauf beruht, dass ein Großteil von Webanwendungen eine gemeinsame Menge an grundlegenden Funktionalitäten implementieren bzw. bereitstellen muss. Ein Webframework kann nun viele dieser Funktionen in vorgefertigter Form, bei etablierten Frameworks optimiert durch viele Entwicklungszyklen, bereitstellen, damit sich Entwickler auf die Anpassung an die eigentlichen Geschäftsanforderungen fokussieren können. Eine weitere Motivation besteht darin, dass viele Web Application Frameworks offene, bewährte, weit akzeptierte Standards und Technologien einsetzen. Dies bedeutet, dass Entwickler mit Erfahrung in diesen Standards weniger Aufwand haben, ein Webframework zu erlernen [2]. Hierbei muss allerdings trotzdem die generell für Webframeworks steile Lernkurve beachtet werden [3]. Die Verwendung von Standardtechnologien bietet zudem den Vorteil, dass die erstellte Webanwendung leicht in existierenden Infrastrukturen eingesetzt werden kann und ist dabei in der Lage, bestehende Hardware, Software und Prozesse auszunutzen [2].

1.3 Nachteile

Ein Nachteil von Frameworks besteht in der bereits genannten oft steilen Lernkurve, um Aspekte des Aufbaus, der Verwendung und des Einsatzes zu erlernen. Solch eine steile Lernkurve besagt allerdings auch, dass in kurzer Zeit viel Wissen und Erfahrung aufgebaut werden kann, welche im Nachhinein durch eine durch das Framework ermöglichte erhöhte Effizienz und Produktivität bei der Webentwicklung ausgeglichen werden kann [3].

Ein weiterer Nachteil stellt die Tatsache dar, dass eine Abhängigkeit zu einem bestimmten Framework oder Hersteller aufgebaut wird, die negative Konsequenzen haben kann. Beispielsweise kann es zu Problemen kommen, wenn die Organisation oder Community, die das Framework erstellt hat, die Entwicklung aufgibt [3].

1.4 Schlüsseldesignprinzipien

In ihrem Artikel „Taxonomy of Java Web Application Frameworks“ haben Tony C. Shan und Winnie W. Hua insgesamt acht Schlüsseldesignprinzipien identifiziert, welche Webframeworks typischerweise aufweisen oder aufweisen sollten: [2]

Einfachheit: Wenig und einfacher Code sollte nötig sein, um das Framework zu verwenden beziehungsweise an ein spezifisches Problem anzupassen.

Konsistenz: Komponenten und Konventionen sollten konsistent sein.

Effizienz: Fertige Applikationen sollten eine hohe Leistung aufweisen und gut skalierbar sein.

Integration: Ein Framework sollte in etablierte, bestehende Lösungen integrierbar sein.

Wiederverwendbarkeit: Teile eines Framework sollten vollständig wiederverwendbar und einfach einzusetzen bzw. zu verteilen sein.

„*Non-intrusiveness*“: Eine eingesetzte Markup-Sprache wie HTML sollte nicht unnötig viel Semantik der Programmiersprache für die Anwendungslogik enthalten, um die Kompatibilität mit HTML-Editoren für Webdesigner zu gewährleisten. Dieser Aspekt wird in der Thematik der Architekturen, die eine Trennung von Anwendungslogik und Präsentation ermöglichen sollen, sowie im Abschnitt zu Trends und Entwicklungen nochmals aufgegriffen.

Diagnose: Ein Framework sollte hilfreiche Diagnose- und Debugging-Informationen bereitstellen.

Entwicklungswerzeuge: Ein Framework sollte eine maximale Kompatibilität mit Entwicklungswerzeugen haben, ohne dabei von speziellen Werkzeugen abhängig zu sein.

Neben diesen acht können zwei weitere zentrale Prinzipien identifiziert werden: „*Don't Repeat Yourself*“ und „*Convention over Configuration*“ [4].

Das erste Prinzip, „*Don't Repeat Yourself*“, kurz DRY-Prinzip genannt, sagt aus, dass Redundanz von Code vermieden werden sollte. Vor allem die Konsistenz einer Anwendung soll hierdurch gewahrt werden, was Vorteile für die Wartung bietet. Das Ziel wird von Andrew Hunt und David Thomas in „*The Pragmatic Programmer*“ wie folgt definiert: „*Every piece of knowledge must have a single, unambiguous, authoritative representation within a System*“ [5].

Das zweite Prinzip, „*Convention over Configuration*“, welches vor allem durch das Webframework „Ruby on Rails“ an Popularität gewonnen hat, wird auch von anderen bekannten Frameworks wie Spring verfolgt. Die Grundidee hier ist, dass ein Framework mit sinnvollen Standardwerten vorkonfiguriert ist und nur um Ausnahmen abzudecken einer extra Anpassung durch den Entwickler bedarf. Das Ziel ist es, dem Entwickler Arbeit abzunehmen, da er nicht jeden Parameter selbst konfigurieren muss.

1.5 Frameworks vs. CMS

Ein Content Management System (CMS) ist per Definition eine Anwendung, die einem Benutzer erlaubt, Daten und Informationen eines Systems zu verwalten. Bei einem web-basierten CMS ist dies der Inhalt einer Webseite [6]. Es ist also viel spezieller definiert als ein Web Application Webframework, welches sich nicht auf einen bestimmten Anwendungsbereich der erstellten Webanwendungen beschränkt [2]. Zudem gibt es Content Management Frameworks (CMF), welche eine höhere Anpassbarkeit und Erweiterbarkeit als CMS bieten wollen. Eine genaue Abgrenzung der Begriffe Webframework, CMS und CMF ist allerdings oft schwer, da diese gemeinsame Funktionalitäten aufweisen können und es Lösungen wie „*Drupal*“ [7] gibt, welche sich sowohl als CMS und CMF klassifizieren wollen [8].

1.6 Frameworks vs. Bibliotheken

Frameworks und Bibliotheken sind beides Möglichkeiten der Codewiederverwendung. Dabei stellen Bibliotheken oder Toolkits bestimmte Funktionalitäten bereit, welche vom aufrufenden Programm oder Programmteil verwendet werden können. Toolkits sind hier eine Menge von Bibliotheken, die einen allgemeinen Zweck erfüllen sollen [1].

Frameworks bestimmen hingegen die gesamte Architektur einer Anwendung. Sie stellen, entsprechend der in Abschnitt 1.1 genannten Definition, kooperierende Klassen bereit, die ein wiederverwendbares Design ausmachen. Wiederverwendung auf einem so hohen Niveau führt dazu, dass der Kontrollfluss nicht mehr von der Applikation selbst ausgeht, sondern vom Framework. Dieses Prinzip, welches durch das „Inversion of Control“-Konzept beschrieben wird, ist also ein definierender Unterschied zwischen Frameworks und Bibliotheken. Da dieses Konzept ein grundlegendes Architekturnmerkmal von Frameworks darstellt, wird es mit seinen Umsetzungsformen im Kontext der Architekturen näher betrachtet [1].

1.7 Frameworks vs. Entwurfsmuster

Eine Abgrenzung von Frameworks und Entwurfsmustern ist sinnvoll, da beide Konzepte Arten von Designwiederverwendung darstellen. Sie weisen grundsätzlich drei Hauptunterschiede auf. Zum einen sind Entwurfsmuster abstrakter als Frameworks [1]. Sie beschreiben nur die abstrakten Interaktionen zwischen kollaborierenden Objekten [9]. Dieser Unterschied wird klar, wenn man betrachtet, dass Entwurfsmuster nicht wie Frameworks in Code implementiert werden können. Lediglich Beispiele für die Umsetzung von Entwurfsmustern können in Code festgehalten umgesetzt werden [1]. Ferner sind Entwurfsmuster kleinere Architekturelemente, weswegen sie auch als Mikroarchitekturen bezeichnet werden [9] und weniger spezialisiert sind als Frameworks, die immer auf ein spezifisches Anwendungsgebiet ausgerichtet sind [1].

2 Architekturnmerkmale

2.1 Frozen Spots vs. Hot Spots

Wenn man Frameworks auf einer abstrakten Ebene betrachtet, kann man „Frozen Spots“ und „Hot Spots“ identifizieren. Dabei bezeichnen „Frozen Spots“ die statischen Teile eines Frameworks, welche ohne weitere Veränderung bereit für die Verwendung sind [10]. Sie definieren die grundlegende Architektur des Softwaresystems [2].

Im Gegensatz dazu werden die flexiblen Teile eines Frameworks, die beispielsweise durch das Überschreiben von Methoden angepasst werden, als „Hot Spots“ bezeichnet. Es ist wichtig, dass die „Hot Spots“ eines Frameworks so gewählt

sind, dass sie eine möglichst flexible Anpassung eines Frameworks an ein spezifisches Problem gewährleisten [10].

2.2 Inversion of Control

Frameworks weisen oft die schon bei der Abgrenzung zu Bibliotheken angesprochene Charakteristik auf, die als „Inversion of Control“ bezeichnet wird. Dabei übernimmt das Framework die Rolle des Hauptprogramms, welches den Kontrollfluss koordiniert und kontrolliert. Der Verwender eines Frameworks konfiguriert zunächst die flexiblen Teile des Frameworks, die „Hot Spots“, um so das Verhalten an ein spezifisches Problem anzupassen [11]. Diese implementierten Teile werden dann vom Framework kontrolliert. Aufgrund seiner Natur wird dieses Prinzip auch als Hollywood-Prinzip - „Don't call us, we'll call you“ - bezeichnet [12].

„Inversion of Control“ ist allerdings nur ein allgemeiner Begriff für das Konzept der Umkehrung des Kontrollflusses. Ein spezielles Design zur Umsetzung dieses Konzepts stellt das „Dependency Injection“-Entwurfsmuster dar. Man unterscheidet hierbei drei Arten der „Dependency Injection“: „Constructor Injection“, „Setter Injection“ und „Interface Injection“. Diese werden, in dieser Reihenfolge, ebenfalls als „Type 1 IoC“, „Type 2 IoC“ und „Type 3 IoC“ bezeichnet, wobei IoC für „Inversion of Control“ steht. Die Grundidee ist jeweils, dass ein Objekt mit Wissen über die Erzeugung anderer Objekte die Aufgabe übernimmt, neue Objekte zu erzeugen. Dabei übernimmt es ebenfalls die Erstellung indirekter Abhängigkeiten. Je nach Art werden Setter, Konstruktoren oder Interfaces für diesen Zweck verwendet [13].

2.3 Model-View-Controller

Die Mehrheit aller Frameworks implementieren, in welcher Weise auch immer, das Model-View-Controller (MVC) Architekturmuster [3], welches in der gesamten Webentwicklung weit verbreitet ist [14]. Ziel dieses Architekturmusters ist eine Trennung von Anwendungslogik und Präsentationsschicht [3].

Dazu findet eine Differenzierung in die drei Objekte Model, View und Controller statt, welche minimal untereinander gekoppelt sind. Das Model hat die Aufgabe die eigentlichen Daten zu halten und Geschäftslogik oder Operationen bereitzustellen, um auf diese Daten zuzugreifen oder sie zu modifizieren. Die grundlegende Aufgabe des Views ist die Darstellung der Daten des Models [14]. Model und View stehen gemäß dem Oberserver-Pattern in Beziehung [1].

Der View hat die Veröffentlichungen des Models abonniert und wird bei einer Veränderung des Models benachrichtigt, worauf er den Zustand des Models abfragen kann, um so seine Darstellung aktualisieren zu können. Des Weiteren hat der View die Aufgabe, Benutzereingaben an den Controller weiterzuleiten. Dieser wiederum definiert das Verhalten der Anwendung. Er bildet die Schnittstelle zwischen Benutzereingaben auf dem View und entsprechende Operationen auf dem Model. Er sendet im Fall einer Webanwendung zu einer Benutzereingabe passende HTTP GET-

und POST-Anfragen zum Zugriff oder Modifikation des Models. Der Controller kann zudem Views für die Darstellung auswählen [14].

Die Beziehung zwischen dem View und Controller lässt sich anhand des Strategy-Patterns beschreiben. Dieses definiert, dass unterschiedliche Strategien austauschbare Elemente darstellen, die verschiedene Algorithmen implementieren können. Im Fall der View-Controller-Beziehung stellen also verschiedene Controller verschiedene Strategien dar. Es ist ersichtlich, dass das MVC-Architekturmuster aus diversen Entwurfsmustern besteht. Ein weiteres Beispiel für ein in MVC integriertes Entwurfsmuster ist das Composite-Pattern, welches vom View eingesetzt wird und die hierarchische Anordnung von Elementen innerhalb des Views definiert [1].

2.4 JSP-Model 2

Java-basierte Webframeworks implementieren MVC typischerweise in der JSP-Model-2-Architektur [2]. Dieses Modell stellt Richtlinien von Sun dar, JSP im MVC-Kontext anzuwenden. Nach dieser Architektur ist der Controller ein Servlet, das Model eine JavaBean und der View eine JSP-Seite. Eine Benutzeranfrage wird vom Browser an ein Servlet gerichtet, welches auf eine JavaBean zugreifen kann und diese – und damit die zugrundeliegenden in einer Datenbank enthaltenen Daten – modifizieren kann. Zudem wählt das Servlet die passende JSP-Seite aus und übergibt dieser die Kontrolle, damit sie die Daten des Models auf dem Browser des Benutzers anzeigen kann [15] [16]. Ein Beispiel für ein Webframework, welches MVC gemäß JSP-Model 2 implementiert, ist „Apache Struts“ [17] [18].

2.5 Weitere Architekturmuster

Auch wenn der größte Teil der Webframeworks dem MVC-Pattern folgen [3] gibt es Frameworks, die auf anderen Architekturmustern aufbauen. Beispiele für weitere Architekturmuster sind Model-View-Presenter (MVP), z.B. unterstützt von „ASP.NET Webforms MVP“ [19], Model-View-ViewModel (MVVM), z.B. unterstützt von „ASP.NET MVVM“ [20] und Hierarchical-Model-View-Controller (HMVC), z.B. unterstützt von „Kohana“ [21].

Während alle diese Alternativen dem klassischen MVC ähneln [22], gibt es auch radikalere Ansätze, wie das Scala-basierte Webframework „Lift“ [23] zeigt. Auch wenn es möglich ist, MVC in Lift zu verwenden, wird es nicht explizit vorgeschrieben. Stattdessen wird ein „View-First“-Pattern vorgeschlagen, welches es ermöglicht, Logik über sog. „Snippets“, die Objekte in Scala darstellen, in den View einzubauen. Dies soll vor allem eine für Webdesigner bessere Entwicklung von View-Komponenten ermöglichen, da die Markup-Sprache nur minimal durch Applikationslogik erweitert wird. Außerdem werden dadurch die Aufgabenbereiche von Webdesignern und Webprogrammieren besser getrennt, damit sich ein Webdesigner möglichst auf das Design und ein Webprogrammierer auf die zugrundeliegende Applikationslogik beschränken kann [24].

3 Klassifikationsmerkmale

3.1 Frameworktyp

Webframeworks können grundlegend in fünf Typen klassifiziert werden: Request-based Framework, Component-based Framework, Hybrid-Framework, Meta-Framework und RIA-based Framework. Bei einem *Request-based Framework* werden die eingehenden Anfragen direkt verarbeitet. Jede Anfrage ist prinzipiell zustandslos, was jedoch durch serverseitige Sessions in einem gewissen Maß ausgeglichen werden kann. Bei einem *Component-based Framework* wird die Logik des „Request Handling“ abstrahiert und zwecks Wiederverwendung in Komponenten gekapselt. Ein *Hybrid Framework* kombiniert Charakteristiken von Request-based und Component-based Frameworks. Durch die Verwendung von Komponenten kombiniert es den Vorteil der Wiederverwendung mit der Kontrolle über Anfragen bei einem puren Request-based Framework. Ein *Meta-Framework* kann als ein Framework von Frameworks gesehen werden. Es stellt ein Framework bereit, welches unter Verwendung des „Inversion of Control“-Prinzips andere Komponenten, Services oder Frameworks integrieren kann. Ein *RIA-based Framework* verlagert einen Großteil der Logik auf die Clientseite, um die Client-Server Kommunikation zu reduzieren. Unter Verwendung eines client-seitigen Container-Modells soll eine vor allem auf dem Browser ausgeführte Applikation bereitgestellt werden [2].

3.2 Server-Plattform

Dass die Server-Plattform, auf der ein Webframework aufbaut, ein grundlegendes Klassifizierungsmerkmal darstellt, kann man zum einen daran sehen, dass dieses Merkmal oft als übergreifender Gliederungspunkt für die Aufführung mehrerer Webframeworks herangezogen wird [25] [26], als auch an der Tatsache, dass sich viele wissenschaftliche Arbeiten auf eine Server-Plattform beschränken bzw. nach ihr differenzieren [27] [2].

Auch wenn es keine aussagekräftige Statistik für die Verbreitung von Server-Plattformen unter Webframeworks gibt, könnte man die generelle Verbreitung bestimmter Webprogrammiersprachen als Indikator hierfür sehen. Nach der Studie „State of Web Development 2010“ wurden als am weitest verbreitete Webprogrammiersprachen folgende identifiziert (Aufzählung mit abnehmender Verbreitung): PHP, ASP.NET, Ruby, Java, Python, Perl, ASP, Javascript, Coldfusion [28]. Diese Theorie wird von der gleichen, im Abschnitt 5.4.1 näher aufgeführten Studie zur Verbreitung von Webframeworks unterstützt. Für die Dominanz PHP-basierter Webframeworks sind von den acht aufgeführten vor allem das „Zend-Framework“ und „CakePHP“ verantwortlich. Zur hohen Popularität von Ruby trägt vor allem das nach der Studie populärste frei verfügbare Webframework „Ruby on Rails“ bei.

4 Häufige Funktionen

4.1 Object Relational Mapping

Eine Funktionalität, die viele Webframeworks bieten ist das sog. „Object-Relational Mapping“ (ORM) [29]. Das Grundkonzept ist, dass mit einer Datenbank nicht direkt interagiert wird sondern die Inhalte dieser auf Objekte der Programmiersprache, die für die Anwendungslogik verwendet wird, abgebildet werden [3].

Dieser Mechanismus bietet einige grundlegende Vorteile gegenüber einem direkten Datenbankzugriff. Zum einen wird die Entwicklung erleichtert, da die Verwendung einer zusätzlichen Datenbanksprache wie SQL innerhalb der Anwendung nicht mehr notwendig ist. Einen weiteren Vorteil stellt die Datenbankunabhängigkeit dar. Das Abbilden von Daten einer Datenbank auf Objekten kann unterschiedlich erfolgen, d.h. unterschiedliche Datenbanken können unterstützt werden. Dem Entwickler und der Anwendung stehen unabhängig vom Datenbanktyp immer die gleichen Objekte und der Methoden zur Verfügung. Dies ermöglicht eine einfache Portierung der fertigen Anwendung von einem Datenbanktyp zu einem anderem. Überdies können durch Caching bessere Zugriffszeiten ermöglicht werden, da im Normalfall der Zugriff auf Objekte schneller durchgeführt werden kann als der auf eine Datenbank. Es hängt allerdings von vielen Faktoren ab, ob diese Effekte eine signifikante Auswirkung haben. Dadurch, dass in populären, etablierten Frameworks oft viel Zeit in die Optimierung, z.B. des ORM, investiert wurde, kann man davon ausgehen, dass diese Algorithmen effizienter sind als selbst Implementierte [3].

Um das allgemeine Konzept von ORM in einem Webframework zu implementieren, gibt es verschiedene Architekturmuster. Vier davon häufig in Webframeworks implementierte stellen „Active Record“, „Row Data Gateway“, „Table Data Gateway“ und „Data Mapper“ dar. Sie definieren wie und welche Daten auf Objekte abgebildet werden [25] [30].

4.2 Template Engine

Viele Webframeworks beinhalten Template Engines [25]. Diese beschäftigen sich damit wie in Views (siehe MVC) Anwendungslogik eingebunden werden kann. Das grundlegende Ziel hierbei ist die Trennung von Anwendungslogik und Präsentation [31]. Eine Template Engine nimmt ein Template, beispielsweise ein HTML-Dokument, entgegen, welches bestimmte Zeichen oder Zeichenfolgen enthält, die durch Anwendungslogik ausgetauscht werden können. Diese Elemente werden dann durch den eigentlichen Code ersetzt [3]. Die durch die Verwendung von Template Engines bedingten Vorteile sind vielfältig. Beispielsweise ist eine bessere Arbeitsverteilung zwischen Webdesignern und Webprogrammierern möglich, die Wiederverwendung von Code wird verbessert und die Klarheit eines Views (siehe MVC) steigt. Daneben wird die Wartung der Anwendung erleichtert, da es immer nur einen Punkt gibt, an der ein Teil der Anwendungslogik geändert werden muss [31].

4.3 Authentifizierung

In Webanwendungen müssen sich Benutzer oft, z.B. mit einem Benutzernamen und entsprechendem Passwort authentisieren. Nach einer Authentifizierung kann eine Webanwendung einem Benutzer anhand seiner Identität Aktionen zuordnen. Die Authentifizierung sollte möglichst sicher ablaufen, beispielsweise mit Hilfe von HTTP über SSL. Damit sich ein Benutzer nur einmal während einer Sitzung autorisieren muss, sind die entsprechenden Daten idealerweise für die Dauer einer Sitzung gespeichert [32].

Alle genannten Aufgaben können nun von einem Webframework unterstützt werden, indem für Authentifizierungszwecke vorgefertigte Funktionen bereitstellt werden. Diese Funktionen können sowohl im Webframework selbst implementiert sein wie z.B. im „Zend Framework“ [33] durch die Klasse „Zend_Auth“ [34], oder müssen durch Erweiterungen hinzugefügt werden, wie es z.B. bei „Ruby on Rails“ [35][36] der Fall ist.

4.4 AJAX

AJAX (Asynchronous JavaScript And XML) ist eine weit verbreitete Technologie [27]. Sie beschreibt, wie Anfragen mithilfe von Javascript durch auf der Benutzeroberfläche ausgelöste Ereignisse asynchron per HTTP an einen Web Server geschickt werden können, ohne den beim Abschicken vorherrschenden GUI-Zustand zu beeinflussen. Die Daten können danach auf dem Web-Server verarbeitet und zurückgeschickt werden, um beispielsweise die ursprüngliche Webseite partiell zu verändern. Hierdurch muss die Webseite nicht komplett neu gerendert werden [32].

Webframeworks können Funktionen bereitstellen, um die Verwendung von AJAX in einer Webapplikation zu erleichtern. Beispielsweise kann die Handhabung von XMLHttpRequests durch vorgefertigte Helferklassen vereinfacht werden, indem nur bestimmte Parameter übergeben werden müssen. Bei der Betrachtung verschiedener Webframeworks lassen sich grundsätzlich zwei Formen der AJAX-Unterstützung differenzieren. Die erste Möglichkeit ist, dass ein Webframework selbst AJAX-Funktionalitäten implementiert [25]. Dies ist z.B. beim Java-basierten Webframework „Apache Struts“ der Fall [18]. Die zweite Möglichkeit ist die Verwendung einer externen Javascript Bibliothek. Hierbei kann zwischen der Unterstützung bestimmter Bibliotheken und einer Bibliothekenunabhängigkeit unterschieden werden. Ein Beispiel für Ersteres ist das Webframework „Ruby on Rails“ bis ausschließlich Version 3, welches seine AJAX-Funktionalitäten vor allem durch die Prototype-Bibliothek[37] und zugehörige Helferklassen implementiert [38]. Ein Beispiel für den zweiten Fall ist das PHP-basierte Webframework „Zend Framework“ [33], welches größtenteils unabhängig von bestimmten Bibliotheken ist, allerdings die Javascript-Bibliothek „Dojo“ [39] enthält, um AJAX-Funktionalitäten von vornherein einzuschließen [40].

4.5 Weitere Funktionen

Dieses Kapitel hat ohne Anspruch auf Vollständigkeit nur vier exemplarisch ausgewählte Funktionen dargestellt. Weitere Funktionen sind beispielsweise Funktionen zur Eingabeverifikation, Verarbeitung von Formulardaten, Verwaltung von Sessions oder dem Mapping von URLs [32].

5 Vergleich/Auswahl

Die immense Menge an bestehenden Webframeworks macht eine Evaluierung bestehender Lösungen unabdingbar, um zu bestimmen, welches Framework am besten für ein spezifisches Problem geeignet ist. Allerdings gibt es hierzu kein einheitliches Verfahren und der Webframework-Vergleich stellt zumindest bis jetzt keine wissenschaftlich etablierte Disziplin dar [4]. Zwei wissenschaftliche Vergleichsmodelle und vier Kriterien sollen im Folgenden betrachtet werden.

5.1 Vergleichsmodell auf Basis von Kategorien

Einen möglichen Ansatz bietet der Artikel „A Comparison Model for Agile Webframeworks“ [4]. Hier findet zunächst eine initiale Auswahl von Frameworks statt, welche miteinander verglichen werden sollen. Das Auswahlverfahren selbst wird allerdings nicht näher erläutert. Danach werden Kennfragen definiert und nach bestimmten Gruppen, welche allgemeine Funktionen oder Probleme von Webanwendungsentwicklung adressieren, untergliedert. In dem Artikel werden folgende Gruppen identifiziert: „Domain Description and Persistence“, „Presentation“, „Security“, „Usability“, „Testing“, „Service Orientation“, „Component Orientation“ und „Adoption“. Die Kennfragen werden nun für die jeweiligen Webframeworks hinsichtlich ihrer Erfüllung bewertet. Das Ergebnis kann letztendlich in einer Matrix dargestellt werden, um einen Überblick über die bewerteten Webframeworks zu erhalten und eine Auswahl zu treffen. Für einen aggregierten Überblick über die evaluierten Daten erfolgt zudem eine Visualisierung mittels eines Kiviat-Diagramms [4].

5.2 Vergleichsmodell auf Basis der Goal Question Metric Methode (GQM)

Ein weiterer Ansatz wird in „An Evaluation of AJAX-enabled Java-based Web Application Frameworks“ beschrieben. Aufgrund der großen Menge existierender Frameworks findet auch hier eine Vorauswahl statt, im Fall des Artikels nach Programmiersprache, Frameworktyp, verwendeten Technologien und Etablierung. Gemäß der Goal Question Metric Methode (GQM) werden Ziele, Fragen und messbare Werte definiert. Anschließend wird ein Scoringmodell mit den Kriterien „Product Maturity“, „Activity of the Community“, „License Type“, „Security“, „Scalability“, „Available Debugging Tools“, „AJAX Support“ und „Level of Documentation“ erstellt. Die Kriterien werden gewichtet und anschließend bewertet.

5.3 Bewertung der Vergleichsmodelle

Beide Ansätze bieten systematische und wissenschaftliche Ansätze ein Webframework auszuwählen. Es werden Kriterien definiert, bewertet und aufgrund der Daten eine Auswahl getroffen. Letzterer scheint allerdings den Vorteil zu haben, sich durch die GQM Methode besser auf einen bestimmten Anwendungsfall zu beziehen und durch die Gewichtung der Kriterien präziserer Ergebnisse zu liefern.

5.4 Vergleichs-/ Auswahlkriterien

5.4.1 Popularität

Die Popularität spielt eine wichtige Rolle für ein Webframework, da durch diese andere Faktoren wie bei einem Open-Source Framework die Größe und Aktivität der Community abgeschätzt werden kann, was Einfluss auf die Weiterentwicklung eines Frameworks haben kann. Zudem ist es denkbar, dass durch eine erhöhte Popularität eine größere Anzahl von Entwicklungswerkzeugen unterstützt wird. Allerdings ist es schwer, solch einen Eigenschaft zu quantifizieren.

Einen Versuch unternimmt die Studie „State of Web Development“, welche jährlich Umfragen durchführt, um den aktuellen Stand von Webtechnologien und Praktiken zu untersuchen. Dabei wurde in der Studie von 2010 die folgenden populärsten Webframeworks festgestellt (Aufzählung mit absteigender Popularität): Ruby on Rails, Django, Zend Framework, CakePHP, Apache Struts, node.js, CherryPy, Ext GWT. Ein offensichtlicher Nachteil dieser Methode ist allerdings, dass an der Umfrage nur 1420 sich selbst ausgewählte Personen teilgenommen haben und sie damit nicht repräsentativ ist [28] [41].

Einen weiteren Versuch die Popularität von Webframeworks zu quantifizieren unternimmt HotFrameworks.com [26] das Frameworks anhand von drei technischer Faktoren bewertet. Zum einem wird die Anzahl von „Github Watchers“ berücksichtigt. Die Annahme hier ist, dass ein Webframework umso populärer ist, je mehr „Github Watcher“ es gibt, d.h. Personen die am Source Code des Frameworks auf der Code Hosting Plattform Github [42] interessiert sind. Als zweites Kriterium wird der Traffic auf den Webauftritten der Webframeworks berücksichtigt, d.h. ein Frameworks ist umso populärer, je mehr Traffic die dazugehörige Webseite aufweist. Das dritte Kriterium ist die Anzahl von Links, die auf den Webauftritt des Frameworks verweisen. Nach diesem Kriterium ist ein Framework umso populärer, je mehr Links auf die entsprechende Webseite verweisen. Die Kriterien werden auf einer Skala von 0-100 bewertet um einen Gesamtscore zu ermitteln, nach welchem alle Frameworks der Popularität nach geordnet werden. Ein Nachteil dieser Methode besteht darin, dass für manche Kriterien keine Werte ermittelt werden können, da Frameworks beispielsweise nicht auf Github vertreten sind oder keine eigene Domain aufweisen, was die Ermittlung des Traffics verhindert [43]. Auch wenn dieses Verfahren den Vorteil hat, weniger subjektiv zu sein als Umfragen, ist dennoch fraglich inwieweit die verwendeten Faktoren die wahre Popularität wiederspiegeln.

5.4.2 Lizensierung

Hinsichtlich der Lizenzierung können Webframeworks in zwei Gruppen eingeordnet werden - Open-Source und proprietär. Der grundlegende Vorteil einer Open-Source-Lizenzierung liegt in der Absenz von Lizenzkosten. Die verschiedenen Lizenzkonditionen, wie das häufig geforderte Offenlegen veränderten Codes dürfen jedoch in der Auswahl nicht vernachlässigt werden. Bei proprietären Webframeworks fallen hingegen Lizenzkosten an. Neben den Kosten müssen hier ebenfalls die Lizenzbedingungen, beispielsweise zusätzliche Kosten bei dem Vertrieb von mit dem Framework erstellten Anwendungen berücksichtigt werden [3].

5.4.3 Dokumentation

Weiterhin stellt die Verfügbarkeit und Qualität einer Dokumentation ein Auswahlkriterium dar. Einige wichtige Elemente einer Dokumentation für Webframeworks sind eine Liste von Funktionen, ein „Cook Book“, eine Design-Dokumentation. Dabei gibt die Liste von Funktionen einen High-Level-Überblick über alle Funktionalitäten eines Webframeworks. Das „Cook Book“ stellt eine ausführliche Schritt-für-Schritt-Beschreibung dar, welche die Erstellung einer Applikation mit dem Webframework beschreibt. Die Design-Dokumentation umfasst Beschreibung von verwendeten Design Patterns, den Grund für die Verwendung eben dieser und allgemeine designbezogene Fragestellungen [3].

5.4.4 Support

Im Bezug auf den Support ist es wichtig Informationen dahingehende zu evaluieren, ob und in welcher Höhe Kosten anfallen, wer den Support leistet, welche Qualität dieser hat, über welche Kanäle er abgewickelt wird oder wie lange die Bearbeitung von Supportanfragen dauert [3].

6 Trends und Entwicklungen

6.1 Striktere Trennung von Anwendungslogik und Präsentation

Die Trennung von Anwendungslogik und Präsentation bei Webframeworks ist ein Problem mit Verbesserungspotenzial. Derzeit gibt es kein Webframework, welches solch eine Trennung auf dem Modullevel vornimmt, was bedeutet, dass Module vollkommen unabhängig sowohl durch Webdesigner als auch durch Webprogrammierer erstellt werden können. Auch MVC-Webframeworks, welche idealerweise durch die Architektur bedingt Anwendungslogik von Präsentation trennen sollen, erreichen dies nicht vollständig. Eine in [15] vorgeschlagene Lösung ist die bessere Integration von Webdesignwerkzeugen mit den jeweiligen Frameworks. Zum einen wäre hierdurch der Webdesigner in der Lage, ohne die Hilfe

von Webprogrammierern Präsentationsteile einer Applikation zu realisieren. Zum anderen könnte er, entsprechende Integration von Webdesignwerkzeug und Framework vorausgesetzt, eine bessere Vorschau über das finalen Ergebnis erhalten, da Daten aus dem Framework in eine Designvorschau eingebunden werden könnten [15].

6.2 Einfachheit

Ein weiterer Aspekt, den man in zukünftigen Webframeworks sehen könnte, ist die einfacheren Handhabung und Implementierung, sowohl für Webdesigner als auch für Webprogrammierer. Webframeworks stehen vor der Herausforderung, eine einfache Anwendung ohne Flexibilitätseinbußen zu ermöglichen. Dies könnte zukünftig beispielsweise durch eine striktere Vorschreibung von Design-Prinzipien bewältigt werden könnte. Solche können zwar heute schon in Webframeworks implementiert werden, sind allerdings nicht explizit vorgeschrieben [15].

6.3 Mobile Web Application Frameworks

Einen möglichen Trend könnten „Mobile Web Application Frameworks“ darstellen. Dies sind Frameworks, die auf die Nutzung der endgültigen Anwendung auf einem mobilen Endgerät ausgerichtet sind. Allerdings gibt es zu diesem Thema noch wenig bis keine wissenschaftliche Literatur. Einerseits könnte man diesen Trend allerdings durch das allgemein große Wachstum mobiler Internetnutzung [44] und der damit verbunden Nachfrage nach speziell für diesen Bereich vorgesehenen Frameworks unterstützt sehen, andererseits gibt es bereits einige verfügbare Lösungen wie JQTouch [45], JQuery Mobile [46] oder SenchaTouch [47]. Während diese Lösungen von Grund auf für mobile Endgeräte entwickelt sind, gibt es auch erste Ansätze, etablierte Webframeworks um Funktionen hinsichtlich mobiler Nutzung zu optimieren. Ein solcher Ansatz stellt das „Zend Framework“ [33] dar, welches Funktionen zur Erkennung von Hardwarefunktionen und der Art eines mobilen Geräts bereitstellt und aufgrund dieser Daten beispielsweise das Layout einer Webanwendung anpassen kann [48]. Inwieweit solche Lösungen native Applikationen ablösen können oder welche Rolle sie in der Zukunft spielen, bleibt allerdings abzuwarten.

7 Literaturverzeichnis

1. Gamma, E., Helm, R., Johnson, R., Vlissides, J.M.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional (1994).
2. Shan, T.C., Hua, W.W.: Taxonomy of Java Web Application Frameworks. Proceedings of the IEEE International Conference on e-Business Engineering. S. 378–385. IEEE Computer Society, Washington, DC, USA (2006).

3. Nash, M.: Java Frameworks and Components: Accelerate Your Web Application Development. Cambridge University Press (2003).
4. Ignacio Fernández-Villamor, J., Díaz-Casillas, L., Iglesias, C.Á.: A comparison model for agile web frameworks. Proceedings of the 2008 Euro American Conference on Telematics and Information Systems. S. 14:1–14:8. ACM, New York, NY, USA (2008).
5. Hunt, A., Thomas, D.: The Pragmatic Programmer: From Journeyman to Master. Addison-Wesley Professional (1999).
6. Kristaly, D.M., Sisak, F., Truican, I., Moraru, S.-A., Sandu, F.: Web 2.0 technologies in web application development. Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments. S. 65:1–65:4. ACM, New York, NY, USA (2008).
7. Drupal - Open Source CMS, <http://drupal.org/>.
8. The Drupal overview, <http://drupal.org/getting-started/before/overview>.
9. Szyperski, C.: Component Software: Beyond Object-Oriented Programming. Addison-Wesley Professional (2002).
10. Pree, W.: Meta Patterns - A Means For Capturing the Essentials of Reusable Object-Oriented Design. Proceedings of the 8th European Conference on Object-Oriented Programming. S. 150–162. Springer-Verlag, London, UK (1994).
11. Greenfield, J., Short, K., Cook, S., Kent, S., Crupi, J.: Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools. Wiley (2004).
12. Larman, C.: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Prentice Hall (2004).
13. Inversion of Control Containers and the Dependency Injection pattern, <http://martinfowler.com/articles/injection.html>.
14. Corcho, O., López-Cima, A., Gómez-Pérez, A.: A platform for the development of semantic web portals. Proceedings of the 6th international conference on Web engineering. S. 145–152. ACM, Palo Alto, California, USA (2006).
15. Shklar, L., Rosen, R.: Web Application Architecture: Principles, Protocols and Practices. Wiley (2003).
16. Kappel, G., Pröll, B., Reich, S.: Web Engineering: The Discipline of Systematic Development of Web Applications. John Wiley & Sons (2006).
17. Apache Struts, <http://struts.apache.org/>.
18. Apache Struts - Key Technologies Primer, <http://struts.apache.org/primer.html>.
19. ASP.NET Web Forms MVP, <http://webformsmvp.com/>.
20. ASP.NET MVVM, <http://aspnetmvvm.codeplex.com/>.
21. Kohana: The Swift PHP Framework, <http://kohanaframework.org/>.
22. Interactive Application Architecture Patterns, <http://www.aspiringcraftsman.com/2007/08/25/interactive-application-architecture/>.
23. Lift :: Home, <http://liftweb.net/>.
24. View First, http://www.assembla.com/spaces/liftweb/wiki/View_First.
25. Comparison of Web application frameworks - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks.

26. Web framework rankings, <http://hotframeworks.com/>.
27. Laakso, T., Niemi, J.: An evaluation of AJAX-enabled java-based web application frameworks. Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia. S. 431–437. ACM, Linz, Austria (2008).
28. Server-side Technologies, <http://www.webdirections.org/sotw10/server/>.
29. Jazayeri, M.: Some Trends in Web Application Development. 2007 Future of Software Engineering. S. 199–213. IEEE Computer Society, Washington, DC, USA (2007).
30. Fowler, M.: Patterns of Enterprise Application Architecture. Addison-Wesley Professional (2002).
31. Parr, T.J.: Enforcing strict model-view separation in template engines. Proceedings of the 13th international conference on World Wide Web. S. 224–233. ACM, New York, NY, USA (2004).
32. Vosloo, I., Kourie, D.G.: Server-centric Web frameworks: An overview. ACM Comput. Surv. 40, 4:1–4:33 (2008).
33. Zend Framework, <http://framework.zend.com/>.
34. Zend Framework: Documentation: Zend_Auth - Zend Framework Manual, <http://framework.zend.com/manual/de/zend.auth.html>.
35. Ruby on Rails, <http://rubyonrails.org/>.
36. How-to: Authentication and Authorization in Ruby on Rails, <http://wiki.rubyonrails.org/howtos/authentication-authorization>.
37. Prototype JavaScript framework: Easy Ajax and DOM manipulation for dynamic web applications, <http://www.prototypejs.org/>.
38. Module: ActionView::Helpers::PrototypeHelper, <http://api.rubyonrails.org/classes/ActionView/Helpers/PrototypeHelper.html>.
39. Unbeatable JavaScript Tools - The Dojo Toolkit, <http://dojotoolkit.org/>.
40. Dojo and Zend Framework Partnership Announcement, <http://devzone.zend.com/article/3545>.
41. The State of Web Development 2010, <http://www.webdirections.org/sotw10>.
42. Secure source code hosting and collaborative development - GitHub, <https://github.com/>.
43. Frequently asked questions about web framework rankings, <http://hotframeworks.com/faq>.
44. Meeker, M., Devitt, S., Wu, L.: Internet Trends, http://www.morganstanley.com/institutional/techresearch/internet_trends042010.html, (2010).
45. jQTouch, <http://jqtouch.com/>.
46. jQuery Mobile, <http://jquerymobile.com/>.
47. Sencha Touch, <http://www.sencha.com/products/touch/>.
48. Zend News - Zend Framework 1.11 unterstützt mobile Endgeräte - Zend.com, <http://www zend com/de/company/news/news-links/zend-framework-1-11-unterstzt-mobile-endger-te>.

Testen von Webanwendungen

Steffen Witt

Betreuer: Prof. Dr. Andreas Henrich
Dipl.-Wirtsch.Inf. Stefanie Sieber

Zusammenfassung Diese Arbeit stellt kurz die Besonderheiten von Webanwendungen dar und zeigt, welche Folgen diese für das Testen von Webapplikationen haben. Danach werden einzelne Testmöglichkeiten der nicht-funktionalen Anforderungen Usability, Barrierefreiheit und Belastbarkeit erläutert. Zum Test von funktionalen Anforderungen wird das Vorgehensmodell Test-Driven Development vorgestellt.

1 Einleitung

Web-Anwendungen nehmen in ihrer Bedeutung für das Leben des Einzelnen fortwährend zu. Ausgehend von statischen Webseiten haben sich im Laufe der Zeit immer komplexere Applikationen im Web entwickelt. Mittlerweile sind selbst umfangreiche Office Lösungen schon als Webanwendung verfügbar. Prominentestes Beispiel hierfür ist Google Docs, aber auch Microsoft stellt mit Office Live eine Variante seiner Office Suite online zur Verfügung. Beide Anwendungen sind darüber hinaus auch noch kostenlos zu benutzen. Das führt dazu, dass Webapplikationen für einen immer größeren Kreis an Anwendern interessant werden. Um der größer werdenden Anzahl an Nutzern ein, hinsichtlich Bedienung, Funktion und Performanz, zufriedenstellendes Nutzungserlebnis zu ermöglichen ist es daher von elementarer Bedeutung, dass diese Webanwendungen vorher umfangreich auf jegliche Arten von Anforderungen getestet werden.

Ziel dieser Arbeit ist es daher einen Überblick darüber zu geben, was es beim Testen von Web-Anwendungen zu beachten gibt. Dazu werden zu Beginn dieser Arbeit Besonderheiten von Webanwendungen aufgezeigt, welche verdeutlichen sollen, dass man zum Testen von Webanwendungen spezielle Methoden benötigt. Ebenso wird auf die, sich aus den Besonderheiten direkt ergebenden Folgen, für das Testen von Webapplikationen eingegangen.

Daraufhin werden die nicht-funktionalen Anforderungen Usability, Barrierefreiheit und Belastbarkeit betrachtet. Diese exemplarisch ausgesuchten Eigenschaften werden kurz definiert und danach werden für alle verschiedene Methoden gezeigt, um diese zu prüfen. Außerdem wird eine kurze Anleitung gegeben wie jeder Webseiten Betreiber Usabilityaspekte seiner Webseite prüfen kann.

Im darauf folgenden Abschnitt wird dann der Fokus auf die funktionalen Anforderungen gerückt. Hierfür wird das Vorgehensmodell des Test-Driven Developments dargestellt. Abschließend wird noch eine Bewertung der vorgestellten Möglichkeiten bezüglich ihrer praktischen Relevanz vorgenommen.

2 Besonderheiten von Web-Anwendungen

Dieses Kapitel soll beleuchten welche Besonderheiten Webapplikationen ausmachen und zu welchen Folgen diese beim Testen führen. So sieht [6, S. 167-170] folgende Besonderheiten:

Hohe Änderungsdynamik Webanwendungen unterliegen häufig kleinen Änderungen die oft in sehr kurzen Abständen hintereinander online geschaltet werden. Besonders ausgeprägt ist dies bei Projekten die das Konzept der „*Perpetual Beta*“ befolgen. Auch die Infrastruktur auf der die Anwendungen gehostet werden kann sich schnell ändern, insbesondere wenn man hierfür die Dienstleistungen eines Dritten in Anspruch nimmt. Um der Änderungen hoher Dynamik gerecht zu werden ist es notwendig möglichst viele Tests zu automatisieren.

Nicht automatisiert prüfbare Inhalte Da die Inhalte von Webapplikationen oft informelle Texte sind, ist es häufig auch nicht mögliche diese vollständig automatisiert zu prüfen. Deshalb ist hierfür ein erhöhter personeller Korrekturaufwand nötig.

Vielzahl an Endgeräten Webanwendungen werden auf einer Vielzahl von Endgeräten betrachtet, welche sich durch Merkmale wie Bildschirmauflösung und -größe, installierte Browser und Plugins unterscheiden. Um sicherzustellen dass eine Seite überall korrekt dargestellt würde, bräuchte man theoretisch auch alle möglichen Geräte.

Vielzahl an Komponenten Bei Webapplikationen finden sich häufig viele verschiedene Komponenten wieder. Deren Zusammenspiel muss mit Integrations- tests getestet werden. Dies bezieht sich nicht nur auf die für den Anwender versteckte Kommunikation zwischen Serverkomponenten wie dem Http-Server und dem Datenbankserver, sondern auch auf Fremdinhalte die über entsprechende zur Verfügung gestellte APIs eingebunden werden.

Immaturität Im Bereichs des Webs erscheinen häufig neue Technologien auf die man zurückgreifen kann. Allerdings stehen für diese nicht immer auch sofort Testframeworks zur Verfügung, was automatisierte Tests erschwert.

Globalität Webanwendungen sind aus der gesamten Welt abrufbar und erreichen somit auch verschiedene Kulturreiche. Möchte man weltweit Nutzer mit seiner Anwendung ansprechen ist es notwendig sich der kulturellen Unterschiede bewusst zu sein und diese auch bei Usability Tests zu berücksichtigen.

3 Testen nicht funktionaler web-spezifischer Anforderungen

Im Gegensatz zu herkömmlichen Anwendungen spielen im Web die nicht-funktionalen Anwendungen eine viel höhere Rolle (vgl. [6, S. 184]), da die Nutzer eine andere Erwartungshaltung haben. Will man Nutzer gewinnen so muss man nicht nur mit Funktionen überzeugen sondern ihm auch ein angenehmes Nutzungserlebnis bieten. Daher werden in diesem Abschnitt exemplarisch für die nicht-funktionalen Anforderungen die für Nutzer besonders wichtigen Aspekte der Usability, der Barrierefreiheit und der Belastbarkeit näher betrachtet und jeweils verschiedene Testmethoden erläutert.

3.1 Usability

Usability ist ein für den Erfolg von Webapplikationen entscheidendes Merkmal, denn wenn diese nicht gut ist wird ein Nutzer eine Anwendung nicht benutzen. Demzufolge wird er auch nicht bereit sein Geld für die Nutzung zu zahlen, respektive im Falle kostenlos angebotener Anwendungen wird er keine Werbeeinnahmen mehr generieren wenn er die Seite nicht mehr nutzt. Deshalb ist es notwendig als Entwickler die Usability stets zu testen. Hierfür muss man sich im Klaren sein was Usability alles beinhaltet. Die Norm ISO 9241-110 (vgl.[5]) nennt folgende Aspekte:

- Aufgabenangemessenheit
- Selbstbeschreibungsfähigkeit
- Erwartungskonformität
- Fehlertoleranz
- Steuerbarkeit
- Individualisierbarkeit
- Lernförderlichkeit

Um diese zu Testen gibt es verschiedene Arten von Testmethoden die nun vorgestellt werden sollen.

3.1.1 Expertentest Um die Usability der eigenen Anwendung zu testen kann man als einfachste Methode auf einen externen Gutachter zurückgreifen. Diese Variante ist besonders geeignet für Firmen denen das Know-How zur Usability-evaluation fehlt und die schnell ein Expertise benötigen (vgl.[8, 206f]). Weniger geeignet ist diese Methode jedoch bei Anwendungen die domänen spezifisches Wissen benötigen, da der externe Gutachter sich dieses auch erst aneignen müsste, womit einerseits die Kosten steigen würden und man andererseits auch kein schnelles Ergebnis mehr hätte.

3.1.2 Befragung Eine naheliegende Methode zur Beurteilung von Usability ist die der Befragung. Dabei sollte man idealerweise diejenigen Befragen die auch die Applikation benutzen und dies auch schon während der Entwicklung um Fehlentwicklungen frühzeitig vorzubeugen. Bei den Befragungen kann man im wesentlichen zwischen Folgenden beiden Arten unterscheiden.

Fragebögen/Web-Formulare Fragebögen bieten die Möglichkeit relativ einfach schnell viele Befragungen durchzuführen. Sofern man ein geschlossenes Antwortformat, d.h. keine freien Texte zulässt, sondern beispielsweise Multiple-Choice Fragen stellt, lassen sich diese auch schnell auswerten (vgl. [11]). Außerdem erreicht man durch die Anonymität häufig auch weniger verfälschte Antworten als bei einer direkten Befragung. Eine weitere Möglichkeit ist es diese Befragungen direkt online in der Anwendung durchzuführen. Damit erreicht man dann auch die wirkliche Nutzergruppe, welche allerdings von der eigentlich angedachten Zielgruppe differieren kann. Gemäß [8, S. 203] sollten sich um einen höheren Rücklauf zu gewährleisten die Fragen auch direkt auf den präsentierten Seiteninhalt beziehen. So bekommt der Nutzer das Gefühl, dass er mit seiner Mithilfe direkt das Angebot beeinflussen kann.

Interviews Im Gegensatz zu Fragebögen hat der Befragte bei Interviews direkten Kontakt zum Interviewer. [11] sieht hier verschiedene Vorteile. Direkte Interviews lassen sich direkt auf relevante Zielpersonen einschränken, wodurch auch die Relevanz der Antworten steigt. Durch das direkte Gespräch lassen sich hier auch eventuelle Unklarheiten bei den Fragen beseitigen. Nachteilig hingegen sind der höhere Zeitaufwand den sowohl die Befragung als auch die Auswertung mit sich bringt. Ferner wird durch die Anwesenheit des Interviewers das Antwortverhalten des Befragten beeinflusst, was zu verzerrten Ergebnissen führen kann.

3.1.3 Beobachtung Neben der Befragung von Nutzern gibt es auch die Möglichkeit diese bei der Benutzung der eigenen Webseite zu beobachten. An dieser Stelle sollen hierfür zwei Möglichkeiten vorgestellt werden. Zum einen Eye-Tracking Verfahren und Auswertungen mittels Web-Analyse Werkzeugen.

Eye-Tracking Verfahren Das Aufzeichnen von Bewegungen und Fixationen des menschlichen Auges bietet auch Einsatzmöglichkeiten zu Usabilitytestzwecken. Dazu gibt man Probanden die Aufgabe ein bestimmtes Ziel auf einer Webseite zu erreichen (vgl. [8, S. 208]). Hierbei ergeben sich so genannten Heat-Maps die aufzeigen welche Stellen der Anwender fixiert hat. Dadurch kann man beispielsweise erkennen, ob die Platzierung von Objekten der Erwartungshaltung des Nutzers entspricht oder ob dieser erst andere Orte fixiert. Die so gewonnenen Erkenntnisse kann ein Entwickler dann zur Umgestaltung seiner Webseite nutzen. Das Eye-Tracking ist allerdings ein aufwendiges und teures Verfahren. Zudem könnten sich Verfälschungen im Verhalten der Probanden durch die Laborsituation ergeben.

Ein gutes Beispiel für dieses Verfahren ist eine Untersuchung des US Zensusamtes (vgl. [7]). Ein Ziel dessen Webseite ist es, den Besucher über die aktuelle Bevölkerungszahl der Vereinigten Staaten zu informieren. Dies wurde mit der

Darstellung selbiger mit einer großen roten Zahl rechts oben auf der Startseite umgesetzt. Allerdings ergab die Untersuchung das lediglich 14% der Nutzer diese Zahl ausfindig machen konnten. Eine Heat-Map hierzu sah folgendermaßen aus:

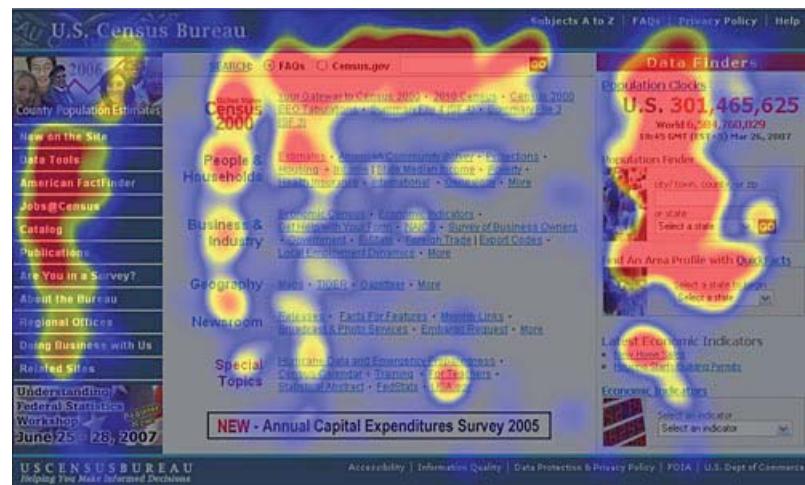


Abbildung 1. Quelle: [7]

Bei der Auswertung stellte man fest dass die Nutzer lediglich die ersten 3 Stellen der Bevölkerungszahl betrachteten. Die restlichen Stellen wurde nur von einer sehr geringen Anzahl an Nutzern fixiert. Dies führte man auf den Effekt der Banner Blindness zurück, das heißt dass die Nutzer die Zahl für einen Werbebanner hielten und ihm deshalb keine weiteren Beachtung zeigten. Somit stellte sich der Gedanke der Designer, dass große rote Zahlen auffällig sind und daher auch gut wahrgenommen werden müssten, als falsch heraus.

Web Analytics Eine deutlich einfachere Methode Webseiten auf Usability zu untersuchen stellen Webanalysewerkzeuge dar. Diese Werten automatisiert das Verhalten und die Umgebung der Nutzer einer Webseite aus. Die bekannteste Lösung hierfür ist Google Analytics, bei der die gesamte Auswertung auf den Servern von Google läuft. Möchte man dies nicht, so kann man auch auf Lösungen zurückgreifen wie die Open-Source Software Piwik, die die Datenhaltung auf dem eigenen Server vornehmen. Hauptanwendungszweck der Webanalysetools ist die Auswertung des Werbeerfolgs auf einzelnen Seiten. Darauf informieren die Werkzeuge mit Kennzahlen wie Werbelinkklickraten. Außerdem können die von ihnen gelieferten Kennzahlen als Grundlage für die Konzeption weiterer Tests dienen. Beispielsweise könnte man entnehmen welche Browser und welche Bildschirmauflösungen die Besucher wirklich benutzen, um die Testfälle dann auf diese auch zu beschränken.

Eine weitere Möglichkeit Web Analytics einzusetzen sind die A/B-Tests. Hierzu erstellt man zwei Varianten einer Webseite und teilt die Webseitenbesucher auf die beiden Alternativen auf. Dabei misst man wie lange die Nutzer jeweils die Webseite betrachten und wieviele Seiten sie insgesamt aufrufen. Daraus versucht man dann zu schließen welche Alternative aus Usabilitysicht die bessere ist. Dies soll nun anhand eines Beispiels verdeutlicht werden.

Gegeneinander getestet wurden zwei verschiedene Schriftarten auf der Seite www.sdc-ebern.de - einem gewöhnlichen Wordpress Blog - mit Hilfe von Google Analytics. Dazu wurde das vorhandene Stylesheet kopiert und mit geänderter Schriftart unter neuen Namen hochgeladen. Die Folgenden Code-Änderungen wurden alle samt in der header.php des aktivierten Themes vorgenommen. Zunächst wurde, damit ein Benutzer während einer Sitzung auch nur eine Variante zu Gesicht bekam, ein Cookie angelegt. Dies erfolgt mit folgendem Code ganz zu Beginn der header.php:

```
<?php
    ini_set("session.cookie_domain",
        ".sdc-ebern.de");
    session_start();
?>
```

Danach wurde die Weiche eingebaut welche auf die alternative Style-Sheet Datei verweist. Dies wurde im Code direkt nachdem die ursprüngliche Stylesheet Datei eingebunden wurde eingebaut. Dabei wurde geschaut ob die font Variable in der Sitzung schon angelegt wurde. War dies nicht der Fall wird sie mit zufällig auf 'alternativ' bzw. 'original' gesetzt. War die Variable nun auf 'alternativ' gesetzt wurde die ursprüngliche Stylesheet Datei durch Einbinden der alternativen Datei überschrieben.

```
<?PHP
if(!$_SESSION['font']){
    if(rand(2,3) == 2){
        $_SESSION['font'] = 'alternativ';
    }
    else{
        $_SESSION['font'] = 'original';
    }
}
if($_SESSION['font'] == 'alternativ'){
?>
    <link rel="stylesheet" href="STYLESHEET" type="text
        /css" />
<?PHP
}
?>
```

Schließlich muss man noch den Code, den Google Analytics stellt, direkt nach dem öffnenden <body>-Tag einfügen und diesen um die benutzerdefinierte Variable

erweitern. Der Code von Google legt eine Queue an in die man in der 4. Zeile die eben angelegte Variable, die aus der Session ausgelesen wird, schreibt. Wichtig ist zu beachten, dass man gemäß aktueller Rechtsprechung bei Verwendung von Google Analytics einen Nutzungshinweis ins Impressum schreiben sollte, um kostenpflichtigen Abmahnungen vorzubeugen (vgl. [3]).

```
<script type="text/javascript">
var _gaq = _gaq || [];
_gaq.push(['_setAccount', 'XX-XXXXXXX-X']);
_gaq.push(["_setCustomVar", 2,"Schriftart","<?php echo $_SESSION['font'];?>",1]);
_gaq.push(['_trackPageview']);
(function() {
    var ga = document.createElement('script'); ga.type
        = 'text/javascript'; ga.async = true;
    ga.src = ('https:' == document.location.protocol ?
        'https://ssl' : 'http://www') + '.google-
        analytics.com/ga.js';
    var s = document.getElementsByTagName('script')[0];
    s.parentNode.insertBefore(ga, s);
})();
</script>$
```

Nun muss man sich ein paar Tage gedulden, bis die benutzerdefinierte Variable in Google Analytics erscheint. Im Beispielversuch ergaben sich folgende Ergebnisse:

Variable	Besucher	Seiten/Besuch	Ø-Besuchszeit	Absprungrate
alternativ	46	4,24	00:04:53	19,57%
original	15	5,13	00:03:07	13,33%

Leider lassen sich aus diesen Zahlen keine signifikanten Schlüsse ziehen, welche der beiden Varianten vorzuziehen ist. Das Besucheraufkommen der betrachteten Seite ist dafür einfach zu gering.

3.2 Barrierefreiheit

Das Internet erreicht immer größere Anteile der Bevölkerung. Dazu gehören auch Menschen mit Behinderung für die es essentiell ist das Webseiten barrierefrei gestaltet sind. Das World Wide Web Consortium definiert Barrierefreiheit im Kontext des Webs folgendermaßen:

„Web accessibility means that people with disabilities can use the Web.
More specifically, Web accessibility means that people with disabilities

can perceive, understand, navigate, and interact with the Web, and that they can contribute to the Web. Web accessibility also benefits others, including older people with changing abilities due to aging.“[10]

Zur barrierefreien Nutzung des Internets gehört demnach das Web wahrnehmen, verstehen, navigieren, damit interagieren und dazu beitragen zu können. Um das zu gewährleisten muss ein Angebot darauf getestet werden. Dieser Abschnitt befasst sich mit den dazu vorhandenen Methoden.

3.2.1 Test mit Screenreader Anwendungen mit einem Screenreadern auf Barrierefreiheit zu testen stellt eine mögliche Herangehensweise dar. Die dazu nötige Software ist teilweise auf Betriebssystemen schon vorinstalliert, wie VoiceOver bei Mac OS X und iOS, oder man kann kostenlose Open-Source Lösungen wie NVDA für Windows oder Orca für Linux benutzen. Allesamt benötigen allerdings zur vollständigen Beherrschung eine lange Einarbeitungszeit. Etwas leichter fallen erste Versuche mit VoiceOver auf iOS Geräten. Würde man die Einarbeitungszeit in Kauf nehmen, könnte man theoretisch zumindest auf Barrierefreiheit für Sehbehinderte prüfen. Allerdings ist dies auch nur ein Aspekt der Barrierefreiheit. Dennoch könnten Versuche mit Screenreadern zu testen zumindest ein Bewusstsein für die Probleme, die durch nicht barrierefrei gestaltete Anwendungen entstehen, schaffen.

3.2.2 Vollautomatisierte Tests Webanwendungen wie WAVE und Cynthia Says bieten Entwicklern die Möglichkeit ihre Seiten vollautomatisiert zu prüfen. Diese funktionieren ähnlich dem bekannten HTML-Validatoren des World Wide Web Consortiums. Mit ihnen werden jedoch fast ausschließlich formal-technische Eigenschaften geprüft (vgl. [4, S. 53]). Sprich ob beispielsweise Alternative Texte vorhanden sind. Was sie nicht oder nur sehr eingeschränkt prüfen können ist, ob diese auch inhaltlich richtig sind. Allerdings gibt z.B. Cynthia Says Hinweise falls ein und derselbe Linktext auf verschiedene Seiten führen.

3.2.3 Checklisten Mit Checklisten bekommen Entwickler einen guten Einstiegspunkt, um ihre Anwendungen zu testen. Im Gegensatz zu den vollautomatisierten Tests kann man mit ihnen auch auf inhaltliche Aspekte eingehen. Jedoch benötigt man um sie sinnvoll einzusetzen ein gutes Verständnis für die einzelnen Punkte (vgl. [4, S. 21]). Daher sollten die verwendeten Checklisten stets umfangreiche Erläuterungen für die einzelnen Prüfaspkte mitbringen. Bei der Accesibility Checkliste 2.0 der Stiftung zur behindertengerechten Technologienutzung und der Checkliste der Universität Erlangen-Nürnberg stehen jeweils ausführliche Erläuterungen zur Verfügung. Um sich einen schnellen Überblick zu verschaffen was man als Entwickler dringend beachten sollten, bieten sich folgende KO-Kriterien der Checkliste der Universität Erlangen-Nürnberg an:

- Valides HTML
- Auch ohne Stylesheets nutzbar

- Variable Schriftgrößen
- Aussagekräftige Alternativtexte für Grafiken
- Aussagekräftige Links, ggf. mit Dateitypbezeichnungen
- Mindestens 80 Punkte im BITV-Test

3.2.4 Expertentest Auch für Barrierefreiheitstests gibt es die Möglichkeit sich von Experten ein Gutachten erstellen zu lassen. Ein solches Angebot ist der BITV-Test des Projekts BIK (Barrierefrei informieren und kommunizieren), welches Webseiten auf die Kompatibilität mit der Barrierefreie Informationstechnik-Verordnung prüft. Hierbei erhält man eine umfassende Prüfung, die auch ein Gutachten mit Verbesserungspotentialen liefert sowie eine Zertifizierung auf einer Skala von 0 bis 100. Wie oben bereits erwähnt setzte die Universität Erlangen-Nürnberg für ihre Webanwendung bei diesem Test ein Ergebnis von mindestens 80 Punkten voraus um Anwendungen online zu stellen. Problematisch hierbei ist jedoch das nicht alle Seiten einer Webseite geprüft werden, sondern nur 3 Seiten stichpunktartig ausgewählt werden. Außerdem fallen hierbei Kosten an, im Falle des Test durch das BIK-Projekt mindestens 1200€ (vgl. [1]) für einen abschließenden Test.

3.3 Belastbarkeit

Die verschiedenen Varianten der Belastbarkeitstests ähneln sich alle in ihrem Grundablauf. So wird für alle Varianten ein Skript erstellt, das mehrere Anfragen von simulierten Benutzern stellt (vgl. [6, S. 179]). Dabei gilt es zu beachten das die Zusammensetzung der gestellten Anfragen in etwa der im realen Betrieb entspricht. Während das Skript durchläuft misst man Datendurchsatz sowie die Antwortzeiten. Die Ziele der einzelnen Test hingegen unterscheiden sich folgendermaßen(vgl. [6, 179f]):

Lasttest Ziel des Last Tests ist es zu validieren, ob die Antwortzeiten und der Datendurchsatz den geforderten Anforderungen genügen.

Stresstest Durch Stresstest will man prüfen, wie sich das System verhält wenn die Belastung die Anforderungen übersteigt. Idealerweise sollte das System Fehlermeldungen liefern wenn Anfragen nicht mehr bearbeitet werden können. Abstürze hingegen sollten nicht auftreten. Außerdem wird getestet wie lange das System nach einer Stresssituation benötigt um wieder normal zu laufen.

Dauertest Dauertests sollen Fehler aufdecken die erst nach längerer Laufzeit des Systems auftreten. Typischerweise möchte man damit Speicherlecks auffinden.

4 Test-Driven Development

Test-Driven Development wurde von Kent Beck im Rahmen der agilen Vorgehensweise des Extreme Programming entwickelt. Dennoch ist die testgetriebene Entwicklung nicht an Extreme Programming gebunden, sonder kann auch mit

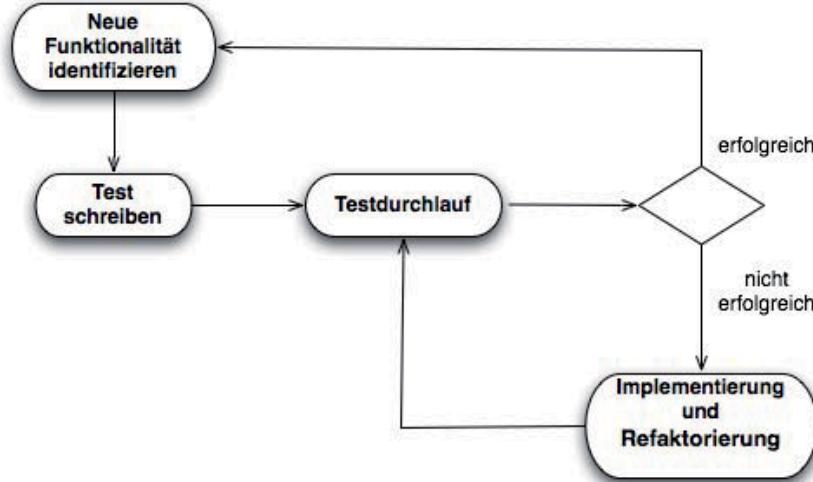


Abbildung 2. Quelle: übersetzt und adaptiert aus [9]

konventionellen Vorgehensmodellen kombiniert werden (vgl. [6, S. 181]). Das Vorgehen dabei ist Folgendes (vgl. [9]):

Der Entwickler identifiziert zuerst eine neue Funktionalität die es zu implementieren gilt. Daraufhin schreibt er einen Test für diese Funktionalität. Lässt diesen durchlaufen und programmiert erst dann wenn dieser Test fehlschlägt. Sobald der Test erfolgreich durchläuft und man Codeduplikate mittels Refakturierung behoben hat, fängt man wieder mit der Funktionalitätsidentifikation an.

Vorteile dieses Vorgehens sind das jede Codestelle getestet ist. Dadurch wiederum werden Regressionstest und das Debugging erleichtert. Außerdem stellt der Testcode eine Art von Dokumentation des Codes dar.

Dem gegenüber steht, dass diese Vorgehensweise nur für neue Projekte möglich ist und nicht für bestehende Systeme einsetzbar ist. Und das sie voraussetzt dass eine automatische Testumgebung wie zum Beispiel JsUnit vorhanden ist.

5 Fazit

Abschließend lässt sich sagen, dass bei Web Anwendungen der Fokus deutlich höher auf den nicht-funktionalen Anforderungen liegt als dies bei konventionellen Anwendungen der Fall ist. Am deutlichsten wird das bei der Usability. So sehen Anwendungen die für den Einsatz im Unternehmen gestalten wurden, die auch niemals ein Kunde eines Unternehmens sehen wird, häufig funktional und lieblos gestaltet aus. Hingegen sind Webanwendungen meist deutlich aufwendiger gestaltet. Dies ist aber auch notwendig, da der Benutzer bei Nichtgefallen im Web deutlich schneller eine Applikation wechselt. Dies macht es insbesondere für werbefinanzierte Angebote wichtig ständig Usability Tests durchzuführen und

die Ergebnisse in die Anwendung einfließen zu lassen. Insbesondere A/B-Tests mit Webanalysewerkzeugen gehen einfach und liefern darüber hinaus auch noch interessante Daten über die Nutzer.

Bezüglich der Barrierefreiheit gilt es insbesondere für kommerzielle abzuwegen, ob es sich lohnt diese zu prüfen und umzusetzen. Das heißt ob der damit verbundene Imagegewinn sowie die Zielgruppenerweiterung die Ausgaben wieder herausholen. Staatliche Webseiten hingegen sind gemäß der Barrierefreiheit Informationstechnik-Verordnung gesetzlich verpflichtet ihre Angebote barrierefrei zu gestalten. Für den privaten Entwickler wird es vermutlich eher darauf ankommen ob er behinderte Personen in seinem Umfeld hat und denen auch Zugang ermöglichen will. Natürlich wäre es für alle Angebote wünschenswert wenn sie barrierefrei gestaltet wären, um auch Behinderten die volle digitale Teilhabe zu gewährleisten.

Die Belastbarkeitstests sind insbesondere für Angebote wichtig die sensible Daten verarbeiten, wie z.B. Finanztransaktionen. Hier könnte man sonst in Missgunst des Nutzers fallen, wenn beispielsweise eine Überweisung wegen Serverüberlastung nicht durchgeführt wurde, aber er keine Erfolgsmeldung bekommen hat.

Automatisierte Tests sind besonders dann wichtige wenn man eine komplette Webanwendung von Anfang an programmiert. Für Webagenturen die hauptsächlich bestehende Software an Kundenwünsche anpassen, wie z.B. Maglist, reichen manuellen Test aus.

Literatur

- [1] BIK-Projekt. *Informationen und Preise*. URL: http://www.bitvtest.de/bitvtest/bitv_test_beauftragen/infos_preise.html (besucht am 18.06.2011).
- [2] David Brandon. *Software engineering for modern Web applications*. Hershey: Information Science Reference, 2008.
- [3] Bartosz Jan Domburg. *Google Analytics und der Datenschutzhinweis*. 29. Mai 2008. (Besucht am 18.06.2011).
- [4] Jan Eric Hellbusch. *Barrierefreiheit verstehen und umsetzen*. dpunkt. Verlag, 2011.
- [5] Britta Hofmann. *Einführung in die ISO 9241-110*. URL: <http://www.fit-fuer-usability.de/archiv/einfuehrung-in-die-iso-9241-110/> (besucht am 18.06.2011).
- [6] Gerti Kappel, Birgit Pröll und Siegfried Reich. *Web Engineering*. Hrsg. von Werner Retschitzegger. 1. Aufl. dpunkt. Verlag, 2004.
- [7] Jakob Nielsen. *Fancy Formatting, Fancy Words = Looks Like a Promotion = Ignored*. 4. Sep. 2007. URL: <http://www.useit.com/alertbox/fancy-formatting.html> (besucht am 18.06.2011).
- [8] Frank Puscher. *Leitfaden Web-Usability*. 1. Aufl. dpunkt. Verlag, 2009.
- [9] Ian Sommerville. *Software Engineering*. 9. Aufl. Pearson, 2011.

- [10] W3C. *Introduction to Web Accessibility*. URL: <http://www.w3.org/WAI/intro/accessibility.php> (besucht am 18.06.2011).
- [11] Henrike Werhahn. *Evaluationsmethoden*. URL: <http://www.fit-fuer-usability.de/archiv/evaluationsmethoden/> (besucht am 18.06.2011).
- [12] Stefan Wienströer. *Usability Tests mit Google Analytics*. URL: <http://blog.stevieswebsite.de/2010/01/usability-tests-mit-google-analytics/> (besucht am 18.06.2011).
- [13] Wolfgang Wiese. *Checkliste zur Gestaltung barrierefreier Webanwendungen und Webaufritte*. URL: <http://www.vorlagen.uni-erlangen.de/regeln/checkliste.shtml> (besucht am 18.06.2011).

MMT Seminar 2011

TYPO3 - CMS

Anton Schreiber

1 Einleitung

Diese Ausarbeitung beschäftigt sich im Rahmen des Multimedia-Technik Seminars des Lehrstuhls Medieninformatik der Universität Bamberg mit dem Open-Source Content-Management System Typo3. Ziel dieser Arbeit ist es dem Leser einen grober Überblick über die Entstehung, über den Aufbau und die Struktur und über die Funktionen des Systems zu geben.

Das System Typo3 wird von der Fachliteratur der Klasse der „Enterprise-Content-Management-Systeme“ zugeordnet, was in der Mächtigkeit jedoch auch in der langen Einarbeitungszeit für Entwickler begründet liegt.

2 Was ist Typo3

Typo3 zählt zu den fünf am weitesten verbreitetsten Open-Source CMS Systemen weltweit, welches auf eine sehr große Liste von Features zurückgreifen kann. Typo3 zählt dabei genau wie viele der großen Systemen zu den Web-Content-Management-Systemen (WCMS). Es wird geschätzt, dass Typo3 weltweit über eine halbe Million (vgl. [wiki2011]) mal auf verschiedenen Plattformen und für verschiedenste Zwecke eingesetzt wird.

So setzen verschiedene große Unternehmen auf Typo3 als Plattform (www.tvmovie.de), für ihr Intranet (Porsche AG) oder für reguläre große Webseiten (www.uni-bamberg.de).

3 Geschichte von Typo3

Der Kopenhagener Programmierer Kasper Skårhøj entwickelte das System, anfangs noch unter einem anderen Namen, im Jahre 1997 als selbstständiger Webdesigner für seine Kunden. Das System, damals noch nicht Open-Source, hatte zu diesem Zeitpunkt noch sehr wenig Konkurrenz am Markt. Daher hatte Skårhøj schnell einen großen Erfolg bei seinen Kunden. Nach ca. einem Jahr wurde eine Web-Agentur auf ihn und sein System aufmerksam und bot ihm an, er könne sein System in dem Unternehmen und für das Unternehmen einsetzen und weiterentwickeln. Somit wurde im Jahre 1998 Skårhøj in der Agentur angestellt und war nun nur noch mit der Weiterentwicklung des jungen Systems beauftragt.

Bereits nach einem Jahr stellt Skårhøj jedoch fest, dass die Agentur Grundsätze verfolgte die sich nicht mit den seinen deckten. Das betraf in erster Linie die Art der Entwicklung, da diese nun von einem finanziellen Gedanken getrieben wurden. Aus diesem Grund entschloss sich Skårhøj bereits im folgenden Jahr die Web-Agentur wieder zu verlassen und das Projekt alleine weiter fort zu führen.

Im Jahre 1999 war er erneut als Freiberufler tätig und arbeitete alleine an dem System weiter, was er drei Jahre fortführen konnte. In dieser Zeit wurde auch der eigentliche Name des Systems erschaffen, inspiriert von einem Schreibfehler mit dem Skårhøj versehentlich die Arbeit von einer Woche löschte. In der Version 3 wurde

dann der Name des Systems festgelegt und seit dem behalten obwohl sich das System inzwischen bei der Version 4 befindet. (vgl. [typo32011a])

Heute wird das System welches unter der GPL-Lizenz veröffentlicht ist von einem Team von Kern-Entwicklern betreut und einer großen Community die die Entwickler unterstützen. Typo3 existiert zur Zeit in der Version 4.5 (stable). Für diese Version wurde darüber hinaus ein Long-Term-Support zugesichert. Das heißt für die nächsten Jahre müssen die Typo3 Systeme mit dieser Version nicht aktualisiert werden.

4 Aufbau und Struktur

Das System wurde bereits 1997 in der Programmiersprache PHP geschrieben und setzte auf die Anbindung einer SQL Datenbank, wobei inzwischen verschiedenste Datenbankstrukturen mit Typo3 genutzt werden können. Typo3 verzichtete im Gegensatz zu vielen anderen CMS vollständig auf die Nutzung eines Frameworks, was hauptsächlich im Alter des Systems begründet liegt. Inzwischen werden teilweise Elemente aus verschiedenen Frameworks genutzt, wobei noch immer drauf geachtet wird dieses so gut es geht zu vermeiden.

Die große Stärke des Systems lag von Anfang an in der Flexibilität des Systems, was mit der Version 3.5.1 durch die Einführung des Extension Manager (EM) noch gesteigert werden konnte. Über den EM lassen sich diverse Plugins, Add-Ons und andere Erweiterung installieren, die das System um weitere Funktionen wie ein Shop- oder ein Blogsystem zu integrieren. Doch auch viele kleine Features wie Anbindungen an Twitter oder Facebook können auf diese Weise ohne im Core des Systems arbeiten zu müssen schnell eingebunden werden.

Typo3 trennt die Rollen des Besuchers der Webseite und der eines Administrators durch ein separates Backend und Frontend. Das Frontend ist dabei die eigentliche Oberfläche die ein normaler Besucher zu sehen bekommt, das Backend ist die Oberfläche für Administratoren, Redakteure oder Entwickler. Die Trennung von verschiedenen Aufgaben und Bereichen zieht sich auch für die Verwaltung von Inhalt und das Bearbeiten des Designs durch. So wird der Inhalt unabhängig vom Design in Typo3 angelegt, anschließend wird ihm ein Design zugeordnet, welches dann die Ausgabe steuert.

Für das erstellen eines Designs wird die Konfigurationssprache Typo-Skript benötigt, auf die später noch eingegangen wird.

5 System Anforderungen und Installation

Da Typo3 Plattform-unabhängig ist, hat es als System insgesamt nur relativ wenig Anforderungen, welche jedoch zwingend erfüllt sein müssen.

Die Grundlage von Typo3 in der Version 4.5 bilden PHP und MySQL, so muss der Server mindestens PHP 5.2.0 fähig sein, Apache in der Version 2.0 oder höher installiert haben und eine MySQL Datenbank ab Version 3.23. Bei der Datenbank

kann wie bereits vorher erwähnt auch auf eine andere DB zurückgegriffen werden, wobei MySQL selber von den Typo3 Entwicklern empfohlen wird.

Zur Installation auf einem Webserver werden neben den oben genannten Anforderungen noch die Installationspaketes benötigt, welche man im Bereich Downloads unter www.typo3.org in der aktuellen Version herunterladen kann. Je nachdem ob ein eigener Server betrieben wird oder nur ein gehosteter Webserver vorhanden ist sollte das entsprechende Packet heruntergeladen werden und auf dem Server entpackt werden.

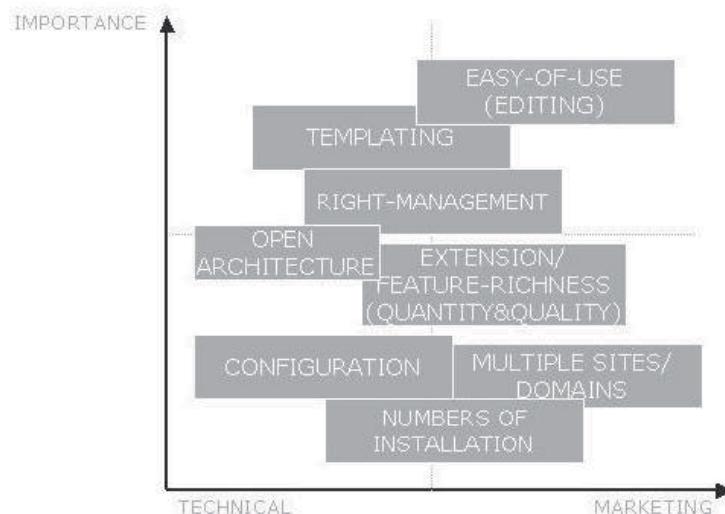
Um die Installation nun starten zu können, muss aus Sicherheitsgründen zunächst die Installation aktiviert werden.

Hierfür muss eine leere Textdatei namens „ENABLE_INSTALL_TOOL“ erstellt und in das Root Verzeichnis der Installation geladen werden. Ruft man nun die URL des Speicherplatzes auf, startet automatisch die Installationshilfe, die den Benutzer in vier Schritten durch die Installation leitet. Vorausgesetzt man installiert die Testdaten mit ist nach anschließend eine lauffähige Typo3 Installation vorhanden, welche bereits angeschaut werden kann.



6 Überblick über Typo3

Typo3 bietet als vollwertiges CMS alle Funktionen die man zum Betreiben einer Webseite benötigt. Die Entwickler, allen voran Kasper Skårhøj, legen dabei auf verschiedene Bereiche unterschiedlichen Wert. Die folgende Grafik bildet die unterschiedlichen Aspekte die bei Typo3 verfolgt werden ab.



Die Grafik zeigt deutlich, dass der Punkt des Inhalte Erzeugen und Editieren (Editing) am wichtigsten ist. Aus diesem Grund wurde auch das Backend in einer der letzten Versionen stark überarbeitet um die Bedienbarkeit zu vereinfachen.

Der Bereich Templating tendiert stark zum technischen Bereich, was später im Kapitel 7 deutlich wird. Die anderen Aspekte wurden bereits angesprochen oder sind an dieser Stelle nicht von Interesse.

Bereits an dritter Stelle findet sich das Rechtemanagement. Der Fokus auf diesem Bereich spricht für den Enterprise-CMS Gedanken, da vor allem bei globalen Unternehmen die ein Intranet erzeugen wollen unterschiedliche Rollen mit genau definierten Zugriffsrechten überaus wichtig sind. So können auch sensible Daten wie Quartalszahlen oder ähnliches veröffentlicht werden und sie sind trotzdem vor unberechtigtem Zugriff geschützt.

6.1 Das Frontend

Das sogenannte Frontend bezeichnet das, was der Besucher der Seite letztendlich zu sehen bekommt. Dabei ist es wichtig zu verstehen, wie sich das Frontend eigentlich zusammensetzt bzw. wie es funktioniert.

Das Frontend wird mittels eines gelieferten Design-Templates implementiert. Durch Typo-Skript wird dabei genau definiert an welcher Stelle welcher Inhalt steht. Wie bei den meisten Webseiten teilt sich auch eine mit Typo3 erzeugte Website standardmäßig in die folgenden Bereiche auf:

- Header-Bereich
- Hauptmenü
- Content – teilt sich standardmäßig in drei Bereiche auf

- Footer

Die Bereiche sind in Typo3 vordefiniert und können so später beim Templatizedesign sehr leicht mit Inhalten belegt werden. Weitere Bereiche sind natürlich erzeugbar, doch auch die Regeln der Gestaltung einer Webseite raten dazu diese Bereiche in einer Website beizubehalten.

Das mächtige Rechtemanagement von Typo3 bietet auch für das Frontend einige Möglichkeiten. So können sowohl für das Backend aber auch für das Frontend verschiedene Rolle definiert werden, welche Zugriff auf unterschiedliche Inhalte haben. Beispielsweise würde eine normaler Benutzer dann die Rolle „Gast“ ein angemeldeter Besucher bspw. „Benutzer“. Auf diese Weise lassen sich besonders im Bereich des Intranets große Portal erzeugen.

Eine weiter Funktion die über das Frontend genutzt werden kann sind sog. Benutzerinteraktionen. Hier ermöglicht man dem Benutzer bspw. über Formulare die Möglichkeit Inhalte der Seite zu beeinflussen. Ein Beispiel für eine sehr häufig genutzt Interaktion sind Umfragen. Hier wird dem Besucher der Webseite die Möglichkeit geboten, eine Frage anhand von zumeist vordefinierten Antworten zu geben. Die Antwort wird dann genau wie die normalen Inhalte auf der Datenbank abgelegt und kann – wenn gewünscht – für andere Besucher öffentlich gemacht werden.

Das Backend ist das eigentliche Herzstück von Typo3. Hier sind die verschiedenen Aufgabenträger die für eine Installation notwendig sind zu finden. Die drei wichtigsten sind die Redakteure, Webdesigner, Entwickler und Programmierer (Fachleute aus anderen Bereich wie Marketing/PR sind an dieser Stelle bewusst nicht betrachtet, da es um die technische Seite des Systems geht).

Die verschiedenen Gruppen haben durch das Rechtemanagement geregelte Zugriffsrechte auf unterschiedliche Bereiche der Typo3 Installation. So können Redakteure bspw. nur Inhalte erzeugen und bearbeiten aber nicht das Template Design verändern.

Das Backend ist relativ einfach aufgebaut und ist – falls nicht anders eingestellt – über www.die-domain.de/typo3 zu erreichen. Auf der linken Seite finden sich in Bereiche getrennte Icons, mit denen alle nötigen Aufgaben erledigt werden können. Je nach zugeteilten Rechten, sind bestimmte Bereiche sichtbar bestimmte hingegen nicht oder nur mit eingeschränkten Rechten.

In der Grafik ist das Standardmenü zu sehen, welches Sie als Administrator zu sehen bekommen, wobei die System Sprache noch auf Englisch eingestellt ist. Die nun folgenden Bereiche, sind die für eine normale Typo3 Website wichtig sind. Die anderen werden aus Zeitgründen nicht betrachtet.



6.1.1 Page – Bereich

Der Page Bereich wird genutzt um jegliche Inhalte von Inhaltsseiten zu bearbeiten. Öffnet man diesen Bereich, sieht man in Form eines Seitenbaums aller bislang erstellten Seiten. Die Seiten können dabei bereits veröffentlicht sein oder noch im Entwurf Status sein.

Möchte man eine der Seiten bearbeiten, klickt man auf diese und es erscheint die Übersicht. Eine Seite ist bei Typo3 Standardmäßig in die Bereiche Left, Normal und Right unterteilt. In jede dieser drei Inhaltsbereiche werden in Form einzelner Inhaltselemente tatsächliche Inhalte erzeugt und abgelegt. Je nachdem wo die Bereiche im Template positioniert wurden, erscheint dann der Inhalt im Frontend an der gewünschten Stelle.

6.1.2 Template – Bereich

In diesem Bereich werden die unterschiedlichen Templates angelegt. Normalerweise reicht es aus ein einziges Template anzulegen, welches dann für alle Seiten genutzt wird. Manchmal ist es aber auch nötig für die verschiedenen Unterbereiche unterschiedliche Templates anzulegen. Eine zweite Möglichkeit wäre, wenn mit einer Typo3 Installation mehrere Webseiten verwaltet werden sollen, so könnte man für jeden Bereich ein eigenes Template erzeugen.

Ein Template setzt sich dabei aus verschiedenen Ressourcen, Konstanten und dem Setup zusammen. In den Ressourcen werden beispielsweise Bilder wie Logos abgelegt, die für die gesamten Website gebraucht werden. In den Konstanten werden feste Variablen definiert, so könnte hier zum Beispiel ein Slogan der auf jeder Seite stehen soll festgehalten werden. In dem letzten Teil, dem Setup, findet sich dann der wichtigste Teil. Hier wird mittels der Konfigurationssprache Typo-Skript das eigentliche Template erzeugt, Bereiche zugeordnet oder auch Menüs definiert.

6.1.3 News Admin – Bereich

Der News Bereich ist aus dem ursprünglich separaten Modul tt-news entstanden. Hier werden spezielle News-Artikel eingepflegt, die dann auf der Seite in einer vorgegeben Struktur erscheinen. So muss man sich dank dieses Features nicht mehr darum kümmern, dass eine Liste der aktuellen News mit einer Zusammenfassung ausgegeben wird. Die wichtigsten Funktionen die ein ausgereiftes Newssystem benötigt, sind alle vorhanden. Zusätzlich ist das News-System sehr individuell anpassbar und kann so für verschiedenste Anforderungen genutzt werden.

Die News-Artikel werden analog zu den normalen Seiten angelegt und bearbeitet. Die Artikel können dabei nach Kategorien geordnet werden um sie auf unterschiedlichen Seiten anzeigen zu lassen. Auf diese Weise muss man die Artikel nur kategorisieren und braucht sich um kaum weiteren Einstellungen zu kümmern. So können bspw. auch Templates mit einer bestimmten Kategorie versehen werden um die Artikel entsprechend anzeigen zu lassen.

Theoretisch ist ein Newsbereich auch mit der normalen Seiten umsetzbar, würde aber einen großen Aufwand bedeuten.

6.1.4 Filelist

Dieser Bereich wird meistens fileadmin genannt. Hier können alle Dateien die für das System wichtig sind manuell hoch geladen werden. Für das Anlegen eines Template, können so zum Beispiel die benötigten HTML Vorlagen oder CSS-Dateien in einen vorgegebenen Ordner hoch geladen werden. Andere Dateien wie zu verlinkende PDF Dokumente werden ebenfalls an dieser Stelle hoch geladen.

An dieser Stelle sollte noch erwähnt werden, dass ein Redakteur auch ohne den Zugriff auf den Filemanager auskommen kann. Das hoch laden von Dokumenten kann auch innerhalb der Inhaltsseiten direkt erfolgen, sofern dieses eingestellt ist.

Usernames:	
admin	
advanced_editor	
anton	
news_editor	
simple_editor	

6.1.5 User Admin

Hier werden die unterschiedlichen User des Systems angelegt und verwaltet. Den jeweiligen Usern werden an dieser Stelle den entsprechenden Gruppen zugeordnet um ihnen benötigte Zugriffsrechte zu verschaffen.

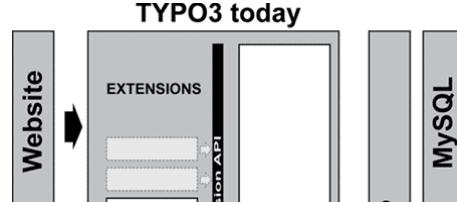
Administratoren sind dabei zur schnelleren Übersicht mit einem blauen Männchen normale User mit einem grünen markiert.

6.1.6 Extension Manager

Der Extension Manager der bereits mehrfach erwähnt wurde verwaltet alle Extensions der Typo3 Installation. Hier können zum einen direkt neue Extensions installiert werden, installierte aktualisiert werden oder deinstalliert bzw. deaktiviert werden.

Der Extension Manager wurde in der Version 3.4.1 eingeführt und gab dem System eine extreme Flexibilität. So entwickeln Unternehmen die unterschiedlichste Extensions und stellen diese kostenlos im Extension Repository zur Verfügung (<http://typo3.org/extensions/>). Die Einführung des Managers führt bei dem immer größer werdenden System zusätzlich zu sehr viel mehr Überblick und klar getrennten Bereichen.

So ist der Core der von Typo3 nicht von der Installation einer Extension betroffen. Das ist besonders wichtig für eine die Update-Strategie. Auf diese Weise kann sichergestellt werden, dass ein Update ein bestehendes System nicht

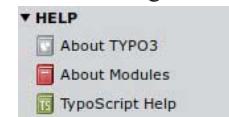


beschädigt. Diese Aussage ist jedoch nur solange korrekt, solange keine grundlegenden Dinge im Core geändert wurden, da die Extensions häufig auf verschiedene Funktionen des Cores zurückgreifen. Fehlen diese ist die Extension ebenfalls nicht mehr lauffähig. Vor jedem größeren Update sollte daher genau überprüft werden, ob die installierten Extensions mit dem Update kompatibel sind.

Alle im Repository veröffentlichten Extensions unterliegen genau wie Typo3 der GNU Public Lizenz. Auf diese Weise können Entwickler bestimmte Extensions auch selber modifizieren, falls Sie bestimmte zusätzliche Funktionen benötigen.

6.1.7 Weitere Menüpunkte

Die anderen Menüpunkte des Backends sind in den meisten Fällen weniger von Bedeutung und werden daher daher hier auch nicht weiter betrachtet. Um zu erfahren was hinter den anderen Punkten steht, empfiehlt es sich die Typo3 Dokumentation anzuschauen die unten im Menü verlinkt ist.

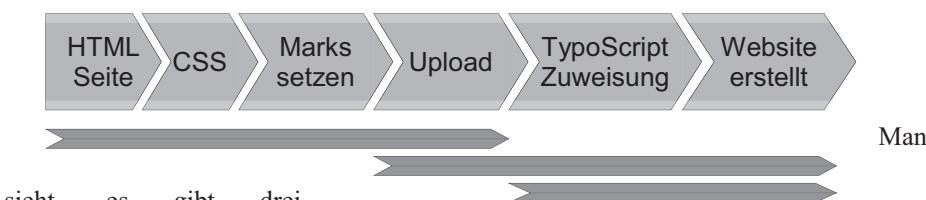


7 Template Design

Der wohl schwierigste Teil und trotzdem sehr wichtige Teil ist die Gestaltung des Frontends, das sogenannte Template Design. Im Gegensatz zur Einarbeitungszeit von ca. einer Stunde für einen Redakteur benötigt ein Typo3 Entwickler ca. ein halbes Jahr um sicher ein Template anlegen zu können.

An dieser Stelle muss jedoch zunächst noch erwähnt werden, dass durch die starke und sehr aktive Community es verschiedene kostenlose Design Vorlagen gibt die besonders am Anfang genutzt werden können. Dieses vereinfacht besonders den Einstieg, da man sich häufig an dem vorgefertigten Design orientieren kann.

Generell gilt bei Typo3, dass es sinnvoll ist die einzelnen Aufgabengebiete sehr strikt zu trennen. Dieses unterstützt auch wieder den Gedanken, dass Typo3 ein WCMS für größere Unternehmen ist die zumeist auf mehr Kapazitäten zurückgreifen können als kleine Unternehmen. So sollte der Workflow wie in der folgenden Grafik gezeigt getrennt werden.



sieht, es gibt drei unterschiedliche Gruppe. Die erste Gruppe ist die der Webdesigner. Diese muss nicht zwingend Fachwissen über Typo3 haben, sondern sind sehr gut im programmieren statischer HTML Seiten, welche mit CSS gestaltet werden. So steht an jedem Anfang eines Templates eine statische HTML Seite, welche mit Dummy Text gefüllt wird.

Diese Seite dient im späteren dann als Grundlage des Designs. Als nächstes sollte gemäß den aktuellen Gestaltungsregeln für eine Webseite der Inhalt so gut es geht mit CSS formatiert werden. Dieses ist besonders unter dem Aspekt wichtig, dass später nicht jedes mal der gesamte Aufbau der Seite verändert werden muss, sondern lediglich das Stylesheet angepasst werden muss.

Der dritte Teil ist der erste, der erkennbar macht, dass es um ein Typo3 Template geht. Hier werden nämlich nun an den Stellen wo später der Text, das Menü oder der Footerinhalt generiert werden soll sogenannte Marks gesetzt. Anschließend werden die vorbereiteten HTML-Templates mittels des Filemanagers hoch geladen.

Als nächstes erfolgt der schwerste Teil, der von einem speziellen Typo3 Entwickler oder Programmierer umgesetzt wird. Hier wird nun mittels Typo-Skript Menüs erzeugt, Inhaltsbereiche den Marks zugeordnet und weitere wichtige Einstellungen getätigt. Sobald dieser Bereich nun abgeschlossen ist, ist die Website fertig und kann von den Besuchern angeschaut werden.

7.1 Marks und wie man sie erzeugt

Die bereits angesprochenen sogenannten Marks werden in dem statischen HTML-Template eingebunden, um diese Bereiche anschließend mit Typo-Skript direkt ansprechen zu können. Es gibt zwei Arten von Marks.

Die erste Art ist die einfache Art, die später einfach nur ersetzt wird. So könnte z.B. für das Menü der Marker wie folgt angelegt werden.

```
< ! - - ### MENU ### - - >
```

Die zweite Art definiert einen Bereich, mit Anfang und Ende. Innerhalb dieses Bereichs können dann weitere Bereiche angelegt werden.

```
< ! - - ### HEADER ### begin - - >
      Weitere Inhalte
< ! - - ### HEADER ### end - - >
```

Nachdem die Marks an den benötigten Stellen angelegt worden sind, wird nun mit dem verknüpfen fortgefahrene.

7.2 Inhalte mit Typo-Skript erzeugen

Dieser Teil wird in dieser Ausarbeitung bewusst kurz gehalten, da die vollständige Ausarbeitung bei weitem den Rahmen sprengen würde. Wichtig ist, dass der grundlegende Gedanke nachvollzogen werden kann, wie Inhalte mittels Typo-Skript zugeordnet wird.

Zum Arbeiten mit Typo-Skript ist wichtig zu verstehen, dass es keine echte Skript oder Programmiersprache sondern lediglich eine Konfigurationssprache. So werden Inhaltselemente immer auf die gleiche festgelegte Art und Weise angesprochen.

Grundlage für das Arbeiten in Typo-Skript ist der TypoSkript-Object-Browser. Hier findet man eine festgelegte Struktur, die man nun gezielt mit den eigenen Elementen überschreibt, bzw. füllt. Zum besseren Verständniss kann man sich den die

Inhalte die der Object-Browser darstellt wie einen vordefinierten DOM-Tree vorstellen. So gibt es Bereiche für den linken Inhaltbereiche, für den Header usw. Mittels der Suchfunktion des Object-Browsers können diese gezielt gesucht und im Setupteil des Templates überschreiben werden.

Der wichtigste Teil zum Bearbeiten eines Templates ist das PAGE-Object, genauer gesagt der Bereich page.10. Zunächst möchten wir nun die zuvor erstellte Vorlage mit den Marks als Template einbinden. Dafür gehen wir wie folgt vor.

```
page.10 = TEMPLATE
page.10 {
    template = FILE
    template.file = fileadmin/ordner/die-vorlage.html
}
```

Nun haben wir das Template eingebunden und können als nächstes damit beginnen, die vorgegebenen Marks mit Inhalt zu befüllen. Um nun beispielsweise das Menü im Mark „MENUE“ zu befüllen gehen wir wie folgt vor.

```
page.10.20 {
    10 = HMENU
    10.1 = TMENU
    10.1.NO.linkWrap = <b> | </b><br>
}
```

Wichtig ist zu verstehen, was das page.10.20 bedeutet. Der Bereich page.10 ist wie oben kurz erwähnt der eigentliche Inhaltsteil, so gehören fast alle Bereiche des Templates die geändert werden diesem Teil an. Als nächstes muss das passende Element mit Hilfe des Object-Browsers herausgesucht werden. So ist das Hauptmenü grundsätzlich das Element page.10.20, welches wir hier nun mit Inhalt überschreiben.

Analog zu diesem Vorgehen werden nun die anderen Bereiche gefüllt.

8 Fazit

TYPO3 ist ein sehr modernes, sehr flexibles und vor allem sehr mächtiges Web-Content-Management-System. Durch die große aktive Community und die Open-Source Lizenz wird das System auch in der Zukunft mit größter Wahrscheinlichkeit weiter entwickelt werden. Man kann daher als Unternehmen gut auf TYPO3 setzen ohne Gefahr laufen zu müssen bald ohne Support darstehen zu müssen.

Alles in allem ist TYPO3 aber definitiv ein Enterprise CMS und empfiehlt sich daher in erster Regel für große Unternehmen. Kleinere Unternehmen sollten, falls diese doch aus bestimmten Gründen auf TYPO3 setzen wollen, auf eine gute Agentur zurückgreifen die sich auf TYPO3 spezialisiert hat.

Für alle die sich selber in TYPO3 einarbeiten wollen empfehle ich das sehr gute und ausführliche Buch „Einstieg in TYPO3 4.5“ von Galileo Design sowie die Tutorials auf www.typo3.org.

Quellenangabe:

- | | |
|--------------|--|
| [wiki2011] | http://de.wikipedia.org/wiki/TYPO3 (Stand 15.07.2011) |
| [typo32011a] | http://typo3.com/History.1268.0.html (Stand 22.07.2011) |
| [typo32011b] | http://typo3.com (Stand 08.07.2011) |
| [BoSt2011] | Einstieg in Typo3, Gallileo Press Bonn 2011 |
| [MeHöRi10] | Das Typo3 Profihandbuch, Addison-Wesley München Mai 2010 |
| [MoRoWö11] | Typo3 4.5. Das Enterprise Content Managemen System, Open Source Press 2011 |

CMS Joomla! (PHP)

Johannes Paul Kupser

Otto-Friedrich Universität Bamberg,
Feldkirchenstr. 21, 96045 Bamberg

1 Einführung

Joomla! ist ein PHP basiertes OpenSource Content-Management-System (CMS), das seinen Ursprung in dem zuerst kommerziellen CMS der australischen Firma *Miro International* hat. Das CMS wurde im Jahr 2002 auch in einer OpenSource Variante veröffentlicht und im Jahr 2005 hat sich die Entwicklergemeinde der offenen Variante des CMS dazu entschlossen, ein neues Projekt auf Basis des damals noch Mambo genannten CMS zu gründen, da die Entwickler befürchteten, dass der OpenSource-Gedanke gefährdet sei. Binnen kurzer Zeit sind die Entwickler und die schon vorhandene Community nach einem offenen Brief des Chefentwicklers Andrew Eddi umgezogen und haben das Projekt *Joomla!* ins Leben gerufen.¹

1.1 Versionshistorie

Joomla! liegt aktuell in den Versionen 1.5 und 1.7 vor. Die Version 1.0 basierte noch vollständig auf der Codebasis des abgespaltenen Mambo CMS. Ende Januar 2008 wurde die erste stabile Version 1.5 als Long-Term-Support (LTS) Version veröffentlicht. Mit dieser Version hat man sich komplett von der Mambobasis getrennt und als weiteren Schritt in die Zukunft entstand das Joomla-Framework. Die wichtigste Erneuerung war jedoch der Einzug des Model-View-Controller-Patterns (MVC) im Bereich der Komponentenentwicklung.² Im Januar 2012 soll die Version 1.8 als nächste LTS-Version veröffentlicht werden.

1.2 Lizenz

OpenSource Matters, Inc. (OSM) ist ein non-profit Verein mit Sitz in New York (U.S.A), der das Projekt Joomla! in organisatorischer, rechtlicher und finanzieller Sicht unterstützt. Zu weiteren Aufgaben des Vereins zählt das langfristige Bestehen des auch in Zukunft freien Joomla! Projekts, das unter der OpenSource Lizenz GNU GPL v2 steht.³

Der Name Joomla! ist urheberrechtlich geschützt, kommt aus dem Suahelischen

¹ The Official Joomla! {book}, S. 1ff

² Mastering Joomla! 1.5 Extension and Framework Development, S. 23

³ <http://opensourcematters.org>

und bedeutet „als Ganzes“[2]. Das Ausrufezeichen ist fester Bestandteil des Namens, da joomla ein Ausruf ist: „zusammen für eine Sache“. Das offizielle Logo (siehe Abbildung 1) setzt sich aus vier ineinander verschlungenen „J's“ zusammen und soll das Community Gefühl des gesamten Projekts unterstreichen.



Abbildung 1. Offizielles Joomla! Logo mit Slogan

1.3 Einsatzbereich

Joomla! hat u.a. 2005 den “Best Linux/Open Source Project” Preis⁴ erhalten und ist ein sehr beliebtes OpenSource CMS. Laut Statistiken nutzen 2,7 % aller Websites Joomla!, was einem CMS-Marktanteil von 2,7 % entspricht.⁵ Die Einsatzbereiche von Joomla! sind sehr vielseitig, so wird das CMS für private Internetauftritte genutzt, als auch von international operierenden Firmen wie zum Beispiel die Website von Porsche-Brasilien.⁶ Zu dem schon mit reichlich Funktionen ausgestattetem Kernsystem gibt es mittlerweile mehr als 8.000⁷ Erweiterungen, die das CMS für fast jeden Bereich einsetzbar machen. Einsatzbeispiele sind Schul- oder Universitätsseiten, Vereine, Magazine oder Firmenauftritte.

2 Architektur

Die Architektur des Joomla! CMS ist in drei Schichten⁸ aufgeteilt (siehe Abbildung 2). Eine Basisschicht, die Framework-Schicht auf der die Application-Schicht mit den Anwendungsfunktionen des Joomla! CMS aufbaut, sowie eine Extension-Schicht, die die Anwendungsschicht durch Komponenten, Module und Templates erweitert. Mit Version 1.5 wurde das modular einsetzbare Joomla! Framework eingeführt, oft auch als Joomla! Core bezeichnet. Das Framework ist so konzipiert, dass theoretisch auch andere eigenständige Anwendungen auf das Framework aufsetzen können. Beispiele sind in der Joomla! Gemeinde jedoch nicht bekannt.

⁴ The Official Joomla! {book}, S. 3

⁵ http://w3techs.com/technologies/overview/content_management/all

⁶ <http://www.porsche.com.br>

⁷ <http://extensions.joomla.org/>

⁸ The Official Joomla! {book}, S. 29ff

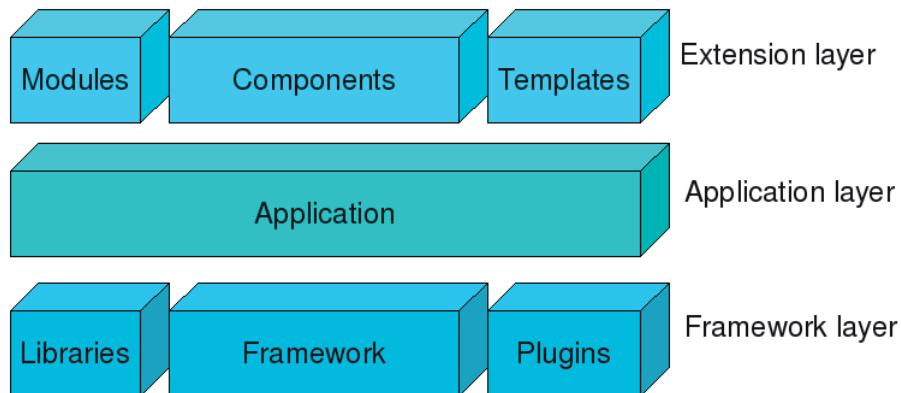


Abbildung 2. Joomla! Architekturaufbau ab Version 1.5

2.1 Framework-Schicht

Die Framework-Schicht bietet die grundlegende Funktionalität für die drei Schichten Architektur des Joomla! CMS und ist selbst in drei Bereiche aufgeteilt. Den größten Bereich stellt das Framework selbst dar, das sich vollständig unter dem Pfad 'libraries/joomla' befindet und so konzipiert ist, dass die Kernfunktionalität leicht wartbar und erweiterbar ist. Hier befinden sich alle Frameworkklassen wie zum Beispiel Klassen für die Datenbankkommunikation oder Caching. Das Library Paket beinhaltet weitere allgemeine Funktionalitäten, die sowohl vom Framework als auch von der Anwendungsschicht aufgerufen und genutzt werden können.

Plugins erweitern die Basisfunktionalität des Frameworks in der Weise, dass sie nach Auslösen bestimmter Aktionen ausgeführt werden. Die Plugins in Joomla! sind folglich nach dem Observer-Pattern konzipiert. Joomla! wird u.a. standardmäßig mit Authentifizierungs- und Beitragsplugins ausgeliefert, so wird zum Beispiel ein Event ausgelöst, wenn ein Benutzer versucht sich auf der Seite anzumelden. Weiterhin werden Plugins häufig genutzt, um spezielle Elemente wie Videos oder Bilder von externen Anbietern einzubinden. Dafür werden im Beitrag spezielle Tags mit geschweiften Klammern eingefügt, die anschließend den gewünschten Inhalt an der markierten Position einfügen.⁹

2.2 Application-Schicht

Letztlich befindet sich auf der Applikationsschicht das Joomla! CMS selbst und greift auf die darunter liegende Framework-Schicht zu. Innerhalb dieser Schicht gibt es in Joomla! vier Anwendungen:¹⁰

⁹ http://openbook.galileocomputing.de/joomla15/joomla_09_module_001.htm#mj22749e7c73684f8c4b6ada5bb4bf53bb

¹⁰ <http://docs.joomla.org/Framework/1.6>

1. JInstallation: Die Anwendung wird für die Installation des CMS auf einem Webserver benötigt und wird nach der Installation vom System entfernt.
2. JAdministrator: Diese Anwendung ist für den Administrationsbereich, also dem back-end des CMS zuständig.
3. JSite: Diese Anwendung steuert den für einen Besucher sichtbaren Bereich, also das front-end des CMS.
4. XML-RPC: Diese Schnittstelle ermöglicht Administratoren beispielsweise Zugang über einen Web-Service.

2.3 Erweiterungs-Schicht

Diese Schicht besteht aus drei Bereichen, die das Joomla! CMS auf der Anwendungsschicht erweitern. Hierzu dienen die Pakete Module, Komponenten und Templates.

Module. Module bieten eine flexible Möglichkeit, aufbereitete Daten oder Strukturen auf bestimmten Bereichen eines Templates im front-end darzustellen. Module können sowohl Daten aus der Datenbank abrufen und anzeigen, zum Beispiel als Menü-, Newsflash- oder Zufallsbild-Modul. Oder sie verarbeiten Daten in Form von Formularen, wie es das Login- oder Such-Modul machen. Hierbei können Module alleinstehend existieren oder mit Komponenten zusammen arbeiten.

Komponenten. Komponenten sind Mini-Anwendungen, die die Application-Schicht um notwendige Kernfunktionalität erweitern. Beispiele für Komponenten sind die zentrale Beitragsskomponente, Benachrichtigungen oder die Startseite. Komponenten sind strikt nach dem Model-View-Controller-Pattern (MVC) konzipiert und ermöglichen so eine Trennung zwischen Darstellung, Anwendungslogik und Datenverarbeitung. In der Joomla! Entwicklergemeinde ist die Kategorie Komponenten am besten mit ausführlichen Entwickler-Beispielen dokumentiert. Komponenten bieten eine sehr mächtige Erweiterungsmöglichkeit für Joomla!, da sie Funktionalitäten im front- und im back-end übernehmen können. Die im front-end angezeigten Inhalte bieten meistens eine Konfigurationsmöglichkeit im back-end. Gleichzeitig können Komponenten mit Plugins oder Modulen interagieren und Joomla! maßgeblich in seinem Funktionsumfang – zum Beispiel durch Wiki- oder Online-Shop Komponenten – erweitern.

Templates. Templates sind Vorlagen, die Joomla! einsetzt, um die grafische Repräsentation des front- und back-ends zu gestalten. Es gibt demnach Templates für das front-end, welche den Look & Feel der Seite ¹¹ für den Benutzer darstellen und back-end Templates, die es ermöglichen, den Administrationsbereich graphisch nach eigenen Ansprüchen umzugestalten, zum Beispiel angepasst an das Corporate Design einer Firma. Joomla! wird in der aktuellsten Version

¹¹ The Official Joomla! {book}, S. 83

mit insgesamt fünf einsetzbaren Templates ausgeliefert, drei front-end Templates und zwei back-end Templates, wobei eines der back-end Templates (*Hathor*) barrierefrei ist und ein front-end Template (*Beez5*) HTML5 Standard konform ist. Eigene Templates lassen sich relativ schnell und ohne weitere Kenntnisse außer HTML und CSS implementieren. Jedes Template benötigt einige Joomla-Template-Tags, die Joomla! nutzt, um zum Beispiel Fehler oder Warnmeldungen auszugeben und Modulpositionen zu bestimmen. Ein Joomla-Template-Tag, der ein Modul Namens "footer" einbindet, sieht folgendermaßen aus:

```
<jdoc:include type="modules" name="footer" />
```

Weiterhin muss in einer XML-Datei (*templateDetails.xml*) eines Templates u.a. angegeben werden, welche Einstellungsmöglichkeiten es bietet und wo welche Modulpositionen angeboten werden. Die Installation eines Templates erfolgt im back-end über den Extension-Manager, der eine Zip-Datei mit allen relevanten Dateien erwartet.

2.4 Voraussetzungen für eine Joomla! Installation.

Joomla! hat nur sehr geringe Systemvoraussetzungen und lässt sich bei Einhaltung der Voraussetzungen schnell und unkompliziert installieren. Joomla! benötigt einen Webserver, empfohlen wird Apache ab Version 2 aufwärts mit den Erweiterungen mod_mysql, mod_xml und mod_zlib. Eine optionale Erweiterung für suchmaschinenfreundlichere URLs ist mod_rewrite. Es ist auch möglich, Microsofts IIS ab Version 7 als Webserver einzusetzen. Momentan kann Joomla! nur ein Datenbanksystem ansprechen, hier setzt man auf MySQL ab Version 5.04. Da Joomla! PHP-basiert ist, wird PHP ab Version 5.2.4 oder höher für eine erfolgreiche Installation benötigt. Für die noch unterstützte LTS-Version Joomla! 1.5 gelten etwas geringere technische Voraussetzungen, da hier noch eine Abwärtskompatibilität zu Joomla! 1.0 möglich ist.¹²

3 Funktionen & Features

Seit der momentan aktuellen Version 1.7 hat sich vieles im Bereich Erleichterung, Verbesserung, Effizienz und Flexibilität in Joomla! geändert. Es gibt im Vergleich zu der Version 1.5 einige Neuerungen, die unterschiedlichste Rollen im Entwicklungs-, Wartungs- und Einsatzbereich des Joomla! CMS ansprechen. Joomla! bietet unzählig viele Funktionen wie eine Suchfunktion mit Volltextsuche, Internationalisierung von front- und back-end sowie innerhalb von Artikeln, Breadcrumbs, Bannerintegration, Benutzerregistrierung mit der Möglichkeit diese über einen Bestätigungslink zu aktivieren oder manuell per Administrator zu aktivieren, Authentifizierungsmechanismen über andere Webservices wie GMail,

¹² <http://www.joomla.org/technical-requirements.html>

OpenID oder über einen LDAP-Server¹³, Newsfeeds, ein internes Nachrichtensystem, zeitgesteuerte Artikel und viele weitere Funktionen.¹⁴ ¹⁵ Im Folgenden sind die bedeutendsten Funktionen für die wichtigsten Rollen beschrieben.

3.1 Webmaster & Administratoren

Administratoren und Webmaster können unterschiedliche Aufgaben übernehmen und doch überschneiden sich davon im Bereich eines CMS diverse. Ein Administrator ist für die Grundinstallation, Wartung und einen einwandfreien Betrieb zuständig, während der Webmaster sich etwas mehr um Inhalte, Menüs und Struktur einer Website kümmert.

Für Administratoren gibt es neben einer gewohnt einfachen Installationsroutine einen verbesserten und in das back-end integrierten Update-Mechanismus, der es ermöglicht, Joomla! und installierte Erweiterungen – im back-end unter dem Menüpunkt Extension Manager unter dem Reiter *Update* – unkompliziert zu aktualisieren. Durch diesen einfachen Prozess können ausgebesserte Softwarefehler und geschlossene Sicherheitslücken durch Updates installiert werden. Joomla! bietet neuerdings auch die Option Module, Templates, Plugins oder Sprachdateien, die via FTP-Server in die entsprechenden Verzeichnisse kopiert wurden, zu entdecken. Der Extension Manager bietet dann unter dem Reiter *Discover* an, einzelne oder alle erkannten Erweiterungen zu installieren.¹⁶ Joomla! setzt seit Version 1.6 auf eine flexibel konfigurierbare Access Control List (ACL), die es einem Administrator erlaubt, Benutzergruppen und Benutzer mit unterschiedlichen Berechtigungen und Zugriffsebenen auszustatten. Wobei sich eine Berechtigung dadurch auszeichnet, dass ein Benutzer oder eine Benutzergruppe Objekte wie Artikel, Module, Komponenten etc. anlegen und editieren darf¹⁷, während spezielle Zugriffsebenen die Sichtbarkeit von Inhalten auf der Homepage regeln und somit zum Beispiel nur registrierten Benutzern weitere Menüpunkte zur Verfügung stehen. Joomla! bietet vordefinierte Benutzergruppen an, lässt sich aber um beliebig viele weitere Benutzergruppen erweitern und Benutzer können mehreren Gruppen zugeordnet werden. Weiterhin können beliebige Zugriffsebenen angelegt werden und Benutzergruppen wiederum können diesen zugewiesen werden.¹⁸ In Version 1.5 musste sich der Webmaster damit zufrieden geben, dass der Inhalt einer Seite nur in Sektionen und Kategorien gegliedert werden konnte, wobei eine Sektion mehrere Kategorien gehalten hat. Mittlerweile verfügt Joomla! über eine ausgereifte Kategorie Struktur, die es erlaubt, Kategorien anzulegen und beliebig zu verschachteln. Diese Möglichkeit, Kategorien hierarchisch aufzubauen erlaubt es jeder Kategorie eine Überkategorie zuzuordnen und jede Unter-

¹³ <http://www.joomla.org/core-features.html>

¹⁴ <http://www.joomla.org>

¹⁵ <http://www.joomla.de>

¹⁶ <http://help.joomla.org>

¹⁷ http://docs.joomla.org/ACL_Tutorial_for_Joomla_1.6#Global_Configuration_2

¹⁸ http://docs.joomla.org/ACL_Tutorial_for_Joomla_1.6#Overview_of_ACL_in_Version_1.6

Kategorie kann beliebig in weitere Kategorien aufgeteilt werden.¹⁹ Joomla! wird nun auch Seiten gerecht, die eine uneingeschränkte Kategorie Struktur benötigen, wie es zum Beispiel von Zeitschriften oder Magazinen verlangt wird. Ein weiteres Feature ist die verbesserte Modulkonfiguration, welche es ermöglicht, die Anzeige von Modulen auf allen Seiten, nur auf bestimmten Seiten oder auf allen bis auf selektierte Seiten anzuzeigen.²⁰²¹ Somit besteht die Möglichkeit abhängig vom Inhalt nur noch thematisch passende Module anzuzeigen. Im Bereich Suchmaschinenoptimierung kann global eingestellt werden, dass alle Artikel im HTML Title-Tag den Namen der Seite als Prä- oder Suffix erhalten. Global kann weiterhin konfiguriert werden, dass Joomla! suchmaschinenfreundliche URLs verwendet und ob mit der Apache Erweiterung mod_rewrite die URLs überschrieben werden sollen, so dass aus www.meinseite.de/index.php/componentX www.meinseite.de/componentX wird. Zusätzlich können zu den globalen Meta-Einstellungen jedem Artikel, Menüeintrag, Modul oder Kategorie Meta-Keywords und Meta-Beschreibungen hinzugefügt werden. Eine weitere nützliche Komponente ermöglicht es, Artikel, die es nicht mehr gibt oder deren URLs sich geändert haben, auf eine andere Adresse umzuleiten um zu vermeiden, dass der Benutzer auf eine Fehlerseite weiter geleitet wird. Letztlich ist noch zu erwähnen, dass Joomla! immer mehr auf effiziente Arbeitsabläufe hinzielt. Bei der Erstellung neuer Inhalte, Menüeinträge, Benutzern oder Zugriffsrechten, erscheint im back-end als Hover-Eintrag neben den genannten Einträgen ein Menüpunkt, der es zum Beispiel erlaubt, einem Menü direkt einen neuen Unterpunkt hinzuzufügen (siehe Abbildung 3). Artikel können nicht mehr nur gespeichert werden, es



Abbildung 3. Effiziente Arbeitsabläufe, ein Beispiel

besteht nun auch die Möglichkeit über einen Button den aktuellen Beitrag zu speichern und direkt einen neuen Beitrag anzulegen. Gleichzeitig kann man noch über einen weiteren Button eine Kopie des aktuellen Beitrags speichern, um so ggf. eine Art Versionierung zu erzeugen oder registrierten Benutzern einen ausführlicheren Inhalt gegenüber einem nicht registrierten Benutzer bietet. Diese Toolbar (siehe Abbildung 4) mit den Buttons für Speicheroptionen und Schließen eines Objekts zieht sich durch das gesamt back-end und vereinfacht den Umgang mit dem back-end und dessen Arbeitsabläufe. Eine weitere Effizienzverbesserung ist, dass Joomla! sich merkt, an welchem Artikel oder an welcher

¹⁹ <http://community.joomla.org/blogs/community/1237-categories.html>

²⁰ http://docs.joomla.org/What's_new_in_Joomla_1.6#Module_Enhancements

²¹ Joomla! 1.6: A User's Guide Third Edition, S. 118



Abbildung 4. Einheitlich strukturierte Toolbar

Komponente man gerade arbeitet. Wenn zum Beispiel ein Beitrag geöffnet ist, aber nicht weiter editiert wird, kann es vorkommen, dass die Sessionzeit für den aktuellen Benutzer abläuft und dieser automatisch vom System abgemeldet wird. Bei erneutem Anmelden leitet Joomla! den Benutzer direkt wieder an die Stelle, an der er zuletzt im System gearbeitet hat.

3.2 Entwickler

Seit Version 1.5 hat die Joomla! Gemeinde darauf geachtet, den Einstieg für Programmierer zu erleichtern. Mit dem Release der Version 1.6 haben sich noch weitere Änderungen und Erleichterungen für Entwickler ergeben.²² Das Hauptaugenmerk wird dabei auf die Einhaltung des MVC-Patterns gelegt, welches dem Entwickler in vielerlei Hinsicht v.a. bei der Entwicklung von Komponenten helfen soll. Zum einen soll durch den Einsatz des MVC-Patterns erreicht werden, keinen unnötigen doppelten Code zu schreiben und zum Anderen soll eine Abstraktion zwischen Präsentations-, Applikationslogik und Datenverarbeitung eingehalten werden. Das Joomla! Framework bietet hierzu die Klassen *JModel*, *JView*, *JController* sowie einige davon abgeleitete Klassen an, zum Beispiel *JControllerAdmin* als Unterklasse von *JController*.²³ In der offiziellen Joomla! Dokumentation ist der Part der Komponenten Entwicklung am ausführlichsten dokumentiert, dies gilt sowohl für die Version 1.5 als auch für die Version 1.6.²⁴ Durch den neuen Update-Mechanismus (siehe Punkt 3.1) ist es auch möglich, Erweiterungen im back-end einfach mit zu aktualisieren. Hierzu müssen Entwickler von Erweiterungen einen Update-Server hinterlegen, welcher die neuesten Versionen der Erweiterungen zum Download anbietet.²⁵ Eine in Version 1.6 hinzu gekommene Funktion verhilft Entwicklern Voraussetzungen von anderen Modulen, Komponenten oder Plugins mit anzugeben, welche Joomla! dann automatisch vor der eigentlichen Installation der Erweiterung mit in das vorhandene System nach installiert. Joomla! bietet Entwicklern sehr viele Möglichkeiten an, das Basissystem einfach zu erweitern, so hat man auch seit Version 1.6 gleiche Ordnerstrukturen für Plugins, Module und Komponenten eingeführt, um Entwicklern die Möglichkeit zu schaffen, sich schnell in allen Erweiterungsbereichen zurecht zu finden. Entwickler von Erweiterungen können auf eine verbesserte Formular API zurück

²² http://docs.joomla.org/JController_and_its_subclass_usage_overview

²³ Mastering Joomla! 1.5 Extension and Framework Development, S. 83ff

²⁴ <http://docs.joomla.org/Developers>

²⁵ <http://docs.joomla.org/Developers>

greifen, welche es ermöglicht, „eigene Formulare innerhalb des Frameworks zu erstellen“[5] oder „Eingaben automatisch zu filtern“.[5] Zusätzlich wurden Datenbankklassen in der API erweitert²⁶ und Entwickler können auch „mit anderen Datenbankengines als MySQL“[5] kommunizieren. Die starke Förderung von modernen Programmierparadigmen, wie der Einsatz von Design-Patterns, hohe Kohäsion und durchgängige Ordner- und Dateistrukturen ermöglichen einen schnellen und professionellen Einstieg für Entwickler.

3.3 Designer

Designern oder front-end Entwicklern, welche sich ausschließlich mit der grafischen Repräsentation und Entwicklung von Templates beschäftigen, bietet Joomla! viele Funktionen und Methoden an, um anspruchsvolle und individuelle Templates zu gestalten. Joomla! bietet Designern fünf verschiedene Templates (siehe Punkt 2.3), die oft als Vorlage für die Erstellung neuer Templates dienen. Anfangen kann ein Designer, indem er sich ein vorhandenes Template und dessen Modulpositionen genauer betrachtet. Dies erreicht man, wenn man an eine URL `?tp=1` als Parameter mit angibt. Voraussetzung ist, dass diese Darstellung über den Schalter Optionen im Template Manager aktiviert ist.

Seit Version 1.6 profitieren front-end Entwickler von weiteren hilfreichen Features, wie Template Styles oder Layout Overrides. Mit Hilfe von Template Styles können innerhalb der Template Konfiguration Änderungen am eigentlichen Template vorgenommen werden, zum Beispiel könnte man erlauben, dass das ein Bild im Header-Bereich austauschbar ist oder sich das Hauptmenü auf der linken oder rechten Seite des sichtbaren Bereichs befinden soll. Um Einstellungen direkt zu übernehmen, kann man das Template und dessen Änderungen direkt speichern oder man kann die Änderungen als Kopie speichern und sich somit unterschiedliche Styles eines Templates speichern und einsetzen oder ausprobieren. Layout Overrides ermöglichen es, Module, Komponenten sowie Kategorie- oder Menüeinträge unterschiedlich zu gestalten und somit die vom System vorgesehenen Layouts unkompliziert zu überschreiben und nach eigenen Wünschen anzupassen. Hierzu muss beispielsweise für ein Modul Layout Override im *html*-Ordner des Templates ein Ordner mit dem Namen des zu überschreibenden Moduls bereit gestellt werden, zum Beispiel *mod_login*, in dem entsprechende Dateien hinterlegt werden müssen, damit Joomla! innerhalb der erweiterten Einstellungen in einer Modulkonfiguration die neue Layout Option findet und ggf. anwenden kann (siehe Abbildung 5).²⁷

4 Sicherheitsrelevantes

Joomla! ist ein browserbasiertes CMS, das sich gegen Sicherheitsangriffe im Internet bewähren muss. Für die Sicherheit und Stabilität von Joomla! ist eine

²⁶ <http://docs.joomla.org>

²⁷ <http://docs.joomla.org>

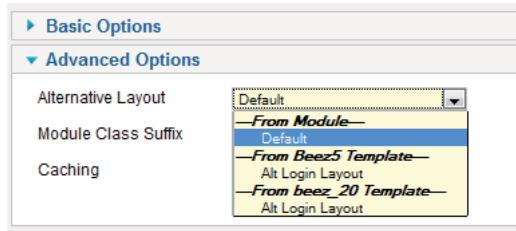


Abbildung 5. Joomla! Modul Layout Override Beispiel

eigens eingerichtete Task-Force – Joomla! Security Strike Team – zuständig, welche aus Entwicklern und Sicherheitsexperten besteht, die sich ausschließlich um sicherheitsrelevante Themen rund um Joomla! kümmern.²⁸ An die Task-Force können auch Bugfixes, Patches oder Nachrichten bzgl. Sicherheitsbedenken geschickt werden. Die Joomla! Dokumentation²⁹ ist der zentrale Ort, wenn man eine Joomla! Installation sicher nutzen und nachhaltig schützen möchte. Die Joomla! Sicherheitsdokumentation richtet sich an Administratoren und bietet eine sehr ausführliche Sicherheits-Checkliste³⁰ an, welche aufgeteilt ist in allgemeine Sicherheitsrisiken einer Webanwendung, Hosting und Server Konfiguration, Test und Entwicklung, Joomla! Installation und Administration inklusive Hinweise und Anleitungen³¹ um die Seite nach einem Verlust von Daten oder nach einem Angriff³² wieder in ihren Ausgangszustand zu bringen. Neben diversen Sicherheits-Checklisten werden sicherheitsrelevante Themen wie Joomla! Basis, Joomla! Erweiterungen, PHP- und Apache-Konfiguration behandelt. Zusätzlich zu diesen Themen wird auch oft auf Foreneinträge hingewiesen³³, die die Sicherheit einer Joomla! Installation diskutieren. Es werden außerdem extra Newsfeeds über Sicherheitsupdates oder -warnungen angeboten.³⁴ Des Weiteren gibt es auch eine Liste von Erweiterungen,³⁵ die man aufgrund von Sicherheitsbedenken nicht einsetzen sollte.

5 Schwächen und Anregungen

Joomla! als modernes und erfolgreiches CMS weist trotz großem Funktionsumfang und Beliebtheit einige Schwächen auf. Ein Beispiel ist der in einer HTML- oder Flash-Variante verfügbare Upload Manager innerhalb des Media Managers. Dieser kann Dateien wie Bilder oder Videos hochladen, die später

²⁸ <http://developer.joomla.org/security>

²⁹ <http://docs.joomla.org/Security>

³⁰ http://docs.joomla.org/Security_Checklist_1_-_Getting_Started

³¹ http://docs.joomla.org/Security_and_Performance_FAQs#Site_Recovery

³² http://docs.joomla.org/Security_Checklist_7

³³ <http://forum.joomla.org/viewforum.php?f=621>

³⁴ <http://feeds.joomla.org/JoomlaSecurityNews>

³⁵ http://docs.joomla.org/Vulnerable_Extensions_List

für Artikel zur Verfügung stehen. Laut dem Änderungsprotokoll³⁶ wurde der Upload-Mechanismus von Version 1.5 auf Version 1.6 überarbeitet, aber die Flash-Variante ist teilweise immer noch unbenutzbar und es wurde offensichtlich wenig Mühe in diesen Bereich des CMS investiert. Hier empfiehlt es sich über die Optionen den HTML-Uploader als Standard zu aktivieren. Ebenso benötigt der integrierte What You See Is What You Get (WYSIWYG) Editor TinyMCE deutlich mehr Funktionen, um mit anderen CMS und deren WYSIWYG Editoren konkurrieren zu können. Trotz Überarbeitungen hat Joomla! in diesen beiden Bereichen noch deutlich Verbesserungspotential. Abhilfe schaffen hier Erweiterungen, die den integrierten WYSIWYG Editor ersetzen und erweiterte Features zur Verfügung stellen. Ein zu empfehlender Editor wäre *JCE*³⁷ der für alle Versionen von Joomla! erhältlich ist. Joomla! scheint oft andere Schwerpunkte zu setzen als Funktionalitäten des Editors zu verbessern und Uploads von Dateien besser zu integrieren, da auch die Ein-Klick Update Funktion bzgl. des Basisystems und Erweiterungen lange gebraucht hat, um den Weg in ein Release zu finden. Das Gleiche gilt für das Feature, Erweiterungen oder Templates per FTP in ihre entsprechenden Verzeichnisse hochzuladen und dann per Joomla! zu prüfen bzw. zu entdecken (*discover*) ob es Pakete gibt, die installiert werden müssen. Als weitere Schwäche ist der Support von nur einer Datenbankengine³⁸ zu nennen, auch hier könnte Joomla! wie zum Beispiel Drupal³⁹ noch weitere Datenbanksysteme unterstützen und somit mehr Attraktivität durch eine unabhängigeren Datenbankabstraktionsschicht erreichen. Joomla! ist standardmäßig nicht Multi-Domain fähig⁴⁰ und kann somit jeweils nur für eine Domain eingesetzt werden. Größere Projekte, die dieses Feature voraussetzen, scheitern somit aufgrund dieser Schwäche und greifen schneller zu Konkurrenten wie Typo3.⁴¹ Letztlich gibt es noch organisatorisches Verbesserungspotential hinsichtlich der online Dokumentation⁴², welche sehr viele umfassende Artikel enthält und sowohl Einsteiger als auch ambitioniertere Joomla! Enthusiasten nützliches Informationsmaterial bietet, jedoch darunter leidet, dass Artikel für insgesamt vier Versionen zur Verfügung stehen. Die Community hat es noch nicht geschafft, hier eine klare Struktur zu generieren, denn oft gibt es einzelne Berichte für bestimmte Versionen, oft werden aber auch alle Versionen gemischt behandelt. Weiterhin ist noch zu erwähnen, dass Joomla! sein Framework und dessen API mittlerweile als *Platform* bezeichnet, dies jedoch nur sehr unübersichtlich bis gar nicht dokumentiert ist und Beiträge zum Framework⁴³ inhaltlich Beiträgen zu Platform⁴⁴ übereinstimmen. Die Entwickler und Verantwortlichen der Dokumen-

³⁶ http://docs.joomla.org/What's_new_in_Joomla_1.6#Media_Manager

³⁷ <http://www.joomlacontenteditor.net>

³⁸ <http://www.joomla.org/technical-requirements.html>

³⁹ <http://drupal.org/requirements>

⁴⁰ <http://www.joomlaportal.de/beta-howtos/58249-joomla-multi-domain-f-hig-23.html>

⁴¹ <http://wiki.typo3.org/De:Multidomain>

⁴² <http://docs.joomla.org>

⁴³ <http://docs.joomla.org/Framework/1.6>

⁴⁴ <http://docs.joomla.org/Platform/11.1>

tation müssen hier schnell Klarheit schaffen, um Joomla! Anfänger nicht durch eine undurchsichtige Hilfestellung abzuschrecken. Für Joomla! gibt es sehr viele Bücher, fast alle befassen sich zum Zeitpunkt der Erstellung dieser Arbeit jedoch noch mit der Version 1.5. Die Themen der zur Verfügung stehenden Literatur beinhalten sehr oft Anleitungen für eine erfolgreiche Joomla! Installation und sind häufig als erweiterte Benutzerdokumentation zu sehen.⁴⁵ Insgesamt gibt es zu wenige aktuelle Bücher, die sich ausschließlich mit der Entwicklung von Erweiterungen – konkret für Version 1.7 – befassen. Hier muss man auf die online Dokumentation und zahlreiche Anleitungen in Blogs oder Foren zurück greifen.

6 Fazit

Viele in dieser Arbeit erläuterten Funktionen und Features des Joomla! CMS gehen besonders auf die aktuellste Version und somit neu erschienenen Funktionen ein. Dies lässt sich damit begründen, dass in Version 1.6 die wichtigsten und interessantesten Features Rund um Joomla! eingeführt oder verbessert wurden und die momentan aktuelle Version 1.7 hauptsächlich Verbesserungen der Version 1.6 enthält. Joomla! ist durch seine genannten Neuerungen auf einem sehr guten Weg, weiterhin auf dem belebten CMS Markt zu bestehen. Es liefert von Haus aus sehr viele Funktionen mit, die schon in der Grundkonfiguration ausreichend sind, um innerhalb kurzer Zeit eine komplette Website auf Basis des Joomla! CMS aufzubauen und produktiv zu nutzen. Hier spielen auch die vielen fertigen Templates eine große Rolle, die nun noch leichter zu erweitern sind und somit schnell individuelle Seiten erstellt werden können. Auch in Richtung zukünftige Web-Technologien hat Joomla! einen Schritt nach vorne gemacht, in dem ein Template (*beez5*) den HTML5 Standard unterstützt. Joomla! überzeugt durch ein sehr ausgereiftes und mit hohem Funktionsumfang ausgestatteten Kernsystem, das stets verbessert wird und mit nur sehr wenig Aufwand produktiv einsetzbar ist. Zusätzlich gibt es noch tausende⁴⁶ Erweiterungen, die Joomla! auf fast alle Bedürfnisse hin anpassen kann. So kann Joomla! zum Beispiel durch die populäre Erweiterung *VirtueMart*⁴⁷ schnell zu einem Online-Shop erweitert werden. Die über 8000⁴⁸ Erweiterungen im Joomla! Extension Directory⁴⁹ erfüllen sämtliche Erweiterungswünsche von administrativen Erweiterungen hin zu e-Commerce Systemen. Kritisch hinterfragen und ausführlich testen sollte man jedoch jede neue Joomla! Installation und jede neu installierte Erweiterung. Im Bereich Erweiterungen programmieren hat Joomla! deutlich nachgebessert und vielen Entwicklern einen einfachen Einstieg ermöglicht. Dies ist einerseits sehr positiv zu sehen, da sich zum Beispiel für Firmen spezielle Erweiterungswünsche mit absehbarem Aufwand umsetzen lassen, jedoch auch viele qualitativ schlechte Erweiterungen zu beziehen sind und man hier sehr ausführlich testen

⁴⁵ A User's Guide Third Edition

⁴⁶ <http://www.joomla.org/core-features.html>

⁴⁷ <http://virtuemart.net/>

⁴⁸ <http://extensions.joomla.org/>

⁴⁹ <http://extensions.joomla.org/>

sollte, ob die installierte Erweiterung allen funktionalen und sicherheitsrelevanten Ansprüchen genügt.

Joomla! wird nachgesagt, dass man sofort erkennt, dass es sich um eine Joomla! basierte Seite handelt,⁵⁰ auch hier hat Joomla! im Bereich Templates weiter gearbeitet und neben vielen frei verfügbaren Templates gibt es professionelle Templates, die u.a. von sogenannten Template-Clubs erworben werden können und sehr individuelle Designs anbieten. Bei der Erstellung eigener Templates darf man den eigentlichen Aufwand nicht unterschätzen, auch wenn hier nur Kenntnisse in HTML und CSS gefordert sind. Um hochwertige und einmalige Templates zu erstellen, bedarf es noch weiteren Kenntnissen im Bereich Web-Design, JavaScript und PHP.

Joomla! ist sehr empfehlenswert, da es sowohl im back-end als auch im front-end flexibel anpassbar und erweiterbar ist. Für ein gelungenes Joomla! Projekt sollten jedoch alle Anforderungen im Vorfeld geprüft werden.

Literatur

1. North, B.: Joomla! 1.6: A User's Guide Third Edition, Prentice Hall (2011)
2. Ebersbach, A.: Joomla 1.5! 2. Auflage, Galileo (2008) <http://openbook.galileocomputing.de/joomla15/>
3. Kennard, J.: Mastering Joomla! 1.5 Extension and Framework Development, Packt Publishing (2007)
4. Eddie, A.: The Official Joomla! {book}, Pearson Education Inc. (2010)
5. Deutschsprachige Landeseite des CMS Joomla! <http://www.joomla.de>
6. Online Dokumentation Joomla! CMS <http://www.docs.joomla.org>
7. Offizielle Joomla! Homepage <http://www.joomla.org>
8. OpenSource Matters Homepage <http://opensourcematters.org/>

⁵⁰ <http://www.joomla.de/entdecken/versionen/versionsreihe-16x/funktionen.html?start=1>

Drupal

Maximilian Geissler

Otto-Friedrich-Universität Bamberg

Zusammenfassung Dieser Artikel gibt einen groben Überblick über den Aufbau des Content Management Systems Drupal, seine Grundfunktionalitäten und ausgewählte Zusatzmodule.

1 Einleitung

Drupal ist ein Content Management System für den Betrieb von Webseiten. Es wurde ursprünglich von Dries Buytaert als Austauschplattform für Freunde an der Universität von Antwerpen entwickelt. Anfang 2001 veröffentlichte er dann Drupal 1.0 als Open Source, zurzeit ist die Version 7.7 aktuell. Mittlerweile betreibt Dries Buytaert seine eigene Firma Acquia¹ die neben einer Vielzahl weiterer Firmen² professionellen Support für Drupal bietet. Nach Wordpress und Joomla ist Drupal mit 6% Marktanteil derzeit die Nummer drei³ unter den Content Management Systemen.

2 Eigenschaften

- Drupal ist prinzipiell nicht abwärtskompatibel (es wird keine Abwärtskompatibilität garantiert)
- alle Unterseiten einer Drupalseite sind unter Permalinks erreichbar
- strikte Trennung von Layout und Design
- zur Programmierung von Drupal sind nur XHTML, CSS und PHP nötig

2.1 Datenbankabstraktionsschicht

Die auf PHPs PDO (PHP Data Objects) Datenbank API aufsetzende Database API⁴ von Drupal ermöglicht Entwicklern die einfache Unterstützung mehrerer Datenbanken.

¹ <http://acquia.com>

² <http://drupal.org/marketplace-preview>

³ http://w3techs.com/technologies/overview/content_management/all

⁴ www.drupal.org/developing/api/database

2.2 Installationsprofile

Installationsprofile bündeln die Konfiguration einer Drupalinstallation und ermöglichen diese anderen zu Verfügung zu stellen. Durch eine Reihe an schon vorhandenen Installationsprofilen⁵ ist es möglich ein für einen bestimmten Einsatzzweck vorkonfiguriertes Drupal zu installieren bzw. selbst für den eigenen Bedarf ein Installationsprofile zu erstellen⁶ um den Aufwand für die Einrichtung mehrerer ähnlicher Drupalinstallationen zu verringern.

3 Installation

Die Installation von Drupal gestaltet sich durch die gute Installationsbeschreibung⁷ und das beim ersten Start aufgerufene Installationsskript (siehe Abb.1), das Schritt für Schritt durch die Installation führt, relativ einfach.

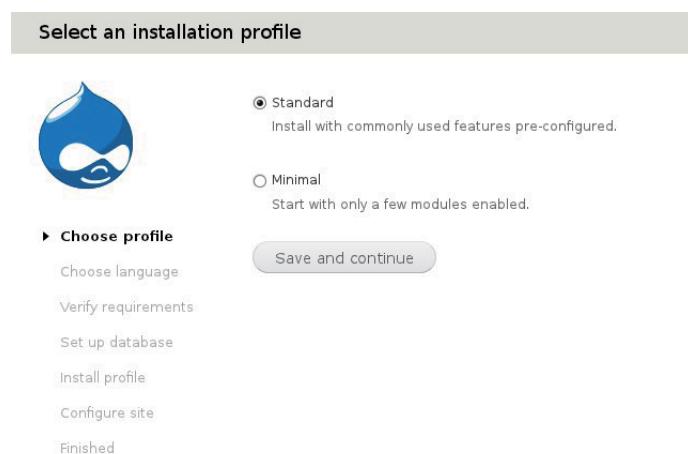


Abbildung 1. Installationsskript

3.1 Installationsvoraussetzungen für Drupal 7 [1]

Webserver: Apache (empfohlen) 1.3 oder 2.x, Microsoft IIS 5,6, oder 7

Datenbank: MySQL (empfohlen) 5.0.15 oder höher (bzw. äquivalente wie z.B. MariaDB), PostgreSQL ab 8.3, SQLite ab 3.3.7

PHP: 5.2.5 oder höher (empfohlen 5.3)

⁵ <http://drupal.org/project/installation+profiles>

⁶ <http://drupal.org/node/159730>

⁷ www.drupal.org/documentation/install

4 Aufbau

Drupals grundlegender Aufbau ist in fünf Schichten unterteilt (siehe Abb.2).

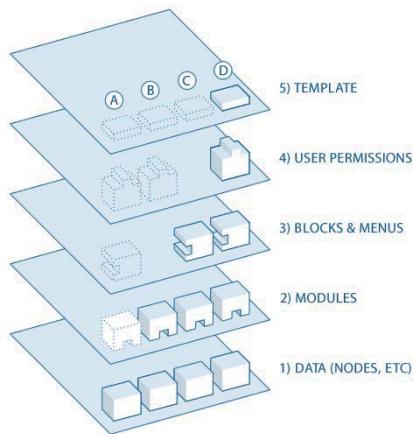


Abbildung 2. Der Aufbau von Drupal [1]

4.1 Templates

Das Aussehen einer Drupalwebsite wird durch das Theme bestimmt. Es stehen bereits hunderte Themes zur Verfügung⁸. Diese lassen sich direkt aus dem Administrationsbereich von Drupal installieren und farblich sowie inhaltlich individuell anpassen. Ein Theme besteht aus der info-Datei, in der die einzelnen Bereiche der Seite und ein Stylesheet definiert werden, der zugehörigen css-Datei sowie weiteren php-Dateien die die Anzeige von themebaren Inhalten bestimmen (siehe Abb.4). Durch die Möglichkeit Subthemes definieren zu können ist es schnell möglich ein Theme auf die eigenen Wünsche anzupassen⁹, da nur die zu ändernden Teile neu erstellt werden müssen.

4.2 User Permissions

Mit Hilfe der Rechte wird geregelt welchem Nutzer was erlaubt auf der Webseite erlaubt ist. Benutzerrechte sind in Drupal rollenbasiert. Standardmäßig werden in Drupal 7 folgende Rollen unterschieden:

⁸ <http://drupal.org/project/themes>

⁹ <http://drupal.org/theme-guide/6-7>

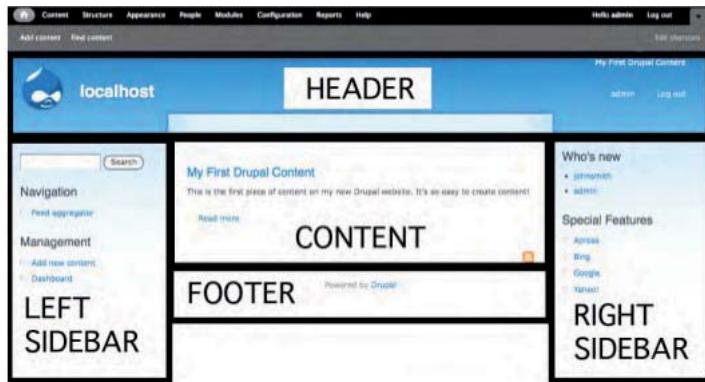


Abbildung 3. Standardbereiche eines Theme [2]

- anonymous user
- authenticated user
- Administrator

Es lassen sich aber auch beliebige eigene Rollen definieren und diesen Benutzer zuordnen. Durch die detaillierte Rechteverwaltung wird es erleichtert Nutzern einer Community selbst die Möglichkeit zu geben sich an der Erstellung von Inhalten zu beteiligen.

4.3 Blocks & Menus

Blocks sind kleine Einheiten für die Anzeige des Outputs von Modulen oder beliebigem selbst definiertem Inhalt (z.B. die 3 neusten Beiträge zu einem Thema). Menus sind die Navigationselemente der Seite und können genauso wie Blocks an dafür vorgesehenen Stellen im Layout der Seite positioniert werden. Mittels der Rollen lässt sich die Anzeige von einzelnen Blocks und Menus auf bestimmte Benutzergruppen einschränken. Es lässt sich aber auch allgemein definieren auf welchen Seiten welche Blocks und Menus überhaupt sichtbar sein sollen.

4.4 Modules

Die Module sind für die Funktionalität von Drupal zuständig. Sie werden unterteilt in core Modules (standardmäßig enthalten) und contributed Modules (Zusatzmodule die von der Community zur Verfügung gestellt werden). Es stehen bereits tausende Zusatzmodule¹⁰ für eine Fülle an Funktionen zur Verfügung wie z.B. Pathauto¹¹ für die Generierung von sprechenden Pfadnamen oder Webform¹² für die Erstellung von Umfragen.

¹⁰ <http://drupal.org/project/modules>

¹¹ <http://drupal.org/project/pathauto>

¹² <http://drupal.org/project/webform>

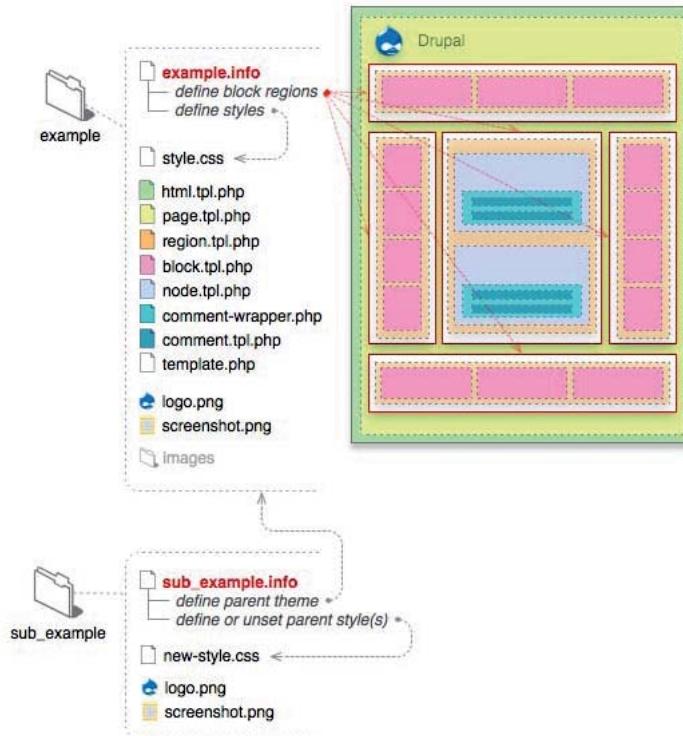


Abbildung 4. Aufbau eines Theme [1]

4.5 Data

Die Daten (der eigentliche Inhalt der Seite) sind in Drupal in sogenannten Nodes organisiert. Ein Node besteht neben dem Inhalt (z.B. der Text eines Artikels) aus den zugehörigen Metadaten (z.B. Autor, Titel, Tags). Diese Standardisierung ermöglicht den standardisierten Zugriff von Modulen auf die Daten. Somit ist es möglich genau zu spezifizieren wie Inhalte auf der Seite angezeigt werden sollen.

5 Grundfunktionalitäten

Drupal bietet von Haus aus jeweils Module für die Erstellung von Foren, Blogs (auch für jeden Benutzer einen eigenen Blog ist möglich) und Kommentaren zu Inhalten.

5.1 Konfiguration

Der Administrationsbereich von Drupal ist nicht strikt von der eigentlichen Webseite getrennt. Benutzer mit den entsprechenden Rechten können viele Elemente

direkt editieren (siehe Abb.5). Des Weiteren lassen sich in den Konfigurationsmenüs Elemente oft per Drag & Drop arrangieren bzw. sortieren. Dies gestaltet die Bedienung, gerade für Einsteiger, sehr intuitiv.



Abbildung 5. Konfigurieren eines Blocks

5.2 Taxonomiesystem

Ein sehr mächtiges Werkzeug von Drupal ist das Taxonomiesystem¹³. Es besteht aus selbst erstellbaren Vokabularen einzelner Themenbereiche. Ein Vokabular besteht wiederum aus einzelnen Begriffen (siehe Abb.6). Diese lassen sich beliebig tief hierarchisch Schachteln, sogar Multihierarchien sind möglich (ein Begriff kann mehreren Wurzelementen zugeordnet sein). Mit Hilfe dieses Systems lassen sich die Inhalte einer Webseite effektiv kategorisieren.

5.3 Contenttype

Standardmäßig stehen die beiden Inhaltstypen Article und Basic Page zur Verfügung. Es lassen sich aber auch beliebige eigene Inhaltstypen definieren. Ein Inhaltstype definiert welche Eingabefelder (z.B. die Möglichkeit ein Bild hinzuzufügen) für einen bestimmten Inhalt vorhanden sein sollen (siehe Abb.7). Hier kommt dann auch wieder das Taxonomiesystem ins Spiel, da es möglich ist Eingabefelder bestehend aus einem Dropdown bzw. Autocomplete Field gefüllt mit den Begriffen eines bestehenden Vokabulars hinzuzufügen. Hierdurch lassen sich Inhalte beim Erstellen effektiv taggen. Dies kann durch die Option Eingabefelder als verpflichtend kennzeichnen zu können auch erzwungen werden.

¹³ <http://drupal.org/documentation/modules/taxonomy>

NAME	OPERATIONS	
+	Bundesliga	edit
+	Mannschaft 1	edit
+	Mannschaft 2	edit
+	Mannschaft 3	edit
+	Regionalliga	edit
+	Mannschaft 4	edit

Abbildung 6. Beispiel für ein Taxonomiesystem

5.4 Updates

Für die Sicherheit von Drupal ist ein eigenes Sicherheitsteam¹⁴ zuständig, das immer die aktuelle und die vorherige Version mit Sicherheitsupdates versorgt. Drupal veröffentlicht zu jeder entdeckten Sicherheitslücke die Art des Angriffs. Ob die eigene Installation auf dem aktuellsten Stand ist lässt sich über den im Administrationsbereich verfügbaren Report „Available updates“ überprüfen (siehe Abb.8). Hier wird übersichtlich dargestellt auf welchem Stand sich die eigene Installation befindet und man kann gegebenenfalls vorhandene Updates direkt über den eingebauten Updatemechanismus einspielen.

6 Ausgewählte Zusatzmodule

6.1 Views

Das Zusatzmodul Views¹⁵ ermöglicht die einfach Erstellung eigener Sichten auf die Inhalte der Webseite. Wie in Abb.9 zu sehen ist, lässt sich definieren welche Inhaltstypen, getagged mit welchem Begriff des Taxonomiesystem, in einer Sicht angezeigt werden sollen. Aus dieser Sicht kann dann eine Seite und/oder ein Block erstellt werden. So lässt sich z.B. schnell ein Block mit den fünf neusten Beiträgen zu einer bestimmten Tag erstellen.

6.2 Panels

Reicht die standardmäßige Layouteinteilung (siehe Abb.3) der meisten Themes nicht aus und will man nicht gleich selbst ein eigenes Theme erstellen bzw. ein

¹⁴ <http://drupal.org/security-team>

¹⁵ <http://drupal.org/project/views>

LABEL	NAME	FIELD	WIDGET	OPERATIONS
⊕ Title	title	Node module element		
⊕ Body	body	Long text and summary	Text area with a summary	edit delete
⊕ Mannschaft	field_basket	Term reference	Autocomplete term widget (tagging)	edit delete
⊕ Bild	field_image	Image	Image	edit delete
Add new field Label: <input type="text"/> Field name (a-z, 0-9, _) Type of data to store: <input type="button" value="- Select a field type -"/> Form element to edit the data: <input type="button" value="- Select a widget -"/>				
Add existing field Label: <input type="text"/> Field to share: <input type="button" value="- Select an existing field -"/> Form element to edit the data: <input type="button" value="- Select a widget -"/>				

Save

Abbildung 7. Ein eigener Contenttype

schon vorhandenes Theme an seine Wünsche anpassen, bietet das Zusatzmodul Panels¹⁶ die Möglichkeit Seiten nach einigen vorgegebenen bzw. selbst erstellten Layouts zu unterteilen (siehe Abb.10). Den einzelnen Bereichen lassen sich dann wie bei einer normalen Seite Inhalte zuordnen.

6.3 Backup and Migrate

Backup and Migrate¹⁷ ist ein sehr nützliches, einfach bedienbares Zusatzmodul um regelmäßig Sicherungen seiner Webseite anlegen zu können (siehe Abb.11). Neben der Möglichkeit manuell eine Sicherung zu erstellen besteht die Möglichkeit Zeitplan gesteuert automatisch Sicherungen anzulegen. Was alles gesichert werden soll kann durch die Erstellung von Sicherungsprofilen definiert werden und auch die Wiederherstellung einer Sicherung ist mit diesem Modul möglich.

6.4 Drush

Für den erfahrenen Drupalentwickler ist Drush¹⁸¹⁹ eine große Arbeitserleichterung. Hierbei handelt es sich nicht um ein Zusatzmodul für Drupal, es bietet aber Kommandozeilenzugriff und ein Scripting Interface auf bzw. für die Funktionen von Drupal. Drush wird seinerseits von einer Reihe an Zusatzmodulen

¹⁶ <http://drupal.org/project/panels>

¹⁷ http://drupal.org/project/backup_migrate

¹⁸ <http://drupal.org/project/drush>

¹⁹ <http://drush.ws/>

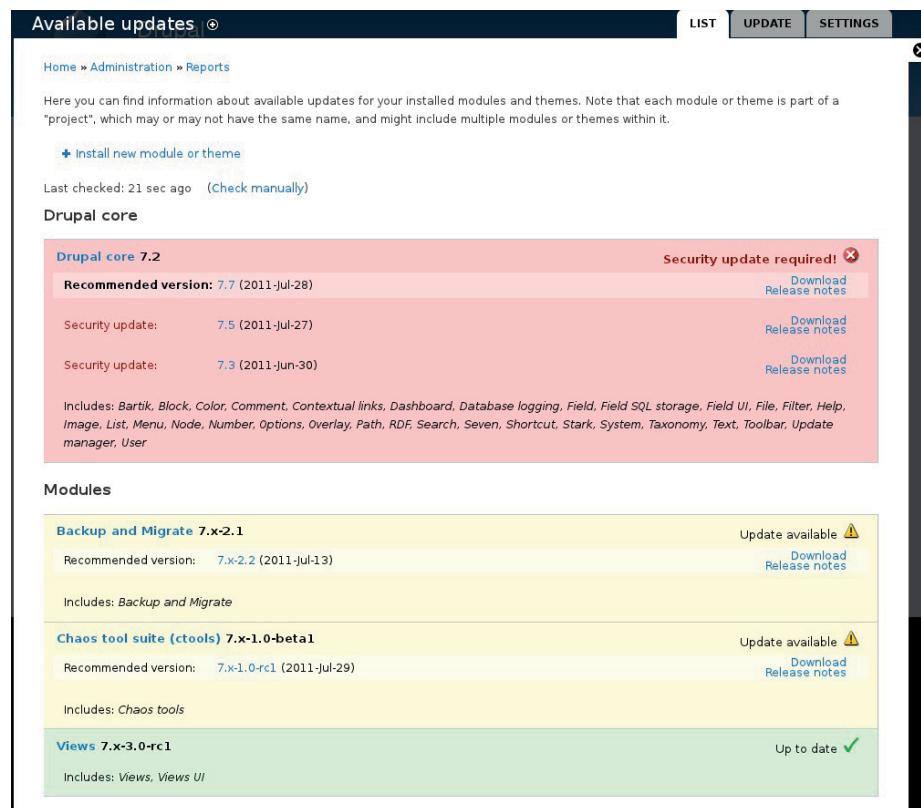


Abbildung 8. Report über vorhandene Updates

unterstützt. Wodurch auch deren Funktionalitäten über die Kommandozeile erreichbar werden.

7 Fazit

Drupal ist durch die gute Unterstützung bei der Installation (speziell den Installationsprofilen) leicht einzurichten. Auch die intuitive Bedienung (Drag & Drop) und die Möglichkeit, durch die nicht strikte Trennung von Administrationsbereich und eigentlicher Seite, die meisten Elemente direkt auf der Seite editieren zu können, erleichtern den Einstieg. Die Vielzahl schon vorhandener Zusatzmodule und Themes ermöglichen die individuelle Anpassung und lassen kaum Wünsche offen. Von der kleinen Vereinssite über den Webshop bis zum Betrieb großer Webseiten wie z.B. dem Internetauftritt des Weißen Hauses²⁰ ist alles möglich. Dies ist einerseits die Stärke von Drupal andererseits aber auch eine

²⁰ <http://www.whitehouse.gov>

The screenshot shows the 'Add new view' dialog in the Drupal admin interface. At the top right are 'LIST' and 'SETTINGS' buttons. Below them is a navigation bar: Home > Administration > Structure > Views. The main area is titled 'Add new view'. It has a 'View name *' field, a 'Description' checkbox, and a 'Show' configuration section with dropdowns for 'Content' type and 'All' status, and a 'sorted by' dropdown set to 'Newest first'. A large central panel contains several sections: 'Create a page' (checkbox checked, 'Page title' and 'Path' fields), 'Display format' (dropdown set to 'Unformatted list' with options 'teasers', 'with links (allow users to add comments, etc.)', and 'without comments'), 'Items to display' (input field set to 10), 'Create a menu link' (checkbox unchecked), and 'Include an RSS feed' (checkbox unchecked). Below this is a 'Create a block' section (checkbox unchecked). At the bottom are 'Save & exit', 'Continue & edit', and 'Cancel' buttons.

Abbildung 9. Erstellung eines eigenen Views

Schwäche, da es für spezielle Einsatzzwecke (z.B. Weblogs) spezialisierte Content Management Systeme wie z.B. Wordpress²¹ gibt. Wer sich selber einmal einen Eindruck von Drupal machen will findet dazu in den Drupal Gardens²² die Möglichkeit. Einen Überblick darüber wie eine Seite mit Drupal aussehen kann bietet daneben auch whousesdrupal²³.

Literatur

- [1] Drupal website. www.drupal.org.
- [2] Todd Tomlinson. *Beginning Drupal 7*. Apress, 2010.
- [3] Todd Tomlinson und John K. VanDykPro. *Pro Drupal 7 Development: Third Edition*. Apress, 2010.

²¹ <http://wordpress.org/>

²² <http://www.drupalgardens.com/>

²³ <http://www.whousesdrupal.com/home>

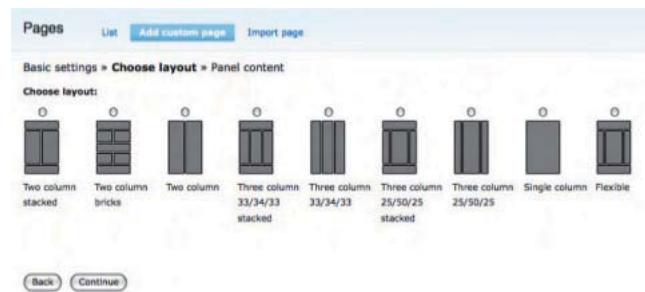


Abbildung 10. Die verschiedenen Panellayouts [2]

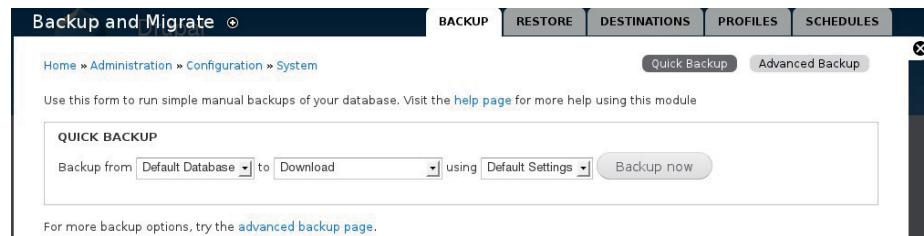


Abbildung 11. Das Modul Backup and Migrate

Spezielle CMS: LMS ILIAS

Johannes Müller

Prof. Dr. Andreas Henrich
Dipl.-Wirtsch.Inf. Stefanie Sieber
Lehrstuhl für Medieninformatik und Angewandte Informatik
Otto-Friedrich-Universität Bamberg, 96045 Bamberg
Email: andreas.henrich@uni-bamberg.de,
stefanie.sieber@uni-bamberg.de
WWW: <http://www.uni-bamberg.de/minf>

Zusammenfassung: E-Learning an Unternehmen, Verwaltungsorganisationen und vor allem an Universitäten ist ein sehr wichtiges und komplexes Thema. In dieser Arbeit soll das Learning Management System (kurz LMS) ILIAS näher betrachtet werden. Beginnend mit einer Definition eines solchen Systems und der historischen Entstehung von ILIAS im ersten Teil, sollen im zweiten Teil der Arbeit die Besonderheiten, Konzepte und Funktionen dieser Lernplattform aufgezeigt werden. Dabei werden drei verschiedene Sichten unterschieden: die Sicht eines Administrators, eines Autors und eines Lernenden. Durch die Verwendung eines HowTo ähnlichen Stils, soll ein potentieller Leser anhand dieser Arbeit die grundlegenden Kenntnisse erwerben, ein Basis-System einzurichten und sich in diesem zurechtzufinden. Auf die explizite Darstellung aller Funktionen wird verzichtet, weswegen nur ausgewählte Möglichkeiten in den einzelnen Sichten erläutert werden. Abschließend folgen einige Informationen über die ILIAS Conference und ein Ausblick auf kommende Versionen sowie das Versionsmanagement.

Schlagwörter: ILIAS, E-Learning, Learning Management System, LMS, Lernplattform, Universität zu Köln, Open Source

1 Was Ist ILIAS?

Das Einsatzgebiet von Lernplattform, wie ILIAS, ist nicht mehr zwangsweise an Universitäten gebunden. Immer mehr Unternehmen und Verwaltungsorganisationen setzen solche Systeme für den E-Learning Bereich ein. Beispiele hierfür bilden zum einen die Bundesagentur für Arbeit und zum anderen die NATO [1]. Letztere unterzog ILIAS, vor dem Einsatz als Lernplattform für das E-Learning Angebot von klassifizierten Inhalten, im Jahr 2008 einem mehrtägigen Sicherheitscheck, den das System als einzige Open Source Software bis zu diesem Zeitpunkt bestand [2].

Das Akronym ILIAS steht hierbei für Integriertes Lern-, Informations- und Arbeitskooperationssystem. Betrachtet man diese Aneinanderreihung von Worten genauer, lässt sich feststellen, dass man hier versucht hat, ein System zu kreieren, welches das Lernen, den Informationsaustausch sowohl zwischen Lehrendem und Lernendem als auch zwischen mehreren Lernenden, sowie das Zusammenarbeiten unterstützen soll.

Anders als beispielsweise die Lernplattform moodle¹, ist ILIAS hauptsächlich an Universitäten im deutschsprachigen Raum verbreitet. Laut einer Auflistung auf der offiziellen Webpräsenz² gibt es inzwischen aber auch viele internationale Installationen, wie zum Beispiel an Universitäten und Bildungseinrichtungen in Frankreich und Ungarn. Auch in den USA ist ILIAS in einigen Einrichtungen und Organisationen vertreten [3].

Folglich haben sich Dienstleistern gebildet, die individuell angepasste Implementierungen von ILIAS-Systemen gegen ein entsprechendes Entgelt anbieten. In diesem Zusammenhang ist vor allem das Unternehmen Qualitus³ zu nennen. Neben der Erarbeitung eines individuellen Konzepts, stellt das Unternehmen je nach Bedarf sowohl eine passende IT-Infrastruktur als auch eine Schulung der Mitarbeiter im Umgang mit der Lernplattform bereit [4].

Im zweiten Abschnitt dieser Arbeit soll zunächst mit der Definition, den Funktionen sowie der Architektur eines Learning Management Systems begonnen werden. Hierauf folgt ein kurzer geschichtlicher Einblick in die Entstehung ILIAS im Rahmen des Forschungsprojekts VIRTUS an der Universität zu Köln. Im Anschluss wird der ILIAS Open Source eLearning e.V. vorgestellt, worauf in Abschnitt fünf die Voraussetzungen zur Installation und der Vorgang selbst sowie die Aktualisierung des Systems durch ein Update dargestellt werden. Den Hauptteil der Arbeit bildet der sechste Abschnitt. An dieser Stelle werden die Funktionen, die Nutzern der Plattform zur Verfügung stehen, erläutert. Dabei werden zunächst die allgemeinen Funktionen dargestellt, die für alle Nutzer gelten. Im Folgenden wird näher auf ausgewählte Funktionen aus der Sicht eines Administrators, eines Autors und eines Lernenden eingegangen. Abschließend werden die ILIAS Conference und das Release Management der Lernplattform vorgestellt.

2 Learning Management System

Ein Learning Management System, kurz LMS, wird im Folgenden auch als Lernplattform bezeichnet. Im *E-Learning Praxishandbuch* werden Learning Management Systeme folgendermaßen definiert: „Unter einer webbasierten Lernplattform ist eine serverseitig installierbare Software zu verstehen, die beliebige Lerninhalte über das Internet zu vermitteln hilft und die Organisation der dabei notwendigen Lernprozesse unterstützt.“ [5]. Daraus lässt sich folgern, dass Learning

¹ <http://www.moodle.org/>

² <http://www.ilias.de>

³ <http://www.qualitus.de>

Management Systeme webbasierte System darstellen und alle Bereiche des E-Learning unterstützen können. Rolf Schulmeister hat weiterhin in seinem Werk *Lernplattformen für das virtuelle Lernen* Funktionen herausgearbeitet, die ein LMS seiner Meinung nach mitbringen sollte [6]. Dazu gehören:

- eine Benutzerverwaltung,
 - eine Kursverwaltung,
 - ein System zur Rollen- und Rechtevergabe mit differenzierten Rechten,
 - Kommunikationsmethoden und Werkzeuge für das Lernen,
 - die Darstellung von Kursinhalten, Lernobjekten und Medien im Browser und
 - die Möglichkeit zur selbstständigen Erstellung von Lerninhalten.

Zudem hat Schulmeister in diesem Werk eine idealtypische Architektur für Learning Management Systeme aufgestellt, die in Abbildung 1 dargestellt ist und drei Schichten unterscheidet.

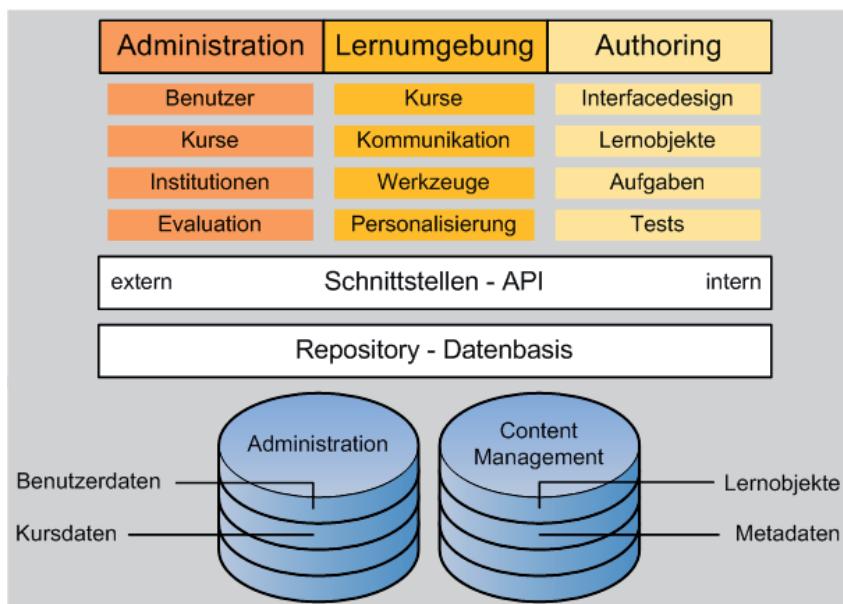


Abb. 1. Idealtypische Architektur eines LMS nach Schulmeister [6]

Die unterste Ebene bildet die sogenannte Datenbankschicht (als *Repository – Datenbasis* gekennzeichnet). An dieser Stelle werden Benutzer- sowie Kursdaten, aber auch Lernobjekte und Metadaten gespeichert, die innerhalb der Lernplattform zur Verfügung stehen sollen. Aufbauend auf dieser Schicht folgt die Schnittstellenschicht (als *Schnittstellen – API* gekennzeichnet), die sich in externe und interne Schnittstellen unterscheiden lässt. Während die erstgenannte Schnittstelle beispielweise eine Möglichkeit bietet, die Benutzerverwaltung über ein anderes, extern aufgestelltes System abzuwickeln, werden bereits in die Lernplattform integrierte Funktionen über die internen Schnittstellen angesprochen. Die oberste

Schicht kann wiederum in drei Sichtweisen gegliedert werden. In der Administrationssicht können unter anderem Benutzer und Kurse erstellt, verwaltet, gelöscht und verändert werden. Die Lernumgebung bildet den Arbeitsbereich für den Lernenden und bietet bereitgestellte Kurse zu den Lerninhalten und Kommunikationsmöglichkeiten im LMS. Die letzte der drei Sichten beschreibt die Autorensicht. Autoren können den Aufbau und die Gestaltung eines Kurses bestimmen und sind für die Integration von Lernobjekten, wie Tests und Aufgaben für Lernende, verantwortlich [6].

Im Abschnitt *Konzept und Funktionen* wird ersichtlich, dass ILIAS viele der hier aufgezeigten Eigenschaften umsetzt und der dargestellten Architektur weitestgehend entspricht.

3 Entstehung im Rahmen des Projekts VIRTUS

ILIAS wurde an der Universität zu Köln, insbesondere an der wirtschafts- und sozialwissenschaftlichen Fakultät, die bereits seit 1997 im E-Learning-Bereich forschend aktiv ist, entwickelt. Ziel war es in der Anfangszeit den *Massenbetrieb* zu beherrschen, da weit über zehn tausend Studenten an der Fakultät eingeschrieben waren, was zu einer Überlastung der Präsenzlehre führte. Aus diesem Grund hatte man sich das Ziel gesetzt, den konventionellen Lehrbetrieb durch den Einsatz von Internet und Multimedia gezielt zu unterstützen. Die Möglichkeiten, die das E-Learning bietet, sollten ebenfalls die Lehre verbessern sowie die Serviceleistungen der Fakultät und die Kommunikation zwischen Lehrenden und Lernenden [7].

Mit Beginn des Projekts zur Entwicklung Virtueller Universitätssystem (kurz VIRTUS) im Jahr 1997 wollte man diese Ziele, neben dem Schwerpunkt, der die Erstellung webbasierter Lerneinheiten bildete, erreichen. Der Begriff *virtuell* wurde im Kontext der Nutzung des Internets verwendet, während ein Universitätssystem als „fakultätsweites Konzept, das einzelne Aktivitäten bündelt und so für einen vernetzten, studiengangbezogenen Einsatz integriert werden kann“ definiert wurde [7]. Unterstützt wurde das Projekt durch verschiedene Stiftungen und Initiativen, für die das Projektteam über die Laufzeit hinweg Jahresberichte verfasste, die den Fortschritt und Probleme schriftlich festhielten.

Um die internetbasierten Lerneinheiten zur Verfügung stellen zu können, wurde noch eine geeignete Lernplattform benötigt. Im damaligen Projektteam hatte man sich Gedanken gemacht, entweder auf eine bereits vorhandene Plattform zu setzen und diese den eigenen Ansprüchen anzupassen oder eine Eigenentwicklung in Angriff zu nehmen. Ergebnis dieser *Make or Buy* Entscheidung war die Eigenentwicklung einer Lernplattform innerhalb des Projekts VIRTUS. Gründe dafür waren einerseits, dass selbst eine sorgfältige Marktanalyse keine passenden Systeme hervorgebracht hatte und andererseits, dass keines der damals aktuellen Plattformen eine der wichtigsten Anforderungen, nämlich den Aspekt der *Deutschsprachigkeit*, vollständig unterstützte. Den Entwicklern war die Integration der deutschen Sprache sehr wichtig, da sie sich so die Akzeptanz bei der Einführung des Systems an der Fakultät sichern wollten [8]. Professoren und Studenten sollten sich so leicht in das System einfinden

und mit dem System arbeiten können. Die Professoren der Fakultät wurden ebenfalls über Arbeitsgruppen an der Entwicklung beteiligt. In denen sie bereits implementierte Funktionen testen und entsprechende Verbesserungsvorschläge darbringen konnten. Mit Abschluss des Projekts im Jahr 2001 standen den Studenten der wirtschafts- und sozialwissenschaftlichen Fakultät über 40 Lerneinheiten zur Verfügung, die über die entwickelte Lernplattform ILIAS bereitgestellt wurden [7].

Inzwischen liegt ILIAS als Open Source Lernplattform in der Version 4.1.7 vom 18. Juli 2011 vor. Einen kleinen Einblick in das System verschafft die Online-Demo⁴ auf der ILIAS Website.

4 ILIAS Open Source E-Learning e.V.

Der sogenannte ILIAS open source e-Learning e.V. bildet einen Zusammenschluss aus Privatpersonen, Bildungseinrichtungen, Unternehmen und Servicepartnern. Letztere werden auch als ILIAS-Dienstleister bezeichnet und bieten entsprechende Serviceleistungen rund um die Lernplattform an. Beispiel für einen solchen Service-Provider ist das schon in der Einleitung genannte Unternehmen Qualitus.

Aufgaben des Vereins sind laut dem offiziellen Internetauftritt [9]:

- die Strategische Entwicklung von ILIAS,
- die Koordination der Softwareentwicklung,
- die Prozessgestaltung und Qualitätskontrolle,
- das Gestalten von Community-Events,
- die Repräsentation der Software nach Außen und
- das Release Management, Bugtracking sowie die Koordination der Dokumentation.

Weitere Informationen zum Verein, wie zum Beispiel die jeweiligen Aufnahmeanträge für Privatpersonen, Institutionen oder Service-Provider und die aktuelle Satzung, können auf der Vereins-Website⁵ eingesehen werden.

5 Installation und Update

ILIAS benötigt nach den Angaben auf der offiziellen Internetpräsenz einen Web-Server, der eine Kombination aus Apache, PHP und MySQL-Datenbank unterstützt. Dabei sind folgende Grundvoraussetzungen für die aktuelle Version von ILIAS zu berücksichtigen [10]:

- Apache Web-Server ab Version 2
- PHP, ab Version 5.1.4 oder höher für ILIAS ab der Version 4
- MySQL-Datenbank, ab Version 5.0.x oder höher für ILIAS ab der Version 4

⁴ <http://demo.ilias.de/>

⁵ http://www.ilias.de/docu_goto_docu_cat_1725.html

5.1 Installation eines Neuen Systems

Eine Installationsanleitung für die verschiedensten Systeme kann auf der Website von ILIAS gefunden werden⁶. Für die Erstellung dieser Arbeit wurde mit dem Programm XAMPP⁷ in der Version 1.7.4 ein lokales Testsystem realisiert, welches die oben genannten Anforderungen erfüllt. Dabei ist zu beachten, dass die online verfügbare Anleitung für die Installation mit XAMPP an manchen Stellen veraltet ist. Die dort erwähnten Tools Zip, Unzip und ImageMagick müssen nicht mehr separat heruntergeladen und konfiguriert werden. Sie sind bereits automatisch in der aktuellen Version von ILIAS eingebunden. Außerdem muss bei der Nutzung von XAMPP ein weiterer Schritt beachtet werden. In der Version 1.7.4 muss in einigen Fällen der XSL-Support manuell in der Datei *php.ini* im XAMPP-Unterordner *php* ergänzt werden. Dazu ist zu der Stelle *Dynamic Extensions* innerhalb dieser Datei zu navigieren und die folgende Code-Zeile unter die vorhandenen Einträge einzutragen:

```
extension = php_xsl.dll
```

In derselben Konfigurationsdatei können ebenfalls die im zweiten Schritt der Installationsanleitung erwähnten, weiteren Einstellungen vorgenommen werden. Nach dem Speichern sollten alle Dienste beendet und das Control Panel neugestartet werden, bevor die Änderungen wirksam werden können. Weitere Einstellungen, abweichend von der im Internet verfügbaren Anleitung, müssen nicht vorgenommen werden. Am Ende der Installation wird der Nutzer *root* mit dem Passwort *homer* automatisch angelegt. Nun kann mit dem Konfigurieren und Aufbauen des Systems begonnen werden.

5.2 Update auf eine Neue Version

Wird eine neue Version von ILIAS veröffentlicht, so sind bei der Installation des Updates auf einem bestehenden System folgende Schritte zu beachten. Zunächst sollte auf jeden Fall eine Sicherung der Datenbank vorgenommen werden, um einen Datenverlust im Fehlerfall während des Vorgangs zu vermeiden. Anschließend kann das aktuell installierte ILIAS-Verzeichnis auf dem Server oder lokalen System umbenannt und ein neues Verzeichnis mit dem ursprünglichen Namen erstellt werden. In diesen Ordner werden die Dateien des heruntergeladenen Updates entpackt. Im nächsten Schritt müssen zusätzlich der Ordner *data* und die Datei *ilias.ini.php* aus dem alten Verzeichnis in das neue kopiert werden. Anschließend kann über den Browser die *setup.php* über die Adresse *ilias-installation/setup/setup.php* aufgerufen werden. Nach der Anmeldung werden alle Klienten aufgezeigt und in der Spalte Status befindet sich ein Hyperlink mit der Bezeichnung *Hotfix available*. Mit einem Klick auf diesen sowie einer weiteren Bestätigung, wird die Datenbank aktualisiert und die aus dem alten Verzeichnis kopierten Dateien werden mit der neuen Version verknüpft. Gegebenenfalls muss der Administrator die Systemsprache aktualisieren,

⁶ http://www.ilias.de/docu/goto_docu_pg_6531_367.html

⁷ <http://www.apachefriends.org/de/xampp.html>

falls die Objektbeschreibungen im System nicht richtig angezeigt werden. Nach diesen Schritten steht die neue Version des Systems zur Verfügung.

6 Konzept und Funktionen

Dieser Abschnitt bildet den Hauptteil der Arbeit. Hier wird zunächst auf das grundlegende Konzept von ILIAS eingegangen und anschließend auf ausgewählte Funktionen des Systems. Dabei werden neben den allgemeinen Funktionen, die für alle Nutzer der Plattform gelten, auch diejenigen Funktionen unterschieden, die speziell für einen Nutzer einer entsprechenden Rolle – Administrator, Autor oder Lernender – gelten.

6.1 Konzept

ILIAS nimmt den Gedanken einer umfassenden Bibliothek auf, die Wissen und Materialien zur Verfügung stellt und strebt somit keine vollständige Fixierung auf die Bereiche Lernen und Zusammenarbeiten an [11]. Dieses Konzept wird mit dem sogenannten Magazin, das die Sammlung und Strukturierung des E-Learning Angebots wiederspiegelt, realisiert. Näheres zum Magazin wird im nächsten Abschnitt *Funktionen* erläutert. Dort wird ebenfalls genauer auf den Persönlichen Schreibtisch eingegangen. Dieser bildet für jeden Nutzer den Ausgangspunkt nach der Anmeldung und lässt sich weitestgehend nach den eigenen Wünschen einrichten.

Eine weitere Besonderheit bildet die standardisierte Benutzeroberfläche von ILIAS. Das grundlegende Design des Systems ist für jeden Benutzer gleich, egal ob er Lernender oder Administrator ist. Unter dem ILIAS-Logo in der oberen linken Ecke, folgt eine blaue Navigationsleiste, in der sich die Hauptmenü-Punkte des Systems befinden (siehe Abbildung 2). Dazu gehört zum Beispiel der Verweis zum Magazin und dem Persönlichen Schreibtisch. Darunter existiert eine Tab ähnliche Leiste, die weitere Menü-Punkte bereithält. Je nach Rolle und entsprechenden Berechtigungen sowie Einstellungen durch den Administrator, fehlen oder existieren weitere (rollenspezifische) Menüpunkte in den beiden Leisten. Als Beispiel kann an dieser Stelle der Eintrag „Administrator“ im Hauptmenü genannt werden, der nur für Administratoren sichtbar ist und zu den globalen Einstellungen der Lernplattform führt, die im Weiteren näher erläutert werden.

Mit dem mächtigen Rollensystem bietet ILIAS die Möglichkeit unterschiedliche Zugriffs- und Bearbeitungsrechte zu definieren, so dass Lernende nur Zugriff auf Kurse und deren Inhalte haben, Autoren in der Lage sind Kurse und deren Inhalte zu erstellen sowie abzuändern und Administratoren alle Verwaltungsfunktionen zur Verfügung stehen. Mehr zum Rollensystem befindet sich im Abschnitt *Sicht des Administrators*.

6.2 Funktionen

Viele Funktionen der Lernplattform ILIAS betreffen mehrere Rollen und Nutzer, weswegen an dieser Stelle alle übergreifenden Punkte beschrieben werden sollen. In den Abschnitten zu den einzelnen Sichten – Administrator, Autor und Lernender – werden daher nur ausgewählte, spezielle Funktionen dargestellt.

Eine Besonderheit des LMS ILIAS ist der bereits genannte Persönliche Schreibtisch, der allen Benutzergruppen zur Verfügung steht. Dieser bezeichnet den individuellen Arbeitsbereich eines Nutzers und bildet eine Art Informationsdrehscheibe, da der Nutzer alle relevanten Informationen auf einen Blick erfassen kann. Durch die übersichtliche, dreispaltige Aufteilung, die individuell angepasst werden kann, falls der Administrator diese Funktion aktiviert hat, können Nachrichten aus Kursen, neue Beiträge in Foren, aktive Nutzer, neue Mails, Notizen, der Kalender und angelegte Bookmarks eingesehen und verwaltet werden. ILIAS bietet die besondere Funktion, nahezu sämtliche Objekte auf den Schreibtisch zu legen, um von dort aus einen direkten und zugleich schnelleren Zugriff auf die entsprechenden Objekte zu gewährleisten. Dabei werden in der mittleren Spalte zwei Anzeigemöglichkeiten unterschieden [12]:

- *Meine Angebote* stellt alle Objekte dar, die man auf den Schreibtisch gelegt hat, inklusive der Kurse, in denen man eingeschrieben ist.
- *Meine Mitgliedschaften* zeigt alle Kurse und Gruppen auf, in denen man Mitglied ist bzw. sich eingeschrieben hat.

Ein weiterer wichtiger Punkt ist der einstellbare Detailierungsgrad der einzelnen Bausteine auf dem Persönlichen Schreibtisch. Hier können mindestens zwei Stufen unterschieden werden. Die unterste Stufe stellt lediglich eine Minimalansicht dar, während die höchste Stufe weitere Informationen präsentiert. Auf der ersten Detailstufe des Blocks *Aktive Benutzer* sieht man beispielsweise nur die Gesamtanzahl der aktiven Benutzer am System, während man auf der dritten Stufe alle Nutzer mit vollem Namen und Benutzerbild erkennen kann. Zudem wird ein Hyperlink mit der Beschriftung *Mail* eingeblendet, welcher einen schnellen Zugriff auf die ILIAS interne Mail-Funktion bietet (vgl. Abbildung 2). Diese ermöglicht es, Nachrichten an alle Nutzer der Plattform zu schicken. Dabei können registrierte Nutzer, denen man eine Nachricht zukommen lassen möchte, ähnlich wie in einem Adressbuch, gesucht werden. Eine Weiterleitung der empfangenen Nachrichten an eine externe Email-Adresse kann ebenfalls eingerichtet werden.

Auf den Unterseiten des Persönlichen Schreibtisches kann der Nutzer sein persönliches Profil bearbeiten, also die Daten zur eigenen Person, und zusätzlich kann er auch sein öffentliches Profil, das in ILIAS als Visitenkarte bezeichnet wird, gestalten. Dabei kann explizit ausgewählt werden, welche Daten öffentlich gemacht werden sollen. Außerdem kann der Nutzer die von Kursen erhaltenen Systemnachrichten im Menüpunkt *Nachrichten* betrachten und Filtern. Der Kalender bietet neben der Darstellung der in den verschiedenen Kursen eingetragenen Termine, auch die Möglichkeit persönliche Termine einzupflegen. Ein Export in das iCal-Format wurde mit Version 4 eingeführt [13]. Die Notizfunktion kann verwendet werden, um sich individuelle Notizen auf dem Persönlichen Schreibtisch anzeigen zu

lassen. Ein Beispiel wäre hier, wie in Abbildung 2 dargestellt, die Erinnerung an die rechtzeitige Abgabe von entliehenen Büchern. Die Bookmark-Funktion bietet die Möglichkeit einen Schnellzugriff auf Webseiten und Seiten innerhalb von ILIAS einzurichten. In Abbildung 2 lässt sich erkennen, dass sich der Nutzer einen Link zum Internetauftritt der Universität Bamberg und des Lehrstuhls für Medieninformatik eingerichtet hat.

Abb. 2. Der Persönliche Schreibtisch in ILIAS

Das Magazin bildet beispielsweise die Struktur einer Universität ab und stellt so eine Gliederung von Kategorien dar, deren unterste Ebene Kurse oder Gruppen bilden. Kurse können nur durch Erfüllung von Voraussetzungen betreten werden, welche beispielsweise das Bestehen eines vorherigen Kurses darstellen oder ein Passwort, das man vom Dozenten in der ersten Vorlesung oder Übung eines Kurses ausgehändigt bekommt.



Abb. 3. Magazin mit eingeblendeter Baumansicht auf der linken Seite

Die Navigation innerhalb des Magazins findet über die Positionsleiste unterhalb der blau hinterlegten Menüleiste oder der links, über einen Button einblendbaren Baumansicht, wie in Abbildung 3 dargestellt, statt. In dieser Abbildung lässt sich auch die Strukturierung des Testsystems gut erkennen. Kategorien bilden die Fachrichtungen innerhalb der Informatik, die dann jeweils weiter in Lehrstühle untergliedert werden können. Im Testsystem wurde nur der Bereich der angewandten Informatik der Universität Bamberg teilweise eingepflegt, um die Funktionsweise und Darstellung des Magazins zu verdeutlichen. Innerhalb eines Lehrstuhls, in Abbildung 3 der Lehrstuhl für Medieninformatik, befinden sich Kategorien für die einzelnen Semester und schließlich die jeweiligen Kurse, die in einem Semester angeboten werden. Kategorien lassen sich ebenfalls auf den Persönlichen Schreibtisch ablegen.

6.2.1 Sicht des Administrators

Nach den allgemeinen Funktionen, die für alle Nutzer gelten, soll in diesen Abschnitt auf ausgewählte Funktionen eingegangen werden, die die Rolle des Administrators mit sich bringt.

Dazu gehört vor allem die Verwaltung der registrierten Nutzer. Diese Funktion steht dem Administrator im entsprechenden Administrations-Menü unter der Bezeichnung *Benutzerkonten* zur Verfügung und bietet eine übersichtliche Auflistung aller Nutzer, mit Namen, aktueller Rolle und Email-Adresse. Hier kann der Administrator auch einen neuen Nutzer anlegen, so dass das Registrierungsformular nicht benutzt werden muss, falls es zum Beispiel an dieser Stelle zu Problemen kommen sollte. Nutzer können natürlich auch wieder gelöscht werden, damit sie keinen Zugriff mehr auf das E-Learning System der Universität haben, falls sie diese

beispielsweise verlassen bzw. exmatrikuliert werden. Eine Änderung der Rolle eines Nutzers ist an dieser Stelle auch möglich.

Durch die unterschiedlichen Zugriffsberechtigungen, die bei einer Lernplattform definiert werden müssen, bietet es sich an, ein entsprechendes Modell im Vorfeld auszuarbeiten. Innerhalb von ILIAS können vom Administrator Rollen angelegt werden, wobei an dieser Stelle zwischen globalen und lokalen Rollen unterschieden wird. Eine globale Rolle entspricht beispielsweise der Rolle *Lernender* oder *Autor* und bildet die Zugriffsrechte sowie die zur Verfügung stehenden Funktion für das komplette System ab. Zusätzlich bietet ILIAS sogenannte lokale Rollen, welche zum Teil automatisch generiert werden. Legt ein Autor einen Kurs innerhalb des Magazins an, so wird er zum Besitzer des Kurses und somit gleichzeitig zum Träger der automatisch generierten Rolle Kursadministrator. Das heißt, dass seine aktuelle globale Rolle an dieser Stelle durch die lokale Rolle überschrieben wird. Dadurch kann er erweiterte Rechte erlangen, die ansonsten für seine globale Rolle im restlichen System nicht gelten. Eine Einschränkung der Rechte durch eine lokale Rolle ist natürlich genauso möglich. Die Einstellungsmöglichkeit *Rolle schützen* für globale Rollen verhindert die Einrichtung lokal abweichender Zugriffsregelungen in unteren Hierarchieebenen. Zusätzlich zur Rolle Kursadministrator werden auch die Rollen Kursmitglied und Kurstutor angelegt [12]. Kursmitglied wird ein Lernender automatisch, wenn er dem Kurs beitritt, während er die Rolle Kurstutor nur dann erlangt, wenn er die Rolle von einem Kursadministrator zugewiesen bekommt. Eigene lokale Rollen, wie zum Beispiel die Rolle für eine studentische Hilfskraft, also ein Student der verschiedene Tätigkeiten an einem Lehrstuhl übernimmt, können erstellt und mit den entsprechenden Rechten versehen werden. Letztgenannte Rolle kann nach der Erstellung selbstständig von einem Lehrstuhlinhaber vergeben werden, damit der Administrator sich im alltäglichen Betrieb nicht um solche Angelegenheiten kümmern muss. In diesem Rahmen bietet ILIAS umfassende Möglichkeiten, ein ausgeklügeltes Rechte- und Rollensystem in die eigene Lernplattform zu integrieren.

Die Installation von verschiedenen Sprachen ist bei ILIAS besonders einfach. Da mit dem Download der Software mehr als 20 Sprachen mitgeliefert werden, muss keine Sprachdatei nachträglich aus dem Internet heruntergeladen werden. Jede im System installierte Sprache kann von den Nutzern ausgewählt und vom Administrator als übergreifende Sprache für die komplette Lernplattform eingestellt werden.

ILIAS bietet außerdem die Integration von Erweiterungen, die nicht direkt vom Entwicklerteam erstellt wurden, sondern von Drittanbietern bereitgestellt werden. Eine dieser Erweiterung ist das Projekt *jsMath*⁸, welches nach korrekter Einbindung in das System, mathematische Formeln, beispielsweise in Lernmodulen, nicht mehr als Bilder darstellt, sondern über das TeX-Format als skalierbaren Text. Ebenfalls kann der Dienst *Google maps* in das System integriert werden. Nach erfolgreicher Generierung eines API-Schlüssels auf der Google-Seite⁹, kann die Funktion aktiviert werden. Anstehende Sitzungen und verstreute Kurs-Standorte lassen sich über die eingebundene Karte darstellen, wodurch ein einfaches Auffinden des Standortes

⁸ <http://www.math.union.edu/~dpvc/jsMath>

⁹ <http://www.google.com/apis/maps>

ermöglicht wird. Auch Nutzer können ihren Standort angeben, um ein Treffen außerhalb der Lernplattform einfacher gestalten zu können [12].

Auf dem offiziellen Internetauftritt von ILIAS befindet sich auch eine Anleitung¹⁰, mit der man einen eigenen Skin erstellen kann. Dieser lässt sich einfach in das System einbinden, wodurch die Lernplattform nach dem eigenen Corporate Design angepasst werden kann. Letztlich muss der Administrator den Skin noch im Unterpunkt *System-Styles* des *Skin/Style*-Menüs als System-Standard auswählen, damit die Veränderungen bei allen Nutzern wirksam werden.

6.2.2 Sicht des Autors

Aufbauend auf der Konfiguration durch den Administrator, kann an dieser Stelle mit der Einrichtung des Systems, im Sinne der Erstellung einer Magazinstruktur und Kursen innerhalb der Struktur, begonnen werden. Bei Betrachtung einer Lernplattform an einer Universität wird der Aufbau des Magazins bereits vorgegeben sein und die Rolle des Autors, in der sich Dozenten und Professoren wiederfinden, wird nur mehr die Möglichkeit besitzen, die eigenen Kurse zu gestalten. Hiervon soll in dieser Betrachtung abstrahiert werden. Der Autor in diesem Abschnitt hat zusätzlich zu der Einrichtung von Kursen die Möglichkeit das Magazin zu gestalten. Dazu stehen ihm Container-, Lern- und Pool-Objekte zur Verfügung. Containerobjekte, also Kategorien, Kurse, Verweise zu bereits erstellten Kategorien und Kursen sowie Gruppen, dienen üblicherweise der Strukturierung des Magazins. Innerhalb von Kursen und Gruppen können Lern- und Pool-Objekte genutzt werden. Unter diesen Objekten versteht man unter anderem Diskussionsforen, Tests mit entsprechenden Pools für die Fragen oder einfache Dateien, die vom Lehrenden hochgeladen wurden. ILIAS bietet ebenfalls die Möglichkeit die Kurse über einen Editor individuell anzupassen und somit Texte und Bilder zusätzlich zu den Objekten einzubinden. Statt den Containerobjekten Kategorie und Kurs, existieren zur Strukturierung innerhalb der Kurse Ordner und Sitzungen. Das Grundgerüst eines Kurses bildet im Regelfall das vom Dozenten bereitgestellt Skript in digitaler Form, woran die weiteren Funktionen von ILIAS anknüpfen können. Dazu zählen [12] [14]:

- Diskussionsforen, zur Kontaktaufnahme mit anderen Lernenden und mit Lehrenden in den verschiedenen Kursen.
- Tests, auf deren Basis der Lernerfolg überprüft werden kann. Es werden unter anderem Multiple-Choice-, Lückentext-, Zuordnungs- und Freitext-Fragen unterstützt. Diese Fragen werden in Pool-Objekten abgelegt.
- Übungen, die zu bearbeitende Aufgabenstellung beinhalten, die nach Fertigstellung im System hochgeladen werden müssen.
- Umfragen, die in anonymisierter Form vorliegen und eine Einsicht der Ergebnisse nach Befragungsende zulassen, falls der Autor dies erlaubt. Fragen werden auch hier in Pool-Objekten abgelegt.

¹⁰ http://www.ilias.de/docu/repository.php?ref_id=1785&cmd=sendfile

Neben den aufgelisteten Funktionen, soll an dieser Stelle der Fokus auf das sogenannte ILIAS-Lernmodul gelegt werden. Dieses stellt neben dem HTML- und SCORM-Lernmodul, die wichtigste Funktion in Verbindung mit dem virtuellen Lernen dar. Dabei ist ein Lernmodul im Handbuch *ILIAS – Die Open Source Lernplattform* als „eine Lerneinheit zu einem bestimmten Thema“ definiert [12].

Das ILIAS-Lernmodul kann von einem Autor direkt über die ILIAS-Plattform erstellt werden. Eine weitere Möglichkeit bietet die OpenOffice¹¹-Erweiterung *eLAIX*¹², welche eine Erstellung abseits des LMS und ohne bestehende Internetverbindung bietet. Allgemein besteht ein solches Lernmodul aus verschiedenen Kapitel und den sich darin befindlichen Unterkapitel bzw. Seiten. Je nach ausgewähltem Style, kann sich der Lernende alle Kapitel sowie Seiten direkt in einem zweiten Fenster anzeigen lassen und an die gewünschte Stelle springen oder er muss sich chronologisch durch die komplette Lerneinheit arbeiten, ohne die Möglichkeit in der aktuellen Sicht auf ein Inhaltsverzeichnis zugreifen zu können. Die Navigation erfolgt hier über Schaltflächen am oberen und unteren Ende der Seite. Die vom Autor verfassten Seiten lassen sich durch Hinzufügen von Bildern, Audio- und Videodateien aufbereiten und bieten die Möglichkeit, den Lernenden tiefer in das Thema hineinzuführen bzw. den Stoff der Präsenzlehre zu erweitern und umfassender zu erläutern. Für die einzelnen erstellten Kapitel lassen sich Voreinstellungen definieren, die vom Lernenden im Sinne von Übungsaufgaben oder Tests erfüllt werden müssen, um mit der Bearbeitung des nächsten Kapitels beginnen zu können. Ebenfalls lassen sich für den Lernenden unklare und wichtige Begriffe mit einem vorher angelegten Glossar verknüpfen. Dadurch können Lernende beim Erarbeiten des Inhalts diese Funktion nutzen, um Begriffe besser verstehen zu können. Durch die bereits erwähnten verschiedenen Darstellungsmöglichkeiten, die für jede Seite gewählt werden können, lässt sich das Glossar sogar direkt in die aktuelle Sicht des Lernmoduls miteinbinden, wodurch der Lernende jederzeit einen Überblick über die eingetragenen Fachbegriffe und deren Erläuterung hat [12].

HTML-Lernmodule sind in der Regel bereits existierende Webseiten, die ein Lehrender in Form von miteinander verlinkten HTML-Seiten realisiert hat. Damit der Autor diese nicht in eine andere Form, wie den SCORM-Standard überführen muss, wenn die Lerneinheit in eine Lernplattform integriert werden soll, bietet ILIAS die Möglichkeit dieses über ein Archiv in das System einzufügen. ILIAS ist weiterhin vollständig SCORM kompatibel und bietet seit Version 4 einen SCORM-Editor. Die Abkürzung steht für Sharable Content Object Reference Model und wurde als Empfehlung zur Standardisierung von der Advanced Distributed Learning Initiative (kurz ADL) entwickelt. Ziel ist es die Spezifikationen von verschiedenen E-Learning Technologien zusammenzuführen sowie eine Trennung von Inhalt und Navigation zu schaffen [12] [13].

¹¹ <http://www.openoffice.org/>

¹² <http://www.elalex.org/>

6.2.3 Sicht des Lernenden

Die Sicht des Lernenden umfasst die Nutzung der von Autoren zur Verfügung gestellten Kurse innerhalb des Magazins und deren Inhalte. Lernende haben im Regelfall keinen Zugriff auf die Objekte zur Gestaltung eines Kurses, sondern können nur die bereits erstellten Objekte, die im Abschnitt *Sicht des Autors* aufgelistet sind, nutzen, das heißt mit diesen lernen und arbeiten.

Eine hilfreiche Funktion für den Lernenden ist es, seinen eigenen Lernfortschritt jederzeit mitverfolgen zu können. Der Lernende kann den individuellen Fortschritt in seinen Kursen über den Persönlichen Schreibtisch im Untermenü *Lernfortschritt* betrachten. Dabei wird einerseits der Fortschritt im Kontext des gesamten Kurses, also ob der Lernende den Kurs bestanden hat oder nicht, unterschieden. Andererseits lassen sich auch die Ergebnisse bzw. Noten oder Punkte einzelner abgegebener Übungsblätter und Tests innerhalb eines Kurses nachvollziehen. Ebenso können die bearbeiteten Lernmodule dargestellt werden. Erstellt ein Student beispielsweise eine Übungsleistung, die ihm Bonuspunkte für die Abschlussprüfung eines Kurses bietet, so kann er nach Abgabe und Korrektur dieser seine erreichte Leistung einsehen und bei Fragen seinen Dozenten bzw. Übungsleiter kontaktieren. Ähnliches gilt für elektronische Prüfungen (Tests unter Prüfungsbedingungen), die über die Lernplattform abgewickelt werden [12].

7 Ausblick

Seit 2002 findet jährlich die sogenannte ILIAS Conference an wechselnden Universitätsstandorten statt. Inhalte dieser Konferenz sind im Regelfall Vorträge zu den Themen des E-Learning und der Lernplattform selbst sowie Workshops rund um ILIAS. In diesem Jahr wird die ILIAS Conference am 18. und 19. August an der Universität Bern in der Schweiz veranstaltet. Die voraussichtlich im September erscheinende Version 4.2 von ILIAS und Themen zu didaktischen Perspektiven bilden die Basis der Vorträge. Die Workshops laufen unter dem Thema *ILIAS entdecken* und bieten eine ausführliche Betrachtung des Systems und seiner Funktionen aus den Sichten, die auch in dieser Arbeit dargestellt sind [15]. Weitere Informationen können über den offiziellen Internetauftritt der Veranstaltung in Erfahrung gebracht werden¹³.

Im Regelfall erscheint laut der ILIAS-Roadmap alle zwölf Monate ein *major release*, also eine Hauptversion. Dabei haben die Entwickler begonnen, bei jeder neuen Hauptversion die Zahl hinter dem Punkt um eins zu erhöhen. Erreicht diese den Wert zehn, so wird die Zahl vor dem Punkt um eins erhöht. Als Beispiel lässt sich der Übergang von Version 3 auf Version 4 betrachten: 3.9, 3.10, 4.0, 4.1, 4.2, usw. Weitere Änderungen innerhalb einer Hauptversion werden mit einer weiteren Zahl hinter der zweiten Zahl, getrennt durch einen Punkt, gekennzeichnet, wie es beispielsweise bei der aktuellen Version 4.1.7 zu sehen ist. Bei der letzten Zahl wird

¹³ <http://www.ilias-conference.org/index.php>

auch über die zehn hinaus weitergezählt. Ein Beispiel dafür ist Version 3.10.14. Hauptversionen werden bis zu zwei Jahren unterstützt, wobei aber maximal zwei Hauptversionen gleichzeitig unterstützt werden. So werden also aktuell die Versionen 4.1 und 4.0 durch unregelmäßig erscheinende Aktualisierungen gepflegt. Mit der Veröffentlichung von Version 4.2 im September, fällt Version 4.0 aus dem Updatezyklus heraus, obwohl der Zeitrahmen noch aktiv wäre, aber zu diesem Zeitpunkt bereits zwei aktuelle Hauptversionen vorliegen. Es soll laut Roadmap aber noch eine finale Version 4.0.12 geben, bevor die Unterstützung endet. Für Version 3.10 werden nur noch auf Nachfrage Updates entwickelt, beispielsweise im Falle von Sicherheitslücken oder schwerwiegenden Fehlern. Ältere Versionen werden offiziell nicht mehr unterstützt [16].

8 Literatur

1. CHECK.point elearning: ILIAS 4.0 zeigt neue Schnittstellen auf der LEARNTEC, <http://www.checkpoint-elearning.de/article/7594.html>
2. CHECK.point elearning: ILIAS als sichere Lernplattform von der NATO bestätigt, <http://www.checkpoint-elearning.de/?aID=5877>
3. ILIAS: Known ILIAS Installations and Examples, http://www.ilias.de/docu/goto_docu_pg_8858_470.html
4. Qualitus: Implementierung Ihrer Lern-Management-Lösung, <http://qualitus.de/de/consulting/implementierung-ihrer-lern-management-loesung.html>
5. Baumgartner, P., Häfele, H., Maier-Häfele, K.: E-Learning Praxishandbuch. Auswahl von Lernplattformen. Marktübersicht – Funktionen – Fachbegriffe. Studienverlag, Innsbruck (2002)
6. Schulmeister, R.: Lernplattformen für das virtuelle Lernen. Evaluation und Didaktik. Oldenburg, München (2003)
7. Von Kiedrowski, J.: Unterstützung selbstgesteuerten und kooperativen Lernens – Erfahrungen mit der Nutzung der Plattform ILIAS an der Wirtschafts- und Sozialwissenschaftlichen Fakultät der Universität zu Köln. In: Bett, K., Wedekind, J. (Hrsg.): Lernplattformen in der Praxis. Waxmann, Münster (2003)
8. Von Kiedrowski, J., Kunkel, M.: Gestaltung von Lernplattformen als Open-Source Software am Beispiel der Plattform ILIAS. In: Euler, D., Seufert, S. (Hrsg.): E-Learning in Hochschulen und Bildungszentren. Oldenburg, München (2005)
9. ILIAS open source e-Learning e.V., http://www.ilias.de/docu/goto_docu_cat_1669.html
10. ILIAS: General Requirements, http://www.ilias.de/docu/goto_docu_pg_6531_367.html
11. ILIAS: About ILIAS – ILIAS Broschüre, http://www.ilias.de/docu/repository.php?ref_id=1855&cmd=sendfile&rep_frame=1
12. Henning, J., Dewal, G., Quenzer, M.: ILIAS – Die Open Source Lernplattform. Handbuch für Nutzer, Dozenten und Administratoren. Uni-edition, Berlin (2009)
13. ILIAS: ILIAS 4.0 Features, http://www.ilias.de/docu/goto_docu_pg_27092_35.html
14. ILAIS: Online-Hilfe für Autoren, http://www.ilias.de/docu/goto_docu_pg_35647_1989.html
15. ILIAS Conference: Programme overview, <http://www.ilias-conference.org/index.php/programme-2011-short.html>
16. ILIAS: Roadmap and Releases, http://www.ilias.de/docu/goto_docu_lm_35.html

Shopsysteme speziell das Shopsystem Interchange

David Wagner

Otto-Friedrich-Universität Bamberg, 96045 Bamberg
david.wagner@stud.uni-bamberg.de

1 Zielsetzung

Im ersten Teil dieser Arbeit versuche ich generell auf das Thema Shopsysteme einzugehen. Dies tue ich indem ich einen Überblick über die aktuell beliebtesten auf dem Markt vorhandenen Shopsysteme gebe, Anforderungen die an Shopsysteme gestellt werden können erläutere und auf die grundlegende Architektur von Shopsystemen eingehe. Bisweilen werde ich als Beispiel das Shopsystem Interchange mit einbringen. Im zweiten Teil der Arbeit werde ich das Shopsystem Interchange etwas näher betrachten und werde versuchen dem Leser anhand eines Beispiels die Arbeit abseits vom Administrationsinterface näher zu bringen.

2 Verbreitung verschiedener Shopsysteme

Eine im Juni 2011 durchgeführte Studie, basierend auf der Liste der 1 Millionen bestgelisteten Seiten von Alexa, konnte unter diesen Seiten 25.592 Webshops finden. Aus dem unten gezeigten Schaubild lässt sich feststellen, dass die in diesem Fall vier populärsten Shopsysteme Magento (4705 Seiten), ZenCart (3167 Seiten), VirtueMart (2701 Seiten) und osCommerce (2554 Seiten) sind.[1] Gleichzeitig lässt sich auch erkennen, dass es eine Vielzahl von weiteren Shopsystemen gibt. Übersichtshalber möchte ich vier Kategorien anführen in diese sich Shopsysteme einteilen lassen.

1. Eigenentwicklungen
2. Kauf-Lösungen
3. Miet-Shops
4. Open-Source Lösungen

Jede dieser Shopsystemarten hat im Einsatz ihre Vor-und Nachteile, auf welche ich hier aber nicht näher eingehen werde. Für weitergehendes sei auf das entsprechende Kapitel des E-Commerce Leitfaden von ibi research [2] verwiesen. Im Groteil sind die im Schaubild abzulesenden Shops Kauf oder Opensource-Lösungen, wobei die poplärsten vier alle Open-Source Lösungen sind. Feststellen lässt sich außerdem, dass das Shopsystem Interchange nicht aufgelistet ist. Auss er den auf der offiziellen Website von Interchange gelisteten 309 Shops[3] waren keine weiteren Informationen zu finden.

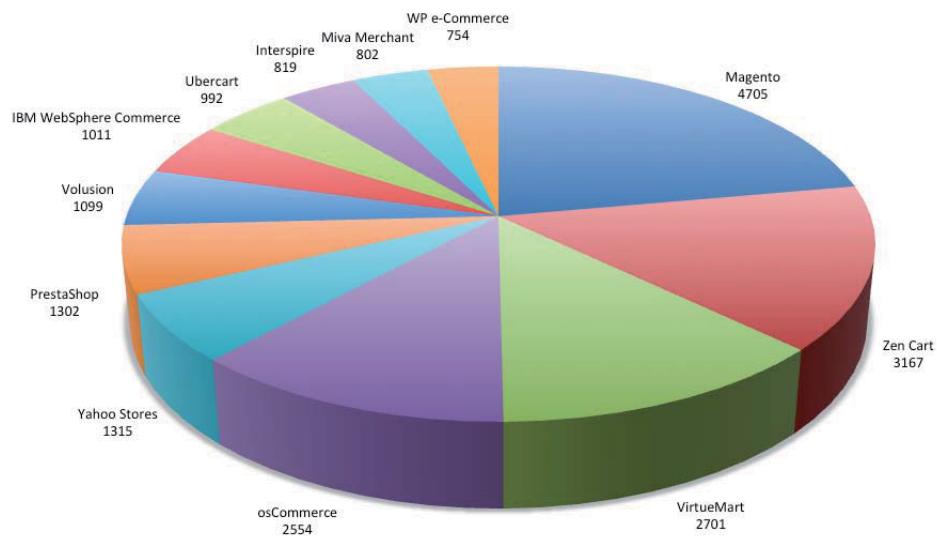


Abbildung 1. Shop system Erhebung - Juni 2011

3 Anforderungen an ein Shopsystem

Die Anforderungen welche an ein Shopsystem gestellt werden können sind vielfältig. So gibt es Funktionen welche ein must-have und andere welche ein nice-to have sind. Dies unterscheidet sich jedoch auch stark je nachdem wie der fertige Shop beschaffen soll. Fur einen kleinen Shop gelten andere Anforderungen als für einen groen Shop. Eine weitere Einteilung wäre die Unterscheidung der Anforderungen nach Kunde und Shopystembetreiber bzw. Administrator. Da jedoch ein zufriedener Kunde auch das Ziel des Betreibers ist und sich somit die Anforderungen beider stark überschneiden, sei hier nur auf der ECommerce Online Lehrgang von der TEIA AG - Internet Akademie und Lehrbuch Verlag verwiesen [4], der bei einer ausführlicheren Behandlung der Anforderungen an ein Shopsystem auch

eine explizite Unterscheidung dieser zwei Personengruppen macht. Ich möchte hier nur kurz die verschiedenen wichtigen Bereiche aufzählen und die von ihnen ausgehenden Anforderungen an ein Shopsystem aufzählen.

Skalierbarkeit

Manchmal ist es nötig mehrere Shops auf einem Server laufen zu lassen. Sind die Ressourcenanforderungen des Shopsystems sehr hoch so ist dies nur begrenzt möglich. Dies ist nicht bei allen Shopsystemen möglich. Interchange hingegen bietet dies Möglichkeit. Liegt das Problem nicht an einer ineffizienten Bandbreitennutzung so kann neben dem Wechsel auf ein anderes Shopsystem auch der Kauf besserer Hardware eine Lösung sein.

Lastverteilung

Shops mit einer hohen täglichen Besucherzahl generieren eine groe Serverlast. Shopysteme müssen in solchen Fällen so geschaffen sein, dass der reibungslose und vor allem schnelle Betrieb weiterhin gewährt ist. Eine Möglichkeit ist dabei beispielsweise das Shopsystem über mehrere Server laufen zu lassen.

Administrierungsinterface

Änderungen, Wartungsarbeiten oder Erweiterungen des Shops müssen über ein für den Betreiber bzw. Administrator zugängliches Interface möglich sein. Dabei ist es auch von Vorteil wenn auf dieses Interface von verschiedenen Standorten zugegriffen werden kann. Wichtig ist ebenfalls, dass dieses Interface nicht unnötig kompliziert aufgebaut ist und so die nötigen Arbeiten erschwert. Natürlich ist dies nicht ganz einfach wenn ein Shopsystem gleichzeitig viel Funktionalität bieten soll. Weitere Funktionen die man von einem Administrierungsinterface verlangen kann, ist die Möglichkeit dieses zwecks effizienterem und komfortablerem Arbeiten an die eigene Bedürfnisse anzupassen. Sinnvoll ist es ebenfalls, dass verschiedene Benutzer eingerichtet werden können. Hinzu kommt dann auch eine Funktion zur gezielten Rechtevergabe. Es ist dabei zu beachten, dass Arbeiten am Shop es nicht verhindern, dass dieser weiterhin für den Kunden bzw. andere Administratoren verfügbar ist.

Statistiken

Mit einigen Shopystemen ist es möglich Informationen über beispielsweise das Nutzungsverhalten der Kunden zu sammeln. Damit dies möglich ist, muss das Shopsystem diese Funktionalität implementiert haben.

Zeitaufwand

Shopysteme unterscheiden sich in ihrer Komplexität und sind damit mehr oder weniger schnell zu erlernen. Der Grad der Komplexität hängt stark mit der Zielgruppe zusammen. So ist beispielsweise der Einsatz eines Shopsystem wie Interchange mit einer sehr flachen Lernkurve für die Erstellung eines kleinen Shops, der sich auch nicht vom Design her stark von anderen Shops abheben muss. Da eine lange Einarbeitungszeit auch Kosten verursacht, beispielsweise

durch Anstellung eines neuen Mitarbeiters, ist dies bei der Entscheidung welches Shopsystem gewählt werden soll zu berücksichtigen.

Kosten

Lange Einarbeitungszeiten und Lizzenzen stellen oft groe Hürden dar. Je nach gewünschtem Ergebnis ist genau darauf zu achten, welches das passende Shop- system ist. Die Unterschiede zwischen den einzelnen Shopsystemen sind teilweise enorm. Oft ist es auch so, dass man sich gegen ein Shopsystem entscheiden sollte, da es wie bei manchen Open-Source Lösungen nur wenig bis gar nicht dokumentiert ist.

Internationalisierung

Für den internationalen Handel ist es notwendig, dass das Shopsystem es ermöglicht verschiedensprachige Varianten des Shops zu erstellen. Zusätzlich zu den Spracheinstellungen müssen Währungs-, Steuer und Versandeinstellungen individuell einstellbar sein.

Sicherheit

Da beim Verkaufen über einen Webshop sensible Daten wie beispielsweise Kreditkarteninformationen, Adressinformationen und Zugangsdaten des Kunden gespeichert werden müssen, ist es wichtig, dass das Shopsystem nicht unnötige Sicherheitslücken enthält. Dies betrifft einerseits die Verbindung zwischen Webshop und Kunden, andererseits auch spezielle Teile eines Shopsystems wie die Datenbank oder den Shopsystemserver selbst. Eigenentwickelte Shoplösungen sind oft noch nicht so ausgereift wie Lösungen aus dem Kauf-oder Opensourcebereich, welche schon mehrere Jahre existieren. Sicherheit betrifft aber auch die Ausfallsicherheit des Shopsystems. So kann eine integrierte Backuplösung viel Ärger ersparen.

Warenangebot

Wichtig bei einem Shopsystem ist auch, wie gro das eingepflegte Warenangebot werden darf. Shopsysteme bei denen nur eine begrenzte Zahl an Waren eingepflegt werden können sind für die Erstellung groer Shops ungeeignet. Soll das Warenangebot im nachhinein verändert werden, so muss dies auch als eine Standardfunktionalität eines Shopsystems relativ einfach funktionieren. Zu beachten ist ebenfalls, ob das Shopsystem es ermöglicht den selben Artikel in verschiedenen Aufführungen und Mengen anzubieten. Dies ist nicht immer ganz trivial, soll eine andere Ausführung eines Artikels auch eine eigene Artikel-und Bestellnummer erhalten. Da der Verkauf von digitalen Medien immer wichtiger wird ist diese Frage auch zu beachten.

Präsentation

Da der Shop die Schnittstelle zwischen Kunde und Verkäufer ist, sollte diese vom Kunden auch möglichst ansprechend empfunden werden. Mit vorgefertigten Templates kann dem Shop meist relativ einfach ein adäquates Design verliehen

werden. Andererseits kann es aber auch gewünscht sein ein komplett eigenes Design zu erstellen. Anpassbare Templates sind dabei eine groe Hilfe, jedoch nicht in jedem Shopsystem vorhanden. Manchmal kommt es auch vor, dass Design und Anwendungslogik im Shopsystem nicht getrennt werden, was meist das Arbeiten erschwert. Man sollte zudem darauf achten welche Möglichkeiten die Artikel zu päsentieren ein Shopsystem mitbringt. Dies geht vom einfachen Bild in verschiedenen Gröen über Videos bis hin zu 3D Darstellungen.

Navigation und Suche

Ein Kunde sollte sich in dem Shop relativ leicht zurecht finden. So stellt sich die Frage, wie flexibel ist das Shopsystem beim Erstellen der gewünschten Navigationsstruktur. Groe Shops benötigen oft eine individuelle Anpassung selbiger. Auch die Klassifizierung der Ware unter Kategorien ist nicht bei jedem Shopsystem einheitlich. Es ist in einigen Fällen jedoch wünschenswert den selben Artikel unter mehreren Kategorien einzupflegen um so dem Kunden eine effizientere Suche zu ermöglichen. Einstellungsmöglichkeiten bei der Suche im Warenbestand sind ebenfalls ein Pluspunkt. Interessant ist dabei inwiefern sich die dem Kunden zur Verfügung gestellte Suchfunktionalität vom Administrator anpassen lässt.

Suchmaschinenoptimierung

Ein groer Teil der Kunden findet den Weg auf den Webshop über eine Suchmaschine. So ist eine Anforderung an ein Shopsystem die Möglichkeit der Suchmaschinenoptimierung, damit die Artikel besser in Suchmaschinen gefunden werden können.

Schnittstellen

Oft muss ein Shopsystem ausser mit einer Datenbank noch mit weiteren Komponenten wie mit Bestandteilen eines Warenwirtschaftssystems harmonieren. Vorhandene und selbstprogrammierbare Schnittstellen ermöglichen einen Datenaustausch und machen so eine Einbindung möglich. Aber auch im Falle der Datenbank ist es wichtig, ob Import und Exportfunktionen vom Shopsystem bereitgestellt werden oder ob das Shopsystem mit einer bereits vorhandenen speziellen Datenbank funktionsfähig ist.

Warenkorb

Einige Shopsysteme bieten die Möglichkeit die in einem Warenkorb abgelegten Artikel auf dem Server zu speichern, so dass es für den Kunden möglich ist zu einem späteren Zeitpunkt den Einkauf fortzusetzen. Interchange bietet auerdem die Möglichkeit, dass in einem Shop von einem Kunde mehrere Warenkörbe eingesetzt werden können oder vom Kunden zusammengestellte Warenkörbe als Vorlagen abzuspeichern.

Marketingmaßnahmen

Sehr interessant sind die verschiedenen Marketingmöglichkeiten. Darunter fallen z.B. Geschenk- oder Rabattgutscheine, Verlosungen oder Bannerwerbung. Neben

der klassischen Newsletterwerbung sind auch Recommendersysteme möglich, die dem Kunden auf ihn abgestimmte Artikelempfehlungen zeigen.

Support

Um den Kunden besser zu binden, ist es mit einigen Shopystemen möglich neben dem üblichen E-mailsupport auch Chat- und Videosupport anzubieten.

Feedback und Meinungsaustausch

Hat der Kunde sein Produkt erhalten so möchte er eventuell die Möglichkeit haben das Produkt zu bewerten oder sich in einem Forum auszutauschen. Interchange bietet dem Kunden sogar die Möglichkeit Videos hochzuladen.

Zahlung und Versand

Einige Käufe kommen nicht zustande, weil es dem Käufer nicht möglich ist die angebotenen Zahlungsweisen wahrzunehmen oder er nicht dazu bereit ist. Je nach Shopsystem gibt es eine unterschiedliche Anzahl an Modulen für verschiedene Zahlungsformen. Hinzu kommen verschiedene Module für den Versand.

4 Architektur

Die Shopsysteme unterscheiden sich nicht nur in ihrer Funktionsvielfalt oder den verursachenden Kosten, sondern auch in der Architektur. Nichtsdestotrotz kann man sagen, dass die meisten Shopsysteme vom Prinzip her gleich aufgebaut sind. Generell ist es so, dass die Architektur einem 3-Schichten Modell folgt:

4.1 Präsentationsebene

Diese Ebene ist repräsentiert das Frontend, welches als das Gegenstück zum Backend zu sehen ist. Das Backend ist normalerweise das Datenbankmanagement und residiert in einer anderen Schicht. Die Präsentationsebene enthält das Shop Frontend und überdies die Komponente zur Administrierung.

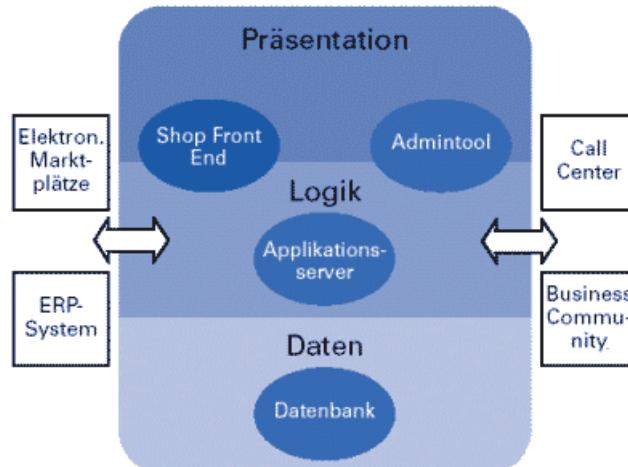


Abbildung 2. Schichtenmodell [5]

Shop-Frontend

Diese Komponente bietet stellt die Schnittstelle für den Kunden zum Shop. Diese enthält alle Basis- und Zusatzfunktionalitäten[5]:

1. Information für den Kunden
2. Darstellung des Produktkatalogs
3. Zahlungsfunktionen und Zahlungsoptionen für den Kunden,
4. Auftragsverfolgung sowie
5. Personalisierung von Vorgängen und Angeboten.

Administrationskomponente

Die Administrationskomponente bietet Funktionalitäten zur Verwaltung der Kunden, Auftrags- und Produktdaten und liefert regelmäßig Möglichkeiten, das Design der Darstellung der Produkte anzupassen. Zur Administration gehören auch weitergehende Einstellungen, wie verschiedene Sprachen, Währungen, Steuerklassen und Wechselkurse des Shopsystems. [5]

4.2 Die Logikebene

Auf der Logikebene lässt sich eine zentrale Komponente ausmachen, der Applikationsserver. Die Geschäftsprozesse des Shopsystems werden in den Prozessen des Applikationsservers abgebildet. Somit wird über die Prozesse der gesamte Datenfluss zwischen Kunde und Shopbetreiber dargestellt. Außerdem auf dem Schaubild zu sehen sind externe Komponenten bzw. Systeme die über diese Ebene an das Shopsystem angebunden werden. Diese können z.B. ERP-Systeme, Call Center oder elektronische Marktplätze sein. [5]

4.3 Datenbankebene

Nach dem Schaubild ist die Komponente dieser Ebene die physikalische Datenbank. Erfasst werden hier die Daten des Shopsystems, wobei zwischen statischen (Stammdaten) und dynamischen (Bewegungsdaten) Daten unterschieden wird. Erstere wären Daten, welche nur geringfügigen Veränderungen unterworfen sind, beispielsweise Produkt- oder Kundenstammdaten. Erst zur Laufzeit des Shopsystems erstellte Daten, wie Auftragsdaten, sind dann die dynamischen Daten. [5]

5 Shopsystem Interchange

Interchange ist ein Open-Source Shopsystem, dass aus den zwei E-Commerce Lösungen MiniVend und Tallyman entwickelt und zeitweise von RedHat gefördert wurde. Es ist in Perl geschrieben und bietet eine Vielfalt an Funktionen. Neben den grundlegenden werden fast alle der optionalen Anforderungen an ein Shop- system werden erfüllt. Problematisch je nach Einsatzzweck kann jedoch der Funktionsumfang werden, da dadurch die Lernkurve beim Erlernen bzw. Arbeiten mit Interchange sehr flach ist. Verstärkt wird dies durch eine überarbeitungs- und ergänzungswürdige Dokumentation. So ist es deshalb nur sehr begrenzt für kleinere Projekte geeignet. Dafür lässt es sich jedoch mittels Perlmodulen fast beliebig erweitern und kann flexibel in verschiedensten Umgebungen eingesetzt werden.

5.1 Voraussetzungen

Interchange ist lauffähig auf einer Vielzahl von Betriebssystemen. Verschiedene Linuxdistributionen, Mac OS X oder Windows(mit Cygwin) sind nur ein paar der Möglichkeiten. Flexibilität besteht auch bei der Wahl des Webservers. Empfohlen wird jedoch der Webserver von Apache. Sogar eine Datenbank ist nur optional, wobei standardmäßig eine SQL Datenbank einzusetzen ist. Eine Besonderheit bei Interchange ist, dass auch ein C Compiler benötigt wird. Warum das so ist, werde ich noch erklären. Und natürlich ist für Interchange Perl notwendig.

5.2 Architektur von Interchange

Anhand des Schaubilds sind die notwendigen Komponenten für den Einsatz von Interchange ersichtlich. Selbsterklärend sind der WebClient und die Datenbank. Interessant ist das link Programm zwischen Webserver und dem Interchange- server. Dieses kleine C Programm, welches als CGI Skript installiert wurde dient dem Parameteraustausch zwischen den beiden Komponenten. Da dieses Programm kompiliert werden muss, ist dies somit der Grund, weshalb ein C Compiler benötigt wird. Die Kommunikation erfolgt dann über Sockets, wobei der Interchangeserver als Dienst im Hintergrund läuft und nur mittels dieses Skripts über Sockets erreichbar ist. Dies tut der Sicherheit sehr zu gute und der Interchangeserver ist vor unbefugtem Zugriff aus dem Internet besser geschützt.

Kataloge. Auffällig ist, dass bei Interchange eine Trennung von Katalog und Server vorzufinden ist. Kataloge repräsentieren die einzelnen Shops, und sind somit was die Websites für den Webserver sind. Bei Interchange ist es möglich mehrere Kataloge, also Shops, auf einem einzigen Interchangerserver zu betreiben. Laut Entwickleraussagen ist die Skalierbarkeit enorm gut. Grundbedingung damit Interchange einsetzbar ist, ist mindestens ein Katalog. Es existiert ein Skript mit der Bezeichnung makecat, mithilfe dessen man sehr schnell einen Shop erstellen kann, welcher dem Demoshop auf der Entwicklerseite von Interchange entspricht. Kataloge werden auf Ordnerebene jeweils durch ein eigenes Verzeichnis, dem Catalog Root Directory kurz CATROOT, repräsentiert welches die jeweils katalogrelevanten Dateien enthält. Jedes dieser Verzeichnisse enthält außerdem eine eigene Konfigurationsdatei, die catalog.cfg. Dort können um den Shop individuell zu konfigurieren, Direktiven gesetzt werden.

Konfigurationsdateien. Wie gerade schon erwähnt besitzt jeder Katalog eine für ihn spezifische Konfigurationsdatei. Diese dient dazu den Katalog unabhängig von anderen Katalogen zu konfigurieren. Es ist jedoch auch möglich, dass Kataloge Datenbanken, Seiten und Sessionfiles teilen, so dass man theoretisch auch virtuelle Malls erzeugen könnte. Der minimale Inhalt einer solchen katalogsspezifischen Datei ist am folgenden Ausschnitt einer Konfigurationsdatei für einen Katalog mit dem Namen tutorial zu sehen:

```
VendURL      http://myhost.mydomain.local/cgi-bin/ic/tutorial
SecureURL    http://myhost.mydomain.local/cgi-bin/ic/tutorial
MailOrderTo   myname@mydomain.local

ErrorFile    /var/log/interchange/tutorial-error.log

Database     products products.txt TAB
ProductFiles products
```

Abbildung 3. catalog.cfg

++ Die VendURL repräsentiert die URL des Katalogs, falls HTTPS gewünscht ist, so kann diese bei SecureURL angegeben werden. Eine E-Mail Adresse ist notwendig und die Angabe wo die Datenbank zu finden ist. Besonderheit ist, dass in diesem Beispiel keine SQL Datenbank, sondern nur eine Textdatei eingesetzt wird. Dies ist zwar möglich, wird aber nicht empfohlen. ProductFiles muss auch angegeben werden, damit, angenommen es wird eine MySQL Datenbank eingesetzt, Interchange weiß welche Tabelle die Artikeldaten enthält. Auf die Syntax gehe ich im Rahmen der Arbeit nicht weiter ein. Dazu sei auf die auf der Interchange Entwicklerseite vorhandene Dokumentation verwiesen.

Neben den katalogsspezifischen Konfigurationsdateien existiert noch ein anderer Typ. Hat man Interchange erfolgreich installiert, so findet sich im ICROOT Verzeichnis eine Konfigurationsdatei namens interchange.cfg. In ihr werden Direktiven gesetzt, welche globale Gültigkeit besitzen. Außerdem müssen in dieser Datei alle Kataloge eingebunden werden. Im folgenden ist eine minimale Version einer solchen Datei zu sehen:

```
Catalog tutorial /var/lib/interchange/catalogs/tutorial /cgi-bin/ic/tutorial
DebugFile /var/log/interchange/debug.log
ErrorFile /var/log/interchange/error.log
```

Abbildung 4. interchange.cfg

Die erste Zeile dieser Konfigurationsdatei zeigt, wie mit der Direktive Catalog, der Katalog tutorial eingebunden wird. Die beiden weiteren Zeilen spezifizieren den Ort von Debug- und Fehlerdatei.

Zu diesem Abschnitt über Katalogdateien ist noch hinzuzufügen, dass das eben Gezeigte nur die absolute Minimalkonfiguration für einen lauffähigen Interchange-server ist. In Konfigurationsdateien können außerdem noch Perl Module registriert werden oder auch selbst geschriebene Tags. Was die von Interchange mitgebrachten Tags sind und wie sie zu nutzen sind, das werde ich gleich noch erklären. Es gibt eine Vielzahl von verschiedenen Einstellungen welche in diesen Dateien mittels Direktiven gesetzt werden können. Eine Auflistung all dieser Direktiven befindet sich auf der Entwicklerseite von Interchange.

Perl und Interchange Tag Language. Interchange wurde wie bereits erwähnt in Perl geschrieben und es ist auch möglich Interchange mittels in Perl geschriebenen Modulen zu erweitern. interessanterweise sind für die Nutzung von Interchange nicht unbedingt Perl-Kenntnisse notwendig. Dem Shopsystembenutzer steht eine eigene Tag-Sprache, die sogenannte Interchange Tag Language, zur Verfügung. Diese Sprache (eingeschlossen in [tag] Klammern) ähnelt stark denen von HTML genutzten <tag>s. Hauptunterschied ist jedoch, dass die ITL Tags auf Serverseite vom Interchange daemon geparsed werden. Diese werden daraufhin in plain text++ und HTML umgewandelt und anschliessend über den Webserver zum End-Benutzer geschickt, woraufhin sie noch einmal für die finale Darstellung geparsed werden. Beispiele für grundlegende Tags:

1. loop (iteriert über die ihm mitgegebenen Argumente; z.B. kann so Suchfunktionalität implementiert werden)
2. page (erzeugt einen hypertext link)
3. form (für die Erstellung von Formularen)
4. if (Konditional)
5. query (SQL Befehle ausführen)
6. item-list (iteriert über Elemente aus dem Warenkorb)

Zu beachten ist, dass Tags immer in eckigen Klammern stehen. Viele müssen wie manche Tags in HTML wieder geschlossen werden. Es ist bei vielen Tags möglich bzw. notwendig ihnen Argumente mitzugeben. Tags können zudem auch mit Perl selbst erstellt werden, müssen dann aber in den Konfigurationsdateien registriert werden. Zu Perl ist noch zu sagen, dass neben den eigens oder von anderer Seite erstellten Perlmodulen und ausser über die Erstellung von Tags, Perl auch direkt in den Dateien, über den Tag [perl] [/perl] eingesetzt werden kann. Diese Mächtigkeit und Flexibilität ist einerseits ein Vorteil, andererseits ist viel Einarbeitungszeit zu investieren und es ist nicht immer leicht, wenn Perl sich mit Tags mischt, den Überblick zu behalten.

5.3 Exkurs: Erstellung einer einfachen Startseite mit Interchange

Um kurz die Funktionsweise von Interchange und seiner Tags zu veranschaulichen, zeige ich kurz wie man das Gerüst einer minimalen Startseite von Interchange erstellt. Im folgenden der klassische Aufbau einer Seite:

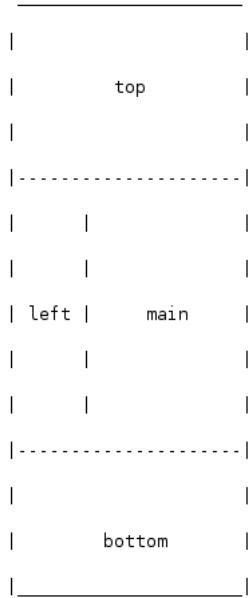


Abbildung 5. Seitenaufbau

Erkennbar ist eine Vierteilung. Der obere Bereich enthält normalerweise Banner, Menüs usw.. Der linke Bereich kann als Sidebar oder Navigationsleiste genutzt werden. Im unteren Bereich können beispielsweise die Kontakt- oder Copyrightinformationen enthalten sein. Bewegt sich der Kunde durch den Shop, so unterliegen diese drei Bereiche normalerweise keinen groen Veränderung. Im Normalfall ist es gewünscht, dass man für diese Bereiche ein Template erstellen kann und sollte dieses geändert werden, dass diese Änderungen alle Seiten des Shops betrifft. Der sich ändernde Bereich ist main. In diesem wird der seiten-spezifische Inhalt dargestellt.

Die Erstellung der Templates erfolgt indem man im CATROOT Verzeichnis des Katalogs für jeden der konstant bleibenden Bereiche eine Datei, vorallem mit einem aussagekräftigen Namen, erstellt. Diese Dateien werden mit HTML Code gefüllt und zwar so, als würden die drei Dateien zusammengefügt eine komplette HTML Datei ergeben. Wichtig dabei ist, dass man beim Abspeichern der Datei keine Endung wie beispielsweise .htm angibt. Hier zur Veranschaulichung der Inhalt der drei erstellten Dateien:

```
<html>
<head>
<title>The Interchange Test Catalog</title>
</head>

<body>
<div align=center>
<table width="80%" border cellpadding="15">
<tr><td colspan="2" align="center">
<h1> The Interchange Test Catalog </h1>
</td></tr>
```

Abbildung 6. CATROOT/top

```
<tr>
<td align="center">
(left)
</td>
<td align="center">
```

Abbildung 7. CATROOT/left

```

</td>
</tr>
<tr><td colspan="2" align="center">
    (bottom)
</td>
</tr>
</table>
</div>
</body>
</html>

```

Abbildung 8. CATROOT/bottom

Anschliessend muss im Ordner in denen sich später die HTML Seiten befinden, in diesem Beispiel der Ordner pages, eine HTML Datei erstellt werden. Diese Datei bekommt die Endung .htm. Der Inhalt der Datei sieht folgendermaen aus:

```

[include top]
[include left]

This is where your content goes.

[include bottom]

```

Abbildung 9. Die fertige Seite

Hier sieht man, dass der include-Tag zum Einsatz kommt. Die Funktion dieses Tags, ist eine Datei in die aktuelle Seite einzubinden und in der einzubindenden Datei sich befindliche Tags parst. In dieser Datei kann nun Content eingefügt werden. Wenn der Interchangeserver und die anderen Komponenten korrekt laufen, ist es möglich diese Seite über den Browser zu erreichen. Wenn die Seite auch in der Konfigurationsseite als Startseite eingetragen worden ist, so ist es nicht mehr unbedingt nötig die genaue URL einzugeben.

Literatur

1. *June 2011 eCommerce Survey.* <http://tomrobertshaw.net/2011/06/june-2011-e-commerce-survey/>.
2. *Der E-Commerce-Leitfaden.* http://www.ecommerce-leitfaden.de/im-internet-verkaufen.html#anchor_2_3_3.
3. *icdevgroup - Hall of Fame.* http://www.icdevgroup.org/i/dev/hall?id=8aRVnsJj&mv_pc=13.
4. *Konzepte in eCommerce-Anwendungen.* <http://www.teialehrbuch.de/Kostenlose-Kurse/eCommerce/>.
5. *Das technische Konzept eines E Shops nach KPMG Consulting.* <http://www.teialehrbuch.de/Kostenlose-Kurse/eCommerce/5801-Das-technische-Konzept-eines-E-Shops-nach-KPMG-Consulting.html>.

CakePHP

Eine Einführung in das Rapid Development Framework

Sebastian Boosz

`sebastian-wilhelm.boosz@stud.uni-bamberg.de,`
Otto-Friedrich Universität Bamberg

Zusammenfassung Im Folgenden soll ein Überblick über die umfangreichen Möglichkeiten, die das Webframework CakePHP in der Version 1.3 bietet, gegeben werden. Anhand eines kleinen Beispielprojekts werden die grundlegenden Elemente jeder auf CakePHP basierenden Webanwendung im Detail erläutert. Zusätzlich werden weiterführende Themen aufgegriffen und exemplarisch dargestellt. Diese Arbeit soll den Leser in die Lage versetzen, die Struktur und Funktionsweise von CakePHP nachzuvollziehen und so als ein Einstiegspunkt in die Arbeit mit diesem Framework dienen. Um den Rahmen der Arbeit nicht zu sprengen, werden auf Grundlagen von Webframeworks nicht gesondert eingegangen. Es handelt sich also keinesfalls um eine Einführung in Webframeworks im Allgemeinen.

1 Charakteristika

CakePHP ist ein sogenanntes *Full Stack* Webframework, d.h. alle gängigen Aufgaben, die bei der Entwicklung von Webanwendungen auftreten, werden bereits durch das Framework unterstützt. Einerseits wird der Entwickler dadurch zwar bei der Strukturierung und Implementierung seiner Anwendung durch das Framework entlastet, andererseits schreibt das Framework aber auch vor, auf welche Weise bestimmte Aufgaben zu erledigen sind und schränkt damit die Freiheit des Entwicklers ein. Zusätzlich beansprucht CakePHP auch ein *Rapid Development Framework* zu sein. Die Arbeit mit CakePHP soll schnell zu sichtbaren Ergebnissen führen und sich damit auch für die Erzeugung von Prototypen eignen. Kenntnisse in PHP, SQL und den Konzepten der objektorientierten Programmierung sind für die Arbeit mit CakePHP Voraussetzung. Einigen Anwendern dürfte die von den Machern von CakePHP konsequent eingesetzte Kuchenmetapher als unseriös erscheinen, so wird das Erstellen einer Webanwendung mit CakePHP als backen und die offizielle Dokumentation häufig als das Kochbuch bezeichnet. Webentwicklern die sich bereits mit *Ruby on Rails* beschäftigt haben, werden feststellen, dass CakePHP viele Aspekte von *Rails* sehr genau imitiert.

CakePHP bietet eine Reihe von Features, die in der Welt der Webframeworks keinesfalls einzigartig sind, aber dennoch genannt werden müssen. *Application Scaffolding* ermöglicht das sehr schnelle Erzeugen von Webanwendungen, die rudimentäre Funktionen zur Datenbearbeitung bieten, auf diese Technik wird in

Abschnitt 3 zurückgegriffen. *Convention over configuration* ist in CakePHP sehr stark ausgeprägt, die Namenskonventionen müssen nicht zwingend eingehalten werden, ersparen aber bei deren Einhaltung sehr viel Zeit. CakePHP realisiert ebenso die von anderen Webframeworks und vor allem von Desktopanwendungen bekannte *Model View Controller* Architektur, das heißt die strikte Trennung von Datenhaltung, Anwendungssteuerung und der Präsentation der Daten beim Nutzer. Verschiedene Konsolenanwendungen, die in Abschnitt 6.3 näher erläutert werden, erleichtern das Erzeugen und Erweitern von Anwendungen. Die automatische Validierung von Nutzereingaben, sowie ein einfaches Templatesystem, welches ohne eine eigene Templatesprache auskommt, und demnach nur PHP und HTML verwendet, können helfen Zeit zu sparen. Weitere Bestandteile von CakePHP werden im Verlauf der Arbeit näher betrachtet.

2 Installation und Konfiguration

In diesem Abschnitt werden die nötigen Voraussetzungen und Konfigurations schritte behandelt, die erfüllt sein müssen, um CakePHP einsetzen zu können.

2.1 Voraussetzungen

Die Einrichtung eines CakePHP Systems erfordert nur sehr wenig Aufwand. CakePHP wird damit auch schon vor der eigentlichen Anwendungsentwicklung dem Anspruch eines *Rapid Development Frameworks* gerecht. Die neueste CakePHP Version kann kostenlos auf der Downloadseite¹ heruntergeladen werden. Eine große Spannbreite verschiedener Systemkonfigurationen erlaubt den Betrieb des Frameworks. Sowohl der weitverbreite *Apache* Server, als auch *lighttpd* und *Microsoft IIS* werden unterstützt. Nach Möglichkeit sollte dabei das Modul *mod_rewrite* aktiviert sein. Obwohl CakePHP zu PHP 4 kompatibel ist, ist es vor allem im Hinblick auf die zukünftigen Entwicklungen ratsam, bereits jetzt PHP 5 einzusetzen. Auch bei der Datenhaltung wird dem Anwender großer Freiraum gewährt, folgende Datenbanksysteme werden unterstützt:

- MySQL ab Version 4
- PostgreSQL
- Firebird DB2
- Microsoft SQL Server
- Oracle
- SQLite
- ODBC
- ADOdb

¹ <https://github.com/cakephp/cakephp/downloads>, letzter Zugriff am 31. Juli 2011

2.2 Die Verzeichnisstruktur

Zunächst muss die heruntergeladene .zip-Datei auf den Webserver übertragen und dort im *htdocs* Verzeichnis direkt oder in einem Unterordner entpackt werden. Dadurch werden u.a. vier Verzeichnisse angelegt. Der *app/* Ordner stellt die Webanwendung selbst dar. Hier befinden sich alle PHP-Dateien und weiteren Ressourcen, die vom Entwickler angelegt wurden. Das *cake/* Verzeichnis beinhaltet die Kerneldateien des Frameworks und sollte im Normalfall unverändert belassen werden. *Plugins/* und *vendors/* werden genutzt um fremden Code in die eigene Applikation einzubinden. *Plugins* sind dabei eigene CakePHP Applikationen, die gewisse Funktionalität anbieten, denkbar ist beispielsweise ein Plugin um ein Forensystem in eine bestehende Cake Anwendung einzubinden. In *Vendors/* werden schließlich alle zusätzlichen PHP Bibliotheken abgelegt.

Diese klare Aufteilung hat den Vorteil, das benötigte Plugins oder Bibliotheken problemlos und konsistent eingebunden werden können. Auffallend ist, dass auch der *app/* Ordner gleichnamige Unterordner für Plugins und Bibliotheken besitzt. Da der *cake/* Ordner konstant bleibt, ist es möglich mehrere Anwendungen mit einer CakePHP Version zu betreiben. Je nachdem ob bestimmte Plugins und Bibliotheken also für mehrere Anwendungen oder beschränkt auf eine Anwendung nutzbar gemacht werden sollen, müssen die Dateien also in den entsprechenden Ordner im *root* Verzeichnis (verfügbar für alle Anwendungen), oder in die Unterordner des *app/* Verzeichnisses abgelegt werden.

Die weiteren Unterverzeichnisse des *app/* Ordners werden im Folgenden jeweils bei Bedarf kurz erläutert. Um mit der eigentlichen Anwendungsentwicklung anfangen zu können, müssen nun noch 2 Konfigurationsschritte ausgeführt werden.

2.3 Konfiguration

Dazu muss zunächst der Pfad *app/config/* aufgerufen werden. Die Datei *database.php.default* muss in *database.php* umbenannt werden. In dieser Datei werden die Zugangsdaten zur Datenbank eingetragen, im Beispiel wird von einer MySQL Datenbank auf einem lokalen Webserver ausgegangen:

```
var $default = array(
    'driver' => 'mysql',
    'persistent' => false,
    'host' => 'localhost',
    'login' => 'cakeUser',
    'password' => 'cakePassword',
    'database' => 'cake_test',
    'prefix' => '',
);
```

Bei Bedarf können zusätzlich unter der Variable *\$test* die Zugangsdaten für eine Testdatenbank definiert werden. In der Datei *app/config/core.php* müssen zuletzt noch die Werte *Security.salt* (Z. 204) und *Security.cipherSeed* (Z. 209) geändert werden, die zur Verschlüsselung eingesetzt werden:

```
Configure::write('Security.salt', 'ogietui3g6634g');
[...]
Configure::write('Security.cipherSeed', '21543215366747356');
```

Während für die *Security.salt* eine beliebige Kombination von Buchstaben und Zahlen gültig ist, dürfen für *Security.cipherSeed* nur Ziffern eingesetzt werden. Während des Konfigurationsprozesses kann jeder Zeit die *Root URL*² von CakePHP aufgerufen werden, um zu überprüfen ob alle Einstellungen korrekt vorgenommen wurden und ob die Datenbank erreichbar ist. Sobald alle Felder in dieser Übersicht grün hinterlegt sind, kann die Anwendungsentwicklung starten.

3 Scaffolding

In diesem Abschnitt soll eine einfache aber voll funktionsfähige Webanwendung mit Hilfe von CakePHPs Scaffolding Mechanismus umgesetzt werden. Das Beispiel ist aus [1] entnommen und zieht sich durch das ganze Buch: Das Ziel ist die Implementierung einer einfachen Jobbörse. Jobangebote sollen angelegt, bearbeitet und gelöscht werden können. Zudem ist eine Detailansicht einzelner Angebote und auch eine Übersicht über alle vorhandenen Jobangebote erwünscht.

3.1 Die Datenbank

Um die Jobangebote persistent speichern zu können, muss zunächst eine passende Datenbankstruktur geschaffen werden. Anders als bei Ruby on Rails können Datenbanktabellen nicht durch das Framework selbst angelegt werden. Stattdessen muss ein Datenbankadministrationsprogramm (z.B. *phpMyAdmin* für MySQL Datenbanken) verwendet werden. Um den oben beschriebenen Sachverhalt abzubilden, genügt eine Datenbanktabelle, folgender SQL Befehl erzeugt die passende Struktur:

```
CREATE TABLE IF NOT EXISTS 'jobs' (
    'id' int(11) NOT NULL auto_increment,
    'company' varchar(128) NOT NULL,
    'title' varchar(128) NOT NULL,
    'description' text NOT NULL,
    'created' datetime NOT NULL,
    'modified' datetime NOT NULL,
    PRIMARY KEY ('id')
);
```

Schon bei der Erstellung der Datenbank spielen Namenskonventionen eine Rolle: Tabellennamen sollten im Plural und mit englischen Namen bezeichnet werden, da sonst später die korrekte Zuordnung zwischen Tabelle und Model gestört

² Beispielsweise <http://localhost/cake>, wenn CakePHP in den Ordner *htdocs/cake/* entpackt wurde

werden könnte. Beispielsweise kann CakePHP im Englischen auch auf irregulär gebildete Pluralformen schließen (*people* als Plural zu *person*). Für andere Sprachen ist ein solcher Mechanismus nicht vorgesehen.

Des Weiteren geht CakePHP davon aus, dass die Spalte mit dem Primärschlüssel immer *id* genannt wird. Die Einhaltung dieser Konvention ist vor allem für die Erzeugung von Formularen für das Editieren von Datensätzen hilfreich. Neben der Verwendung von normalen *Ids*, welche innerhalb einer Tabelle einzigartig sind, werden auch sogenannte *Universally unique identifiers (UUIDs)*, welche in der gesamten Datenbank einzigartig sind, unterstützt. Dazu muss der Datentyp von *int* auf *char(36)* oder *binary(36)* geändert werden.

Die Felder *created* und *modified* jeweils von dem Datentyp *datetime* sind ebenfalls Teil des *Convention over Configuration* Prinzips, sie sind allerdings optional. Wenn vorhanden, werden sie von CakePHP automatisch ohne das Zutun des Programmierers mitverwaltet.

Zur Erhöhung der Anschaulichkeit können bereits jetzt ein paar Beispieldatensätze in die Datenbanktabelle eingefügt werden. Dazu kann folgendes SQL Statement mehrmals mit veränderten Werten verwendet werden:

```
INSERT INTO 'jobs'
    ('company', 'title', 'description', 'created', 'modified')
VALUES
    ('Siemens AG', 'Elektrotechniker',
     'Das ist eine Beschreibung', NOW(), NOW());
```

Nach der Erzeugung der benötigten Datenbankstruktur wird üblicherweise das zugehörige *Model* als Datenbankabstraktion erstellt.

3.2 Model

Da aus Sicherheitsgründen und zur Sicherstellung des einheitlichen Zugriffs auf die Datenbank, die Ausführung von herkömmlichen SQL Statements in CakePHP vermieden werden sollten, bieten Modelklassen einige Funktionen um *CRUD* Operationen³ auf den Tabellen auszuführen. Hervorzuheben ist, dass Modelklassen auch den Zugriff auf Datenbanktabellen erlauben, die mit der entsprechenden Tabelle in Beziehung stehen. Außerdem können im Model auch Validierungsregeln für einzelne Tabellenfelder festgelegt werden.

Damit die Modelklassen richtig zu Datenbanktabellen zugeordnet werden können, müssen beim Erstellen von Modelklassen wiederum einige Konventionen beachtet werden: Models werden im Ordner *app/models/* angelegt, es handelt sich dabei um PHP Dateien, benannt werden sie nach der Name der Datenbanktabelle im Singular. Für die eben erstellte Datenbanktabelle *jobs* muss also die Datei *app/models/job.php* angelegt werden. Folgender Inhalt genügt, um CakePHP den einheitlichen Zugriff auf die Tabelle *jobs* zu gewähren:

³ Eine komplette Beschreibung aller CRUD Operationen befindet sich auf <http://book.cakephp.org/de/view/1017/Retrieving-Your-Data> und den folgenden Seiten, letzter Zugriff, 31. Juli 2011

```
<?php
class Job extends AppModel {
    public $name = 'Job';
}
```

Modelklassennamen werden im *camel case* geschrieben und beginnen mit einem Großbuchstaben. Models erweitern stets die Klasse *AppModel*, die in der Kerndateien von CakePHP definiert ist. Die Variable *\$name*, der der Modelname zugewiesen wird, stellt die Kompatibilität zu PHP 4 sicher. Nachdem das Model erstellt ist, muss für die Verwendung des Scaffoldings nur noch der zugehörige Controller erstellt werden.

3.3 Controller

Der Controller ist der Motor der Anwendung: Er wird anhand der angefragten URL aufgerufen und kümmert sich darum, dass die Daten unter Berücksichtigung der Parameter und Konditionen vom Model geholt werden. Er gibt diese dann an das View weiter, das dem Benutzer angezeigt wird [1].

Wie gewohnt, müssen auch bei den Controllerklassen gewisse Konventionen berücksichtigt werden. Alle Controller befinden sich im Ordner *app/controllers*. Controller sind ebenfalls PHP Dateien. Ihr Name setzt sich aus dem Namen der Tabelle mit angehängtem '_controller' zusammen. Für das konkrete Beispiel bedeutet das also, dass der für die Datenbanktabelle zuständige Controller in der Datei *app/controllers/jobs_controller.php* abgelegt wird. Ähnlich wie bei der Deklaration der Modelklassen, gibt es auch für den Inhalt von Controllern bestimmte Regeln. Der Klassenname beginnt mit einem Großbuchstaben und wird im *camel case* geschrieben. Er setzt sich aus dem Namen der Datenbanktabelle und einem angehängten 'Controller' zusammen. Alle Controllerklassen erweitern die *AppController* Klasse. Mit der Variable *\$name* kann wiederum für garantierte PHP 4 Kompatibilität gesorgt werden. Das Scaffolding selbst wird durch das Setzen der *\$scaffold* Variable aktiviert. Im konkreten Beispiel sieht der Code für die Controllerklasse wie folgt aus:

```
<?php
class JobsController extends AppController {
    public $name = 'Jobs';
    public $scaffold;
}
```

Da durch das Aktivieren von Scaffolding zugehörige Views, d.h. die fertigen HTML-Seiten die an den Client ausgeliefert werden, automatisch erzeugt werden, kann die Applikation bereits jetzt aufgerufen und getestet werden. Dazu wird an die URL zur CakePHP Installation - abgetrennt durch '/' - der Name

des eben erzeugten Controllers angehängt⁴.

Hier werden die bereits in der Datenbank befindlichen Jobangebote angezeigt. Vorhandene Jobangebote können detailliert betrachtet, editiert oder gelöscht werden. Außerdem ist das Anlegen neuer Jobangebote möglich. Auffallend ist der untere Bereich, in dem standardmäßig alle SQL Abfragen, die während der Erzeugung der Ausgabe ausgeführt wurden, angezeigt werden. Diese Informationen, die während der Entwicklungsarbeit sehr hilfreich sein können, sollen den späteren Nutzern der Webanwendung natürlich vorenthalten werden. Die Ausgabe wird zentral über das sog. *Debug Level* gesteuert, die Einstellung hierzu kann in der Datei *app/config/core.php* in Zeile 51 vorgenommen werden. Gültige Werte sind Zahlen aus dem Bereich 0-2. Der voreingestellte Wert '2' sorgt für die meisten Ausgaben inklusive SQL Abfragen. Der Wert '1' zeigt weiterhin alle Fehler und Warnungen an, blendet aber SQL Statements aus. Lediglich Debug Level '0' ist für den Produktiveinsatz gedacht, hier werden alle Debuginformationen vor dem Nutzer verborgen.

Nachdem nun die erste einfache Webanwendung erstellt und einsatzbereit ist, sollen nun die einzelnen Schritte erläutert werden, die bei einem *Request* an eine CakePHP Webanwendung ablaufen.

4 Behandlung eines Requests

Nachdem bisher nur die nötigsten Elemente einer CakePHP Webapplikation vorgestellt und eingesetzt wurden, wird nun ein ganzheitlicher Blick auf das Zusammenspiel der Komponenten entworfen. Abbildung 1 zeigt schematischen den Ablauf eines Nutzerrequests.

Eingehende Anfragen werden zunächst durch den *URL Dispatcher* geparst (1). Das hat den Hintergrund, dass URLs in CakePHP einer bestimmten Logik folgen. Funktionen in Controllerklassen, welche als *public* deklariert sind, werden auch *Actions* genannt. Diese Actions stellen die eigentlichen Funktionen der Webanwendung dar. Beispielsweise erzeugt der in Abschnitt 3 eingesetzte Scaffolding Mechanismus implizit die Actions *index*, *view*, *add*, *edit* und *delete*. Einige dieser Actions benötigen zusätzliche Parameter um die richtigen Aktionen ausführen zu können. So muss etwa der *delete* Action eine *Id* mit übergeben werden, damit CakePHP bestimmen kann, welcher Eintrag aus der Datenbank entfernt werden muss. Standardmäßig folgen alle URLs diesem Aufbau:

```
host/controller/action/parameter1/parameter2/parameter3/...
```

Der Aufruf `http://localhost/jobs/edit/4` würde also dazu führen, dass in der Controllerklasse 'Jobs' die Funktion 'edit' mit '4' als Parameter aufgerufen wird. Enthält eine URL lediglich den Namen des Controllers wird per Konvention innerhalb dieses Controllers nach einer index-Methode gesucht und diese aufgerufen, ist diese nicht vorhanden kommt es zu einer Fehlermeldung. Der URL

⁴ Für den Fall der Installation auf einem lokalen Webserver, bei dem sich CakePHP im Unterordner *htdocs/cake/* befindet, lautet die URL also `http://localhost/cake/jobs`.

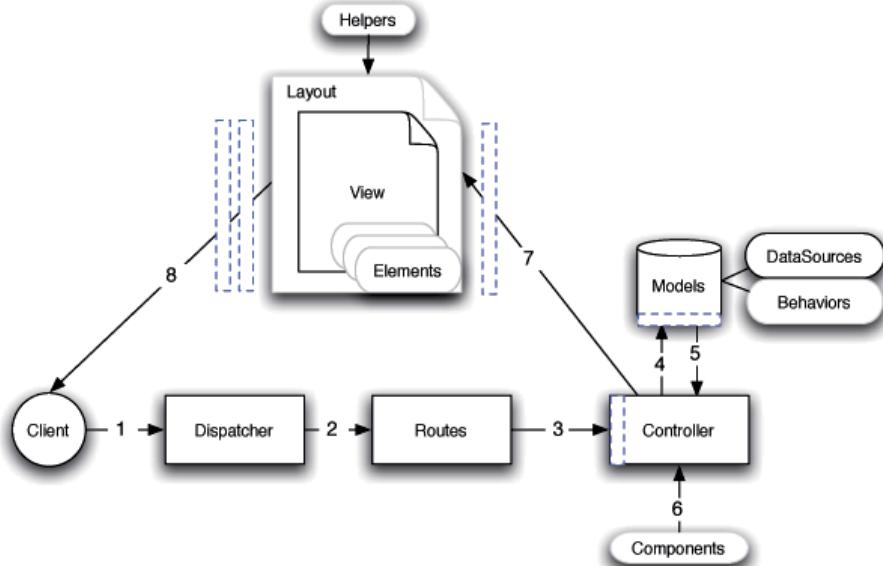


Abbildung 1. Ablauf und beteiligte CakePHP Komponenten bei der Behandlung eines vom Client eingehenden Requests [4].

Dispatcher löst eingehende URLs also nach diesen Regeln auf, um Requests den richtigen Controllern und Actions zuweisen zu können.

Manchmal kann es aus Gründen der Lesbarkeit von URLs, sowie der Suchmaschinenoptimierung sinnvoll sein, von diesem Standardaufbau abzuweichen. Ausnahmen von diesen Regeln können in der Datei `app/config/routes.php` definiert werden. Hier wird auch festgelegt, was beim Aufruf des `root` Verzeichnisses also `'/'` passieren sollen. Anfangs führt dieser Aufruf zu der in Abschnitt 2 vorgestellten Übersicht über die richtige Konfiguration von CakePHP. Nachdem die Konfiguration abgeschlossen ist, bietet es sich an, diese URL auf einen Controller umzuleiten. Bezogen auf das Jobangebotsbeispiel wird dazu Zeile 29 der `routes.php` durch folgende Zeile ersetzt:

```
Router::connect('/', array('controller' => 'jobs'));
```

Der Aufruf der Hostadresse führt jetzt zur Weiterleitung an den Jobs-Controller, dieses Verhalten wird durch das Abgleichen der Ergebnisse des URL Dispatchers mit der `routes.php` ermöglicht (2).

An bestimmten Zeitpunkten der Bearbeitung eines Requests erlaubt CakePHP sogenannte *Callback* Funktionen ablaufen zu lassen. Diese Callbacks können für jeden Controller einzeln definiert werden und gelten dann für alle Actions innerhalb dieses Controllers. Sie werden als *public function* in der jeweiligen Controllerklasse realisiert und anhand ihres Namens identifiziert, beispielsweise wür-

de die *public function beforeFilter* vor jedem Aufruf einer Action ausgeführt werden.

Die Funktionen *afterFilter* und *beforeRender* werden jeweils vor dem Rendern der View durchgeführt. Ein Unterschied besteht dann, wenn die Action keine eigene View besitzt: Die *afterFilter* Funktion wird trotzdem ausgeführt, *beforeRender* jedoch nicht. Neben den Controllerklassen können auch Models eine Reihe von Callbackfunktionen beinhalten, dazu sei an dieser Stelle auf die entsprechenden Seite in der CakePHP Dokumentation⁵ verwiesen. Callback Funktionen können auf verschiedenste Weisen genutzt werden, ein möglicher Anwendungsfall ist die Prüfung der Nutzereberechtigung im *beforeFilter*.

Die Anfragebearbeitung wird dann durch den Aufruf der eigentlichen Action im Controller fortgeführt. Nach möglichen Interaktionen mit dem Model (4, 5) - denkbar ist etwa das Aufbereiten von Daten zur Anzeige in der View oder das Speichern eines neuen Datensatzes in der Datenbank - können CakePHPs *Components* eingesetzt werden (6), die speziell für die Controllerklassen häufig benötigte Funktionalität zur Verfügung stellen und die in Abschnitt 6 gesondert vorgestellt werden.

Vor der Auslieferung der fertigen Seite (8) an den Client, erfolgt noch das Rendern der eigentlichen View (7). Dazu können Daten vom Controller an die View übergeben werden. Views werden in das sogenannte Layout eingebettet. Das Layout eignet sich demnach für alle statischen Elemente einer Seite, also z.B. für Kopfleiste und Navigation. Elements sind in sich geschlossene Bauteile einer Seite, die dann in den Views wiederverwendet werden können. Ein Beispiel hierfür wäre eine Hilfebox, deren Text beim Erzeugen dynamisch gewählt werden kann und, die in den verschiedenen Views an verschiedenen Stellen platziert werden kann. Während Components die Arbeit in den Controllern erleichtern, werden *Helper* in Layout, View und Elements eingesetzt. Die vorhandenen Helper werden überblicksartig in Abschnitt 6 beleuchtet, doch zunächst soll der in Abschnitt 3 teilweise manuell nachimplementiert werden, um ein besseres Verständnis der Views CakePHP zu erlangen.

5 Erweiterung des Prototyps

Um das Konzept und die Funktionsweise der Views zu erklären, wird in diesem Abschnitt die erstelle Jobangebotsbörse abgeändert. Für das Anlegen von Jobangeboten soll nicht mehr der Scaffolding Mechanismus, sondern manuell erstellter Code verwendet werden. Die von der *\$scaffold* Variable erzeugten Funktionen können einzeln überschrieben werden. Zunächst muss der Controller geöffnet und folgende Funktion eingefügt werden:

```
function add() {
    if (!empty($this->data)) {
        $this->Job->create();
```

⁵ <http://book.cakephp.org/de/view/1048/Callback-Methods>, letzter Zugriff 31. Juli 2011

```

        if ($this->Job->save($this->data)) {
            $this->Session->setFlash('Job angelegt!');
            $this->redirect(array('action' => 'index'));
        } else {
            $this->Session->setFlash('Fehler!');
        }
    }
}

```

Um die Arbeitsweise dieser Funktion zu erklären, muss bekannt sein, dass am Ende einer Action im Controller, immer nach einer zugehörigen View gesucht und diese dann angezeigt wird. Zunächst werden zwei verschiedene Fälle unterschieden. Der erste Fall ist, dass die oberste *if*-Bedingung nicht zutrifft und *\$this->data* (Diese Datenstruktur enthält alle *Post* Parameter des aktuellen Requests) leer ist. In diesem Fall wird direkt die zugehörige View aufgerufen.

Andernfalls, wird durch den Befehl *\$this->Job->create()*; ein neues Job-Objekt angelegt. In der zweiten (*if*)-Bedingung wird dann versucht das neue Job-Objekt mit den übergebenen Post Parametern zu füllen und in der Datenbank zu speichern. Ist das erfolgreich, wird mit Hilfe der *Session Component* und deren Funktion *setFlash* eine Erfolgsmeldung festgelegt. Die Funktion *redirect* ermöglicht direkt bestimmte Controller und Actions aufzurufen, wird nur der Name einer Action übergeben, so wird diese im aktuellen Controller ausgeführt. *Redirect* verhindert dabei auch, dass die eigentlich zur Action gehörende View gerendert wird und ermöglicht damit Actions zu definieren, die nicht zwingend eine eigene View besitzen müssen. Schlägt das Speichern des neuen Jobangebots fehl, z.B. weil falsche Post Parameter vorhanden sind, oder Validierungsregeln verletzt werden, wird mithilfe der *setFlash* Funktion eine Fehlermeldung ausgegeben.

Für den Fall, dass gar keine Post Parameter im Request vorhanden sind, muss jetzt noch eine geeignete View erstellt werden. Grundsätzlich sind dabei folgende Konventionen zu beachten: Für jeden Controller, der mindestens eine Action mit eigener View hat, wird im Ordner *app/views/* ein weiters Verzeichnis mit dem Namen des Controllers angelegt, im Beispiel *app/views/jobs/*. Für die einzelnen Actions des Controllers werden dann innerhalb dieses Unterverzeichnisses gleichnamige *.ctp (Cake Template)* Dateien angelegt. Im konkreten Fall soll die *add* Funktion des *Jobs*-Controllers eine View bekommen, also lautet der Pfad der zugehörigen Datei: *app/views/jobs/add.ctp*. Bei dem geringen Inhalt dieser Datei, zeigt sich eine der Stärken von CakePHP:

```

<h2>Neuer Job</h2>
<?php
    echo $this->Form->create('Job');
    echo $this->Form->input('company');
    echo $this->Form->input('title');
    echo $this->Form->input('description');
    echo $this->Form->end('Speichern');

```

```

echo $this->Html->link('zurueck',
    array('controller' => 'jobs',
        'action' => 'index'));
?>

```

Hier wird der *Form-* und *HTML-Helper* eingesetzt. Mit dem *Form->create('Job')* Befehl wird für CakePHP kenntlich gemacht, für welches Model das erzeugte Formular zuständig ist. Mit der *input(\$field)* Funktion werden die Datenbanktabellenfelder angegeben, für die Eingabemasken erzeugt werden sollen. CakePHP erzeugt dann automatisch anhand des Datentyps dieses Feldes in der Datenbank passende Eingabemöglichkeiten. Die *end(\$text)* Funktion erzeugt dann noch einen Submit Button mit der übergebenen Beschriftung.

Um auf die Jobübersicht zurückkehren zu können wird mithilfe der *link* Funktion des HTML Helpers noch ein entsprechender Link definiert. Die Funktion nimmt dabei zwei Parameter entgegen: Der erste ist der Linktext, der zweite ist ein assoziatives Array, in dem die aufzurufende Action und der Controller definiert werden.

Jetzt kann die eigene *add* Action angewendet werden. Klickt der Nutzer auf der Übersicht auf den 'New Job' Knopf, werden keine Post Parameter übergeben, weshalb der zugehörige View mit dem Eingabeformular angezeigt werden. Werden hier Daten eingegeben und auf den Submit Knopf geklickt, wird wiederum die *add* Action aufgerufen, dieses Mal sind aber passende Post Parameter vorhanden und das Jobangebot kann angelegt werden.

Nachdem dieses Beispiel nun einen kleinen Eindruck von der Vielseitigkeit von CakePHP gegeben hat, werden im nächsten Abschnitt weitere hilfreiche Konzepte angesprochen.

6 Weitere Konzepte

Da im Rahmen dieser Arbeit natürlich nur ein Bruchteil der theoretisch vorhandenen Möglichkeiten von CakePHP erläutert geschweige denn an konkreten Codebeispielen demonstriert werden können, werden in diesem Abschnitt zusätzliche Konzepte überblicksartig vorgestellt.

6.1 Components

Components sind Teil der CakePHP Kerndateien und enthalten Applikationslogik für verschiedene Aufgabenbereiche, die die meisten typischen Anwendungsfälle bei der Entwicklung von Webanwendungen abdecken. Sie werden dabei ausschließlich innerhalb der Controllerklassen verwendet. Wenn bei der Entwicklung eigener Anwendungen, immer wieder gleiche Codeabschnitte innerhalb verschiedener Controller gebraucht werden, bietet sich die Definition einer eigenen Component an. Selbsterstellte Components werden im Ordner *app/controllers/components* abgelegt. Tabelle 1 zeigt die bereits in CakePHP vorhandenen Components.

Component	Funktion
Sessions	Steuert das Session-Handling innerhalb der Applikation.
Cookies	Setzt Cookies und liest sie aus.
ACL	Access Control List: Steuert Userberechtigungen.
Authentication	Component für einfache Authentifizierung
Email	Kapselt das Versenden von Text- und HTML-E-Mails.
Security	Fasst verschiedene sicherheitsrelevante Funktionen zusammen.
Request Handler	Gibt Informationen über den Request und den Client zurück.

Tabelle 1. Die Core Components von CakePHP [1].

6.2 Helpers

Analog der Rolle der Components für die Controller, unterstützen die Helpers den Programmierer bei der Ausgabe von Webseiten an den Client. Obwohl der Programmierer innerhalb der Views, alle Html Tags auch auf herkömmliche Weise definieren könnte, bietet es sich an, wenn immer möglich auf vorhandene Helper zurückzugreifen. Dadurch kann wiederum Zeit gespart werden und der Aufbau der View bleibt einheitlich. Neben den bereits vorhandenen Helpern können eigene Helperklassen in *app/views/helpers* definiert werden. Tabelle 2 beinhaltet die wichtigsten vordefinierten Helper und gibt Hinweise auf deren Anwendungsmöglichkeiten.

6.3 CakePHP Konsole

Bei der CakePHP Konsole handelt es sich um eine Anwendung, die sogenannte *Shells* zu verschiedenen Aufgabengebieten anbietet. Hier können unter anderem die Grundgerüste für alle Bestandteile einer CakePHP Anwendung, Lokalisierung und Internationalisierung sowie Testklassen, generiert werden. Die Konsole befindet sich in *cake/console/console.bat*. Als Voraussetzung für die CakePHP Konsole muss der PHP Interpreter in der Kommandozeile erreichbar sein. Ob das der Fall ist kann durch die Eingabe von

```
php -v
```

überprüft werden. Werden Informationen zur verwendeten PHP Version ausgegeben, ist PHP bereits in die Umgebungsvariablen des Betriebssystem integriert. Ist das nicht der Fall muss die Variable *Path* bearbeitet werden und der Pfad zu PHP hinzugefügt werden. Es bietet sich an, den Pfad zur Konsolenanwendung ebenfalls in dieselbe Variable einzutragen, damit in jedem Verzeichnis direkt über den Befehl *cake* aufgerufen werden kann. Für einen lokalen, mit XAMPP betriebenen Webserver sieht die Ergänzung der *Path* Variablen in etwa folgendermaßen aus:

Helper	Funktion
Html	Unterstützt Sie bei der Erstellung von HTML-Tags und erleichtert die Ausgabe komplexer HTML-Strukturen, zum Beispiel von Links und Tabellen.
Form	Erstellt Formularelemente und befüllt diese automatisch. Darüber hinaus wird über diesen Helper das Ergebnis der Validierung von Eingaben ausgegeben.
Number	Formatierung von Zahlen
Js	Dient zur Einbindung von JavaScript-Libraries und kapselt den Zugriff auf verschiedene JavaScript-Funktionalitäten. [...]
Paginator	Seitennummerierung und Sortierung. [Ermöglicht die Aufteilung großer Datenmengen auf mehrere Seiten, durch die der Nutzer navigieren kann.]
Session	Helper für den Umgang mit Sessions und den darin gespeicherten Werten
Text	Formatierung von Texten und Links
Time	Formatierung von Zeit- und Datumsangaben sowie einige logische Funktionen zum Umgang mit diesen Werten, beispielsweise für Berechnungen.

Tabelle 2. Auswahl der in CakePHP verfügbaren Helperklassen [1].

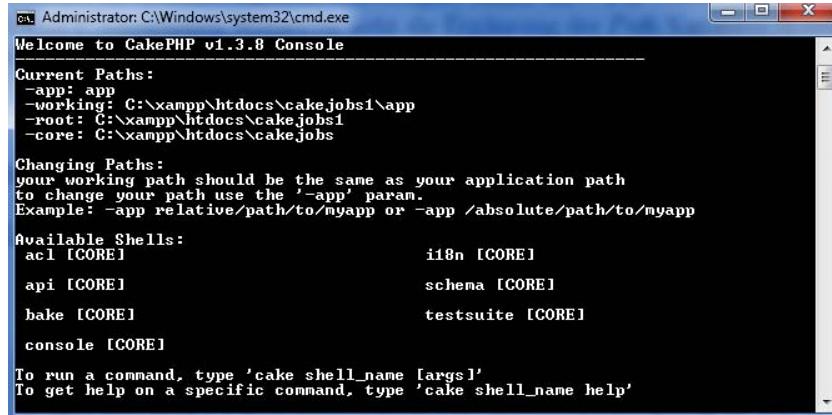


Abbildung 2. CakePHP Konsole: Übersicht über die angebotenen Shells.

C:\xampp\php;C:\xampp\htdocs\cakejobs\cake\console;

Die Konsole sollte immer innerhalb eines *app*/ Ordners genutzt werden, damit klar ist, auf welche CakePHP Anwendung der Aufruf bezogen ist. Die Eingabe von 'cake' erzeugt zunächst einen Überblick über alle vorhandenen Shells, wie sie in Abbildung 2 zu sehen ist. Der Aufruf einer konkreten Shell erfolgt über den Befehl 'cake <shellname>', also z.B. 'cake bake', wenn die *bake* Shell aufgerufen werden soll. Diese Shell ist vor allem für Einsteiger in CakePHP die relevanteste. Alle Strukturen einer Applikation können einzeln oder auch als ganzes Projekt erzeugt werden.

Es ist sogar möglich, den Code, der bei Setzen der *\$scaffold* Variable implizit erzeugt wird, explizit in die generierten Controller und Views einzufügen, sodass dieser leicht anpassbar ist. Dazu muss zunächst ein Model manuell oder per *Bake* Shell erstellt werden, danach den zugehörigen Controller mit der shell erzeugen und nach der Auswahl der entsprechenden Datenbanktabelle die Frage nach dem dynamischen Scaffolding verneinen. Die nachfolgende Frage zur Erzeugung der grundlegenden Klassenmethoden bejahen.

Die Cake Shells sind umfangreiche und nützliche Werkzeuge für die Entwicklung von CakePHP Anwendungen, lediglich das Fehlen einer Möglichkeit Datenbanktabellen über die Konsole zu erstellen kann kritisiert werden. Wie die meisten anderen Bestandteile von CakePHP kann auch die Konsole mit eigenen Shells erweitert werden, z.B. ist in vielen Fällen eine Shell zur Steuerung von *Cronjobs* sinnvoll einsetzbar.

7 Fazit und Ausblick

Selbstverständlich kann im Rahmen einer Seminararbeit kein ganzheitlicher Eindruck über ein so umfangreiches Webframework wie CakePHP vermittelt wer-

den, dennoch soll im Folgenden eine qualitative Bewertung versucht werden. Eine große Stärke von CakePHP ist sicherlich die Verwendung von PHP als Programmiersprache, die weiterhin eine deutlich höhere Verbreitung als *Ruby* oder *Python* vorweisen kann. Das Webframework ist sehr leicht zugänglich, sei es durch die einfache Konfiguration oder durch die sehr gute Dokumentation, die auch ein umfangreiches Tutorial enthält. Die Struktur von CakePHP ist in sich schlüssig und sehr gut nachvollziehbar. Erste sichtbare Ergebnisse können in sehr kurzer Zeit erzielt werden. Ein weitere Pluspunkt ist die sehr aktive Community, die offizielle *Bakery*⁶ (die CakePHP Communityseite) ermöglicht Kommunikation mit erfahrenen CakePHP Entwicklern, außerdem können hier Komponenten, wie neue Helper und Components heruntergeladen werden. Gerade für Einsteiger ist sicher auch die Tatsache hilfreich, dass CakePHP ein *full stack* Framework ist und somit die Suche nach passenden Komponenten, die z.B. bei *Zend Framework* der eigentlichen Programmierung vorausgeht, entfällt. Andererseits büßt ein Webframework durch die Auslieferung als Komplett paket natürlich immer auch Flexibilität ein.

Als wirklicher Negativpunkt muss die vergleichsweise schlechte Performance von CakePHP Systemem genannt werden. Tatsächlich ist CakePHP bei Projekten, bei denen vergleichsweise viele Requests pro Zeiteinheit beantwortbar sein müssen, nicht mit *Ruby on Rails* oder *django* konkurrenzfähig. Die Möglichkeit Datenbanktabellen direkt über eine Konsolenshell zu erstellen, sollte auf jeden Fall in zukünftigen Versionen in Erwägung gezogen werden. Ein Kritikpunkt ist das schon erwähnte starke Imitieren von Konzepten des *Ruby on Rails* Frameworks. Ob die Verwendung der Kuchenmetapher als störend oder unseriös empfunden wird, ist eine Frage des persönlichen Geschmacks, in den meisten Fällen dürfte sie aber nicht als nachteilig empfunden werden.

Innerhalb dieses Jahres wird die finale Version 2.0 des CakePHP Frameworks veröffentlicht werden. Vor allem die Umstellung auf *Lazy Loading* der Models, Controller und Helper könnte eine signifikante Performanceverbesserung bewirken. In jedem Fall wird CakePHP aufgrund der einfachen Zugänglichkeit und des hohen Funktionsumfanges auch weiterhin eine Rolle als ernstzunehmendes Webframework spielen.

Literatur

1. Ammelburger Dirk, Scherer Robert. *Webentwicklung mit CakePHP*, 2. Auflage. O'Reilly Verlag, Köln, 2011.
2. Hudson Paul. *PHP in a nutshell*, 1. Auflage. O'Reilly Verlage, Köln, 2006.
3. *CakePHP: the rapid development PHP framework*. <http://cakephp.org>, letzter Zugriff 31. Juli 2011.
4. *The CakePHP 1.3 Book*. <http://book.cakephp.org>, letzter Zugriff 31. Juli 2011.
5. *CakePHP 1.3 API*. <http://api13.cakephp.org/>, letzter Zugriff 31. Juli 2011.

⁶ <http://bakery.cakephp.org/>, letzter Zugriff 31. Juli 2011

Zend Framework – Ein Überblick

Martin Schmalfuß

Abstract. Diese Arbeit gibt einen groben Überblick über den Funktionsumfang und die Verwendungsmöglichkeiten des Zend Frameworks. Neben Vor- und Nachteilen werden wichtige Funktionalitäten beleuchtet. Zum besseren Verständnis dient ein fiktives Beispielprojekt eines Web-Logs als roter Faden durch das Dokument, welcher eine mögliche Kombination aller vorgestellten Einsatzmöglichkeiten darstellt und somit die Interoperabilität der einzelnen Komponenten verdeutlicht. Neben Kernaufgaben des Frameworks, wie das MVC-Pattern, Formularbearbeitung oder die Datenbank-Anbindung, werden auch noch weitere Aspekte wie bspw. die Integration von Web Services näher betrachtet. Die Arbeit schließt ab mit einem Ausblick auf kommende Entwicklungen beim Zend Framework.

Keywords: Zend, Framework, Einführung, Vorteile, Nachteile, MVC, Datenbank, Formulare, Filter, Validierung, Mail, Nutzerverwaltung, Authentifizierung, Web Services, Feeds, Amazon, Google, Internationalisierung, Anpassung, Umrechnungen, Caching, Aktuelles, Ausblick

1 Allgemeine Betrachtung

1.1 Vorteile des Zend Frameworks (ZF)

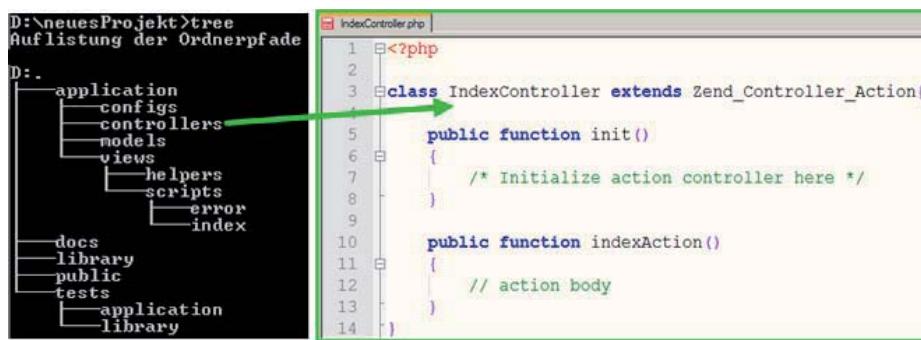
Wie der Name bereits vermuten lässt, handelt es sich um ein Framework. Somit sind die Vorteile eines Frameworks auch bei diesem gültig:

- Code-Ersparnis bei eigenem Code durch Übernahme bereits implementierter Komponenten
- Entwicklungs-Erleichterung der Gesamtanwendung
- Stabilität und Zuverlässigkeit des übernommenen Codes
- Zeitersparnis

Bereits beim Einrichten des Projekts auf dem Rechner unterstützt das im Lieferumfang enthaltene, kommandozeilenbasierte Zend-Tool den Benutzer. Über

```
zf create project neuesProjekt
```

wird ein neues Projekt („neuesProjekt“) inklusive aller initialen Ordner, Klassen, Interfaces und Methoden erstellt [1]. Im nachfolgenden Screenshot dargestellt für die automatische Erstellung der IndexController.php:



Über das Zend-Tool lassen sich ebenso im späteren Entwicklungsverlauf viele Bestandteile (Model, View, Controller, Config, Form, Layout, ...) erstellen.

Einige Vorteile ergeben sich aus der losen Kopplung der einzelnen Komponenten. Beim ZF handelt es sich um ein „Glue Framework“ [2], welches sich dadurch auszeichnet, dass die Komponenten beliebig kombiniert (verklebt) und eingebunden werden können. Es existiert somit keine vorgeschriebene Architektur der Bestandteile, sodass der Benutzer seine eigene Anwendungsstruktur beibehalten kann. Diese lose Kopplung minimiert zudem die Abhängigkeiten der Komponenten untereinander, was einen „schlanken“ Einsatz ermöglicht.

Ebenfalls bedingt durch die lose Kopplung können Komponenten in Entwicklungsumgebungen mit fremden Frameworks eingesetzt werden, sodass das ZF indirekt den Funktionsumfang fremder Frameworks erweitern kann [3]. Dies ist möglicherweise auch nachteilig für Zend, da der Fokus bei Entwicklern somit eher auf anderen Frameworks liegen könnte.

Da ohne die nähere Kenntnis eines Strukturzusammenhangs der Komponenten untereinander diese trotzdem eingesetzt werden können, ist das ZF aus eigener Erfahrung vergleichsweise anfängerfreundlich, mehr als die Funktion des übernommenen Codes muss dem Benutzer nicht bekannt sein.

Eine der Kernaufgaben bei der Entwicklung von ZF ist das MVC-Pattern [4]. Daraus lässt sich schließen, dass das MVC-System stabil und fortschrittlich umgesetzt wird, was bei der Entwicklung größerer Anwendungen von Vorteil ist. Natürlich wird auch auf ein modernes Software-Engineering geachtet, bspw. durch die Nutzung diverser Design-Patterns [5].

Zend selbst als einer der führenden Entwickler von PHP (Zend Engine I + II [6]) dürfte für die Zukunftssicherheit und ständige Weiterentwicklung des Frameworks die beste Garantie sein. Erweiterbar wird die Entwicklungsumgebung des Benutzers durch weitere Produkte von Zend:

- Zend Server (inkl. für ZF optimiertem Logging und Caching-Backend) [7]
- Zend Studio (Entwicklungs-Komplett paket, basierend auf Eclipse PDT) [7]

Aufgrund der großen Community hinter Zend ist der Support von Seiten anderer Benutzer sehr gut. Auch Zend selbst stellt eine umfangreiche, mehrsprachige Dokumentation¹ bereit, welche zudem anschaulich an Anwendungsbeispielen die jeweilige Funktionsweise erklärt. Diese Dokumentation ist Pflicht für alle Bestandteile des ZF [8].

Weiterführend eignet sich ZF auch für den kommerziellen bzw. professionellen Einsatz, die Lizenz-Restriktionen der New BSD-Lizenz^{2,3} sind weniger scharf als bei anderen Frameworks, was die gewerbliche Nutzung vereinfacht.

1.2 Nachteile des Zend Frameworks (ZF)

Die allgemeinen Nachteile des Einsatzes eines Frameworks gelten auch beim ZF, wobei die folgenden beiden Punkte aufgrund des Glue-Frameworks eher geringer ausfallen als bei Konkurrenzprodukten:

- Relativ hohe Einarbeitungszeit
- Für Anfänger ungeeignet

Ein größerer Kritikpunkt ist die fehlende ORM-Unterstützung [9], [10]. Durch die Community werden allerdings bereits einige Eigenimplementierungen⁴ angeboten. Alternativ kann auf Drittanbieter wie bspw. Propel⁵ zurückgegriffen werden.

Die Dokumentation des ZF ist äußerst umfangreich, jedoch fehlt ein Gesamtbild, in welchem alle Bestandteile in einer Architektur dargestellt werden. Bedingt durch die lose Kopplung besteht die Gefahr, dass das Framework eher als Library angesehen werden könnte.

2 Einsatz-Beispiel

Dieses Beispiel soll als roter Faden dienen und die Breite der Einsatzmöglichkeiten aufzeigen. Es geht um die Erstellung eines Web-Logs, welcher folgende Funktionalitäten beinhalten soll:

¹ <http://framework.zend.com/docs/overview>

² <http://framework.zend.com/license>

³ <http://de.wikipedia.org/wiki/BSD-Lizenz>

⁴ <http://code.google.com/p/zend-framework-orm/>

⁵ <http://www.propelorm.org/>

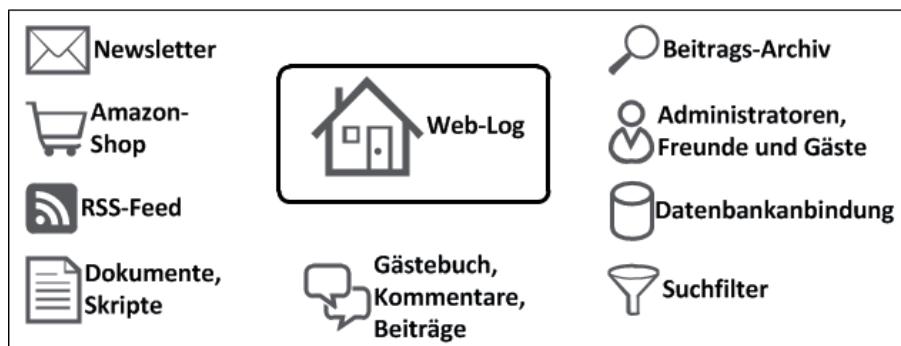


Abbildung 2: gewünschte Web-Log Funktionen

Für jede dieser Funktionen bietet ZF eine Lösung [11] an:

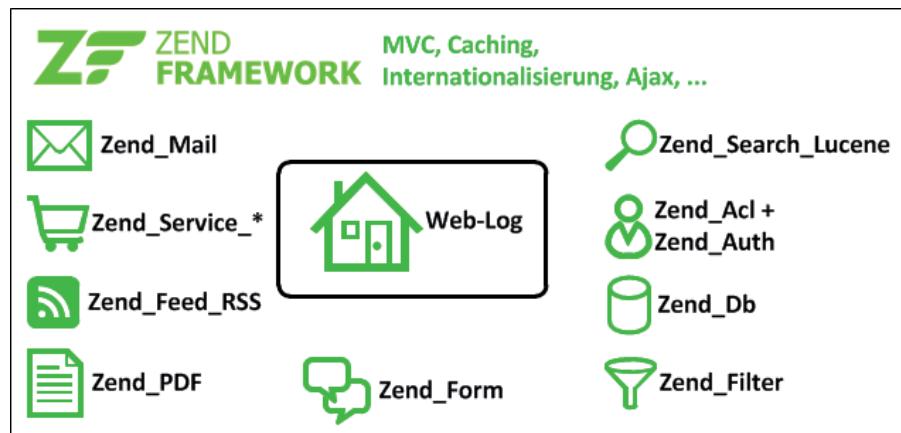


Abbildung 3: Umsetzung der Funktionen mit ZF

Einige dieser Lösungen werden anschließend vorgestellt, jedem Kapitel ist die entsprechende Einsatzmöglichkeit vorangestellt.

3 Umsetzung des MVC-Patterns

Einsatzmöglichkeit. Dient einem modularen, flexiblen und übersichtlichen Aufbau. Für spätere Erweiterungen und Code-Wiederverwendung geeignet. Bietet sich bei jeder größeren Anwendung an.

3.1 Allgemeines und Start

Für den initialen Aufbau sollte das Zend-Tool verwendet werden, wie bereits gezeigt.

Der Aufbau und Ablauf einer Anwendung mittels MVC-Pattern dürfte bekannt sein. Der clientseitige Zugriff geschieht über die Bootstrap-Datei⁶, welche aus Sicherheitsgründen [12] als einzige in /public abgelegt ist⁷. Für den Front-Controller wird ein Singleton-Pattern umgesetzt [13], um Redundanzen zu vermeiden. Damit kann er als Registry eingesetzt werden, um bspw. Plugin-Broker am System anzumelden. Des Weiteren werden im Front-Controller auch die Environment-Variablen gesetzt[13].

Für die gesamte Anwendung sind Dispatcher⁸ und Router⁹ konfigurierbar [14]. Bei Einhaltung der Namenskonventionen[15] ist dies allerdings nicht zwingend notwendig.

Nachfolgend ist ein vereinfachtes Ablaufschema in Anlehnung an [16] dargestellt:

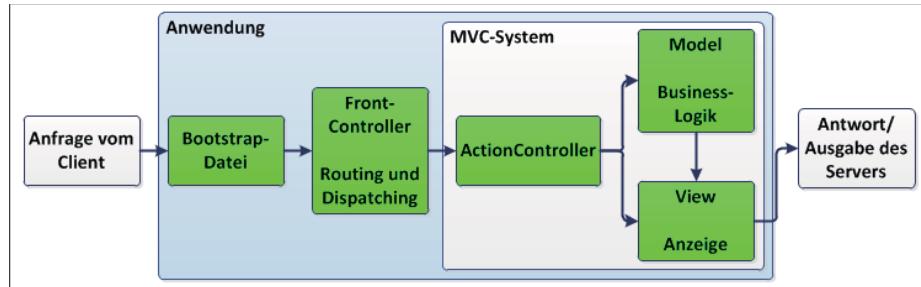


Abbildung 4: Ablauf MVC bei Anfrage

3.2 Controller

Alle Controller werden aus Zend_Controller_Action abgeleitet. Je Controller können beliebig viele Actions implementiert werden. Dank den Namenskonventionen kann auf diese einheitlich zugegriffen werden. Die Dokumentenstruktur ist den Namenskonventionen entsprechend aufgebaut. Als initiale Action dient die indexAction().

Des Weiteren sind im Controller Referenzen auf die View (Zend_View) möglich, sodass eine Verknüpfung von Controller und View über die Variable \$view umsetzbar ist [17].

⁶ http://de.wikipedia.org/wiki/Bootstrapping_%28Informatik%29

⁷ minimaler Zugriff auf die Anwendung von außen, Anfragen werden zentral gekapselt

⁸ <http://framework.zend.com/manual/de/zend.controller.dispatcher.html>

⁹ <http://framework.zend.com/manual/de/zend.controller.router.html>

3.3 Model

Für das Model gibt es von Zend keine festen Vorgaben, die Umsetzung der Businesslogik bleibt dem Benutzer selbst überlassen. Unterstützt wird er hierbei durch die Möglichkeit der Datenbankanbindung (`Zend_Db_*`) und die verfügbaren Schnittstellen zu Web Services (`Zend_Service_*`).

3.4 View

Die View ist im Front-Controller (de-)aktivierbar. Verwendete Templates werden in `/view/scripts` abgelegt. Da ZF auf PHP basiert, ist es dieses Template-System ebenfalls. Das Einbinden weiterer Template-Systeme wie bspw. Smarty ist möglich [18].

4 Datenbank-Zugriff

Einsatzmöglichkeit. Beinahe jede Web-Präsenz benötigt eine Datenbank zum Verwalten von Daten. Im Falle des Web-Logs bspw. Beiträge, Gästebucheinträge, Zugriffsstatistiken oder Zugangsdaten.

Für den Datenbankzugriff stehen austauschbare Abstraktionslayer¹⁰ zur Verfügung, was eine gewisse Unabhängigkeit vom Datenbanktyp schafft. Diese implementieren das Table Data Gateway Pattern¹¹[19], welches den entsprechenden SQL-Code wrapppt.

Der Zugriff auf die Datenbank kann über zwei Wege geschehen. Entweder über die Definition eines eigenen Adapters¹² oder über den tabellenbasierten Zugriff¹³ [20]. `Zend_Db_Table` implementiert dabei das Active Record Pattern¹⁴, welches eine Tabelle in einem Objekt kapselt und als Objekt behandelt.

Für die PDO-Auswahl wird die statische `factory()`-Methode¹⁵ eingesetzt, welche ein Array mit den benötigten Zugangsdaten übergeben bekommt. Alternativ lässt sich auch `Zend_Config` nutzen [21][22], diese Komponente erlaubt das Auslesen von Konfigurationsdaten aus Arrays, INI- oder XML-Dateien.

Prepared Statements sind natürlich möglich (`prepare()` und `execute()`), SQL-Injections werden mittels Verwendung von `quote()` bzw. `quoteInto()` vermieden.

¹⁰ Besitzt Adapterfunktion zwischen Datenbank und Programmiersprache

¹¹ <http://www.longacre-scm.com/blog/index.php/2006/02/design-pattern-table-data-gateway>

¹² Ableitung von `Zend_Db_Adapter`

¹³ Ableitung von `Zend_Db_Table`

¹⁴ <http://blog.oncode.info/2008/03/12/active-record-pattern-in-php/>

¹⁵ Verwendung des Factory-Pattern

Für die Ausgabe der Datenbank-Daten bietet sich die `fetch()`-Methode an. Hiervon gibt es noch zahlreiche Abwandlungen je nach Bedarf, bspw. `fetchAll()`, `fetchRow()`, `fetchCol()` etc.

Besonders nützlich ist die Umsetzung von Transaktionen [23]. Diese sind entgegen bisheriger Datenbankoperationen bis zum abschließenden `commit()` reversibel. Umgesetzt wird eine Transaktion einfach über folgende Befehle:

```
beginTransaction();  
  
//Datenbank-Operationen...  
  
commit();           //bei Abbruch: rollBack();
```

Zur Analyse der Geschwindigkeit der Datenbankanbindung ist in Zend_Db ein Profiler integriert [24]. Dieser fungiert als Abfragenprotokoll, dessen Ergebnisse sind direkt in Firebug (Mozilla Firefox Add-on) auslesbar (siehe Abbildung), was das Debugging erheblich vereinfacht:

```
//Profiler wird gestartet, Verbindung zu Firebug  
//eingerichtet  
$profiler = new Zend_Db_Profiler_Firebug('DB-  
Abfragen');  
$profiler->setEnabled(true);  
  
$db->setProfiler($profiler);  
  
$request  = new Zend_Controller_Request_Http();  
$response = new Zend_Controller_Response_Http();  
$channel  =  
Zend_Wildfire_Channel_Headers::getInstance();  
$channel->setRequest($request);  
$channel->setResponse($response);  
  
//Start der Aufzeichnung  
ob_start();  
  
//Programmcode mit Datenbankzugriffen  
  
$channel->flush();  
$response->sendHeaders();
```

Nachfolgend das Resultat der Integration in Mozilla Firebug:



Abbildung 5: Integration von Mozilla Firebug

5 Formulare

Einsatzmöglichkeit. Überprüfung von Eingaben bei Formularen wie bspw. Bestellungen, Gästebucheinträgen, Kontaktformularen, Anmeldemasken, etc.

5.1 Validierung

Mittels Zend_Validate können eine Vielzahl von Daten überprüft werden, wie bspw. Werte, Bereiche, Buchstaben, Text, Datumsangaben, Syntax von Kreditkartennummern, Arrays etc. Hervorzuheben ist die Validierung von Mailadressen, welche sich nicht nur auf die Syntax beschränkt, sondern auch die MX-Records überprüft. Hostnames lassen sich einerseits mittels DNS prüfen, andererseits über die IP-Adresse, auch lokale Hostnames sind möglich [25].

Prinzipiell werden alle Werte mittels der isValid()-Methode der entsprechenden Klasse überprüft, sodass die Validierungssyntax der verschiedenen Prüfungen auf ein Minimum beschränkt bleibt.

Die Fehlermeldungen bei Nichtbestehen von Validierungen lassen sich mittels setMessage() einfach selbst definieren und anpassen [26].

5.2 Filter

Über Zend_Filter lassen sich einige Daten filtern, bspw. Werte, Buchstaben, Ausgabe der BaseNames, Ausgabe des Directory, Digits, htmlEntities, stripTags (Entfernung von html-tags, leistungsfähiger als PHP-Standard) etc.

Eigene Filter können über Zend_Filter_Interface abgeleitet und definiert werden. Dabei ist auf eine Implementierung der filter()-Methode zu achten.

Ein Spezialfall stellt Zend_Filter_Input dar. Hiermit können Validatoren und Filter verknüpft werden [27]. Dazu wird einfach der folgende Code verwendet:

```
$ergebnis = new Zend_Filter_Input($filterArray,
$validatorenArray, $daten, $optionen);
```

Zu Debugging-Zwecken lassen sich mittels getMessages() alle während des Vorgangs erzeugten Meldungen ausgeben.

5.3 eMail

Zend erweitert hier das bisherige Angebot von PHP-Funktionen deutlich. Der PHP-Standard sieht nur eine Abrufmöglichkeit von Mails vor, der Versand funktioniert ausschließlich über den lokalen MTA¹⁶. Mit Zend_Mail lässt sich die gesamte Funktionalität eines Mailclients wie bspw. Mozilla Thunderbird (bezogen auf Mailempfang und –versand) realisieren, es kann über externe SMTP-Server gesendet werden, alle IMAP-Funktionen (Flags, Ordnerstrukturen) sind implementierbar. So sind Header definierbar (Dringlichkeiten etc.), HTML-Mails stellen kein Problem dar [28].

Auch Anhänge bei HTML-Mails sind möglich über `createAttachment()`, diese werden serialisiert um sie den Mails zuordnen zu können [29].

Eine Verschlüsselung der Übertragung mittels SSL ist möglich.

6 Nutzerverwaltung und Authentifizierung

Einsatzmöglichkeit. Sollen Nutzergruppen unterschiedliche Rechte bekommen, bspw. eine Unterscheidung im Web-Log von Beitragssautoren, Besuchern und angemeldeten Benutzern vorgenommen werden, ist eine Nutzerverwaltung inkl. Authentifizierungsmöglichkeit unerlässlich.

6.1 Zugriffsrechte

Ressourcen (Zugriffsrechte) werden für Rollen (Nutzertypen) definiert. Zunächst werden die Ressourcen als Ableitungen von Zend_Acl_Resource angelegt, danach die entsprechenden Rollen (`addRole()`) über Zend_Acl_Role.

Das Zugriffsrecht auf eine Ressource wird mittels `allow()` oder `deny()` gewährt oder verweigert. Die Rechte sind hierbei jederzeit wieder entziehbar bzw. setzbar. Die Vererbung von Rechten ist vorgesehen, sodass die Rechte von übergeordneten Rollen übernommen werden.

Ausnahmen der Regeln können über `assert()` definiert werden, bspw. für eine zeitliche Begrenzung (Beispiel: Zugang von 0:00 – 5:00 für nicht jugendfreie Medien) [30].

6.2 Authentifizierung

Um einem Benutzer Zugang zu Ressourcen zu gewährleisten, muss dieser sich gegenüber dem System als berechtigte Instanz authentifizieren. Hierbei wird der Nutzernname und das Passwort gegen eine Datenbank oder eine Datei abgeglichen. Der Nutzernname ist setzbar mittels `setIdentity()`, das Passwort mittels `setCredential()`, die Überprüfung wird mittels `authenticate()` gestartet [31]. Eine Verschlüsselung des Passworts ist auch hier möglich.

¹⁶ http://de.wikipedia.org/wiki/Mail_Transfer_Agent

7 Web Services

Einsatzmöglichkeit. Viele große Web Service-Anbieter stellen eigene APIs zur Einbindung der Dienste auf die eigene Seite an. Im Falle des Web Logs lassen sich somit Funktionen wie Büchersuche zum Beitragsthema, Feederstellung, Kalenderanbindung an Google Calendar, Internet-Suche oder Twittern leicht implementieren ohne nähere Kenntnisse der Funktionsweise zu besitzen.

7.1 Feeds

Feeds können automatisch auf einer Seite gefunden werden mittels `findFeeds()`. Unterstützt werden die beiden bekanntesten Feed-Arten RSS und Atom. Das Einlesen eines Feeds ist denkbar einfach:

```
$feed = new Zend_Feed_Rss(
    'http://www.eine-seite.de/feeds/index.rss');
```

Die Methodennamen zur Ausgabe von Autoren, Titeln etc. entsprechen den Parameternamen (`author()`, `title()` etc.)

Auch für die Erzeugung von Feeds sind keine XML-Kenntnisse erforderlich, über `send()` lassen sich entsprechende Daten in einen Feed umwandeln, mittels `importArray()` wird das Feed-Objekt erstellt [32].

7.2 Amazon

Um Zugriff auf die Product Advertising API¹⁷ von Amazon zu bekommen, benötigt man einen speziellen Amazon-Account, welcher kostenlos eingerichtet werden kann. Die Verbindung zu Amazon wird folgendermaßen aufgebaut:

```
$amazonZugriff = new
Zend_Service_Amazon($key,$land,$secretKey);
```

Die beiden Keys bekommt man von Amazon bei der Einrichtung des Accounts, über `$land` lässt sich das Zugriffsland bestimmen (für amerikanische Bücher idealerweise „US“).

Über `itemSearch()` wird die Abfrage erstellt, dabei werden alle Suchoptionen (Kategorie, Autor, Titel, Schlüsselwörter) und der Informationsumfang der Antwort (ResponseGroup) angegeben [33]. Die ResponseGroup „Large“ beinhaltet unter anderem ein Cover, durchschnittliche Bewertungen, Preise und Rezensionen, sodass theoretisch eine Amazonseite inhaltstechnisch gesehen nachgestaltet werden kann.

¹⁷ <https://partner.net.amazon.de/gp/advertising/api/detail/main.html>

7.3 Google Services

Über die Google Web Service API¹⁸ hat man Zugriff auf nahezu alle Google-Dienste, ausgenommen der Suche. Somit lassen sich einbinden [34]:

- Google: Apps, Analytics, Blogger, Base, Book Search, Calendar, Code Search, Earth, Spreadsheets, Notebook, Documents
- Youtube
- Picasa
- AdSense
- AdWords

Der Zugriff auf die Dienste erfolgt im Hintergrund über Feeds, welche die Kommunikation zwischen Client und Server übernehmen [35]. Teilweise ist eine Authentifizierung bei Google notwendig, um die Dienste nutzen zu können. Dies kann entweder über die normalen Zugangsdaten des eigenen Google-Accounts abgewickelt werden oder über die Einbindung einer Google-Authentifizierungsseite (Google Account Authentication¹⁹).

8 Internationalisierung

Einsatzmöglichkeit. Das Internet ist ein globales Netzwerk, sodass eine dem jeweiligen Zugriffsland angepasste Website/Anwendung vielerlei Vorteile bringen kann. Besonders bei international agierenden Unternehmen bietet sich eine Umsetzung an. Das Web-Log könnte beispielsweise mehrsprachig mit unterschiedlichen Währungssystemen im Büchershop o.Ä. aufgebaut werden.

8.1 Anpassungen

Anpassungen bezüglich Sprache, Datumsformatierung, Zahlen- und Währungsformaten werden mit Zend_Locale umgesetzt. Hierfür kann ein Locale gesetzt werden oder direkt aus den Browser-Einstellungen ausgelesen werden [36]. Dieses Locale definiert die verwendete Sprache sowie die Region, was bspw. bei Österreich sinnvoll ist:

```
$locale = new Zend_Locale('de_AT');
```

Mehrsprachigkeit lässt sich über Zend_Translate umsetzen [37]. Hier können Übersetzungstabellen genutzt werden, welche auch komplette Texte mit der jeweiligen Übersetzung beinhalten können. Diese Übersetzungstabellen sollten in Form einer .csv-Datei vorliegen. Dabei ist es möglich, mehrere Sprachen anzumelden.

¹⁸ <http://code.google.com/intl/de-DE/more/>

¹⁹ <http://code.google.com/intl/de-DE/apis/accounts/>

8.2 Umrechnungen

Viele Regionen besitzen spezifische Maßsysteme, was automatisierte Umrechnungen passend zur ausgewählten Region (über Browser) sinnvoll erscheinen lässt. Das ZF stellt dafür einige Werkzeuge bereit:

- Zend_Measure: dient der Umrechnung von Maßsystemen wie bspw. „teaspoon“ → „Gramm“[11]
- Zend_Currency: unterstützt Währungsumwandlungen [11]
- Zend_Date: bietet die Möglichkeit, einerseits die Syntax einer Datumsausgabe zu verändern, andererseits ist das Rechnen mit Datumsangaben möglich. Des Weiteren können verschiedene Zeitzonen verwaltet werden. [11]

9 Caching

Einsatzmöglichkeit. Zur Optimierung der Performance einer Seite ist Caching äußerst nützlich. Zielt das Web-Log auf eine stetig steigende, große Nutzerzahl ab, so ist Caching sinnvoll, um die Auslastung und somit die Antwortzeit des Systems gering zu halten.

9.1 Allgemeines

Das Caching im ZF besteht aus einem Frontend und einem Backend. Das Frontend dient der Aufnahme und Ausgabe der zu cachenden Daten, im Backend wird der Cache (Zwischenspeicher) gehalten. Sowohl für Frontend und Backend gibt es verschiedene Typen, welche unterschiedliche Anwendungsarten erlauben. Der Benutzer kann frei aus diesen Typen wählen und somit die Performance durch eine maßgeschneiderte Lösung zusätzlich verbessern.

Der Cache-Zugriff erfolgt über IDs, alle gecacheten Daten werden serialisiert. Zum Starten des Caching wird die `start()`-Methode verwendet, welche eigenständig das Vorhandensein von Daten im Cache überprüft, sodass Redundanzen unterbunden werden.

Auf Daten im Cache kann auch manuell zugegriffen werden.

9.2 Frontend

Es werden fünf verschiedene Frontends zur Verfügung gestellt [38]:

- Output: cached alle Ausgaben, `end()` notwendig
- Page: gesamte Website, endet automatisch mit Ende der Website
- Function: Funktionsergebnisse
- Class: Objekte und statische Methodenaufrufe
- File: eingelesene Dateien als Objekte im Cache

Wie bereits erwähnt, lässt sich das Frontend auch manuell konfigurieren. Die Lifetime von Inhalten ist angebar, auch der Garbage Collector lässt sich konfigurieren. Über Zend_Log ist ein Logging des Caches möglich. Die Serialisierung der zu cachenden Daten erfolgt automatisch. Durch die Auswahl geeigneter Frontends lassen sich einzelne Seitenteile cachen, diese Differenzierungsmöglichkeit bietet Potenzial für Performance-Steigerungen [38].

9.3 Backend

Wie für das Frontend existieren auch für das Backend mehrere Typen, welche ausgewählt werden können, nachfolgend die Wichtigsten:

- File: schreibt den Cache in Dateien auf der Festplatte, ein Locking der gerade verwendeten Daten ist möglich
- Memcached: schreibt den Cache in den Arbeitsspeicher für schnelle Antwortzeiten, Cache kann serverübergreifend genutzt werden
- APC oder XCache: Speicherung im Shared Memory, cached Bytecode
- SQLite: Abspeicherung in SQLite-Datenbank

Für die Produkte Zend Server und Zend Platform gibt es eigens angepasste Backends, die für die normale Nutzung mit Standardsoftware allerdings keine weiteren Vorteile bringen [39].

10 Aktuelles / Ausblick

Das ZF ist Gewinner des Bossie Awards²⁰ 2010 als „Best Open Source Development Software“²¹.

Die Roadmap für das ZF 2.0 steht [40]. Allgemein gesagt soll die Bedienung vereinfacht und Komplexität (bspw. bei MVC) reduziert werden. Zu diesem Zweck sind vereinheitlichte Konstruktoren in Arbeit, die setOptions () -Methode soll zur Standardmethode für die Konfiguration erhoben werden. Zusätzlich sollen Konfigurationen selbst gebündelt werden in Zend_Options, was eine weitere Reduktion der Redundanzen bedeutet. Über die Entschlackung des MVC-Systems gibt es noch keine konkreten Aussagen.

Ein weiteres Thema für zukünftige ZF-Versionen ist die Cloud-Anbindung. Hier wird bereits mit Microsoft (Azure), Amazon (S3) und IBM zusammen an einer Lösung gearbeitet [41].

²⁰ http://en.wikipedia.org/wiki/Infoworld_Bossie_Awards

²¹ <http://www zend.com/de/company/news/news-links/zend-framework-gewinnt-infoworld-bossie-award>

Das Thema Mobility ist auch an Zend nicht vorbei gegangen, allerdings lassen sich noch keine konkreten Aussagen über den Entwicklungsstand bezüglich des Mobile Apps Development treffen.

References

1. Zend Framework: Documentation: Ein Projekt erstellen – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.tool.project.create-a-project.html>
2. Glue vs. Full Stack, <http://littlehart.net/atthekeyboard/2007/05/30/glue-vs-full-stack>
3. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.16
4. Mörke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 25
5. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.17
6. Zend Technologies: PHP and Zend Engine - Zend Contributions to PHP, <http://www zend com/de/community/php/>
7. Zend Technologies: Alle Produkte auf einen Blick, <http://www zend com/de/products/>
8. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.18
9. dev.Talk: Propel ORM in das Zend Framework integrieren, <http://www.dev-talk.info/prophel-orm-in-das-zend-framework-integrieren/>
10. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.29
11. Zend Framework: Documentation: Zend Framework Komponenten – 1.5 – Zend Framework Manual, <http://framework.zend.com/manual/1.5/de/requirements zendcomponents.html>
12. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.35
13. Zend Framework: Documentation: Der Front-Controller – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.controller.front.html>
14. Padilla, A.: Beginning Zend Framework. Apress (Springer-Verlag), New York (2009), S. 62 ff.
15. Mörke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 31 ff.
16. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.42, Abb. 2.4
17. Mörke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 36
18. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.33
19. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.137
20. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.51 ff.
21. Mörke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 194
22. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.67 ff.

23. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 73
24. Zend Framework: Documentation: Zend_Db_Profiler – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.db.profiler.html>
25. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 158
26. Allen, R., Lo, N., Brown, S.: Zend Framework im Einsatz. Carl-Hanser-Verlag, München (2009), S.189
27. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 175
28. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 279 ff.
29. Zend Framework: Documentation: Zend_Mail – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.mail.html>
30. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 116 ff.
31. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 124 ff.
32. Zend Framework: Documentation: Zend_Feed – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.feed.htm>
33. Zend Framework: Documentation: Zend_Service_Amazon – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.service.amazon.html>
34. Wikipedia, the free encyclopedia: Google Code, http://en.wikipedia.org/wiki/Google_Code#Google_APIs
35. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 274
36. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 370
37. Mörhrke, C.: Zend Framework: Das Entwickler-Handbuch. Galileo-Press, Bonn (2008), S. 381 ff.
38. Zend Framework: Documentation: Zend_Cache_Frontend – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.cache.frontends.html>
39. Zend Framework: Documentation: Zend_Cache_Backend – Zend Framework Manual, <http://framework.zend.com/manual/de/zend.cache.backends.html>
40. Zend Framework Wiki: Zend Framework 2.0 Roadmap – Zend Framework 2.0, <http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap>
41. Heise Developer: Zend, Microsoft und IBM kooperieren bei Simple Cloud API für PHP-Anwendungen, <http://www.heise.de/developer/meldung/Zend-Microsoft-und-IBM-kooperieren-bei-Simple-Cloud-API-fuer-PHP-Anwendungen-789724.html>

Django

Thomas Heinz

Otto-Friedrich Universität Bamberg

Abstract. Diese Arbeit soll eine Einführung in das auf Python basierte Webframework Django geben. Zunächst werden die Philosophien die hinter dem Projekt stehen erklärt, anschließend ein grober Überblick über die Architektur und die Bestandteile des Frameworks gegeben. Zuletzt wird anhand eines Beispiels die Entwicklung einer Webanwendung dargestellt. Dem Leser soll somit vermittelt werden, welche Möglichkeiten Django Nutzern zur Hand gibt, und nach welchen Prinzipien die Entwicklung einer Anwendung vorgehen sollte. Die Beispielanwendung soll die Mächtigkeit ausgewählter, mitgelieferten Software Bibliotheken und die einfache Verwendung dieser demonstrieren.

1 Einführung in Django

Django ist ein in Python geschriebenes „Web Application Framework“. Es besteht aus einer Ansammlung von Python Bibliotheken, die den Kern des Frameworks bilden, und einer Reihe von integrierbaren Anwendungen und Werkzeugen, welche die Entwicklung erleichtern und beschleunigen. Es umfasst alle Bestandteile die nötig sind um komplexe Webanwendungen zu entwickeln. Es wird als „full stack framework“ bezeichnet, da es verschiedene Bibliotheken und Werkzeuge zu einer zusammenhängenden Software Ansammlung zusammenfasst, die von Entwicklern genutzt werden können.

Die Abgrenzung zu anderen Web Frameworks besteht darin, dass der Fokus eindeutig auf der agilen Entwicklung von Web Anwendungen liegt. Es muss nicht auf die sogenannten „best practices“ der Web Entwicklung verzichtet werden. Django Entwicklerteams bestehen in der Regel aus einer kleinen Gruppe von Entwicklern, die unabhängig von den Designern und Grafikern arbeiten können.

Der Quellcode zum Projekt ist der Öffentlichkeit zugänglich und steht unter einer BSD Lizenz und wird beständig von einer großen Community erweitert.

2 Entstehungsgeschichte und Entwicklung

Django entstand aus den Bedürfnissen einer Gruppe von Web Entwicklern. Diese arbeiteten im Herbst 2003 an einem online Zeitungsprojekt namens „World Online“. Dieses Projekt verlangte nach sehr kurzen Entwicklungszeiten für die zu erstellenden Web Anwendungen. Aus diesem Grund sahen sie sich gezwungen sich von der bisher

benutzten PHP Technologie abzuwenden und sich nach neuen Möglichkeiten umzusehen. Jedoch fanden sie keine der vorhandenen Frameworks oder Bibliotheken ihren Ansprüchen genügend. So begannen sie eine Reihe von Web Anwendungen in Python zu entwickeln, aus deren Grundgerüsten sich später Django entwickelte.

Das Framework wurde über die folgenden zwei Jahre stetig weiterentwickelt und optimiert. Im Sommer 2005 entschied die Gruppe des „World Online“ Projektes den Quellcode als Open-Source Software der Öffentlichkeit zugänglich zu machen. Ihr Hauptbeweggrund war es, der Open-Source Gemeinde etwas zurückzugeben, da Django ohne deren zahlreichen Projekte wie zum Beispiel Apache, Python und PostgreSQL nicht entstehen hätte können.

Im Juni 2008 wurden die Marke „Django“ und die Rechte am Projekt an die neu gegründete, gemeinnützige „Django Software Foundation“ übertragen. Die sich seither um das Wohl und die Weiterentwicklung des Projektes kümmert.

Benannt wurde das Framework nach „Django Reinhardt“, einem Jazz Gitarristen. Er wurde 1910 in Belgien geboren, starb 1953 in Paris und galt als Begründer des europäischen Jazz.

Zum Zeitpunkt des Entstehens dieser Arbeit liegt Django in der stabilen Version „1.3“ vor. Große Meilensteine (Version „2.0“, „3.0“) sind nur im Abstand einiger Jahre angedacht. In untergeordneten Veröffentlichungen („1.3“, „1.4“, ...), die etwa alle neun Monate erscheinen, sollen neue Funktionalitäten hinzugefügt werden und alte entfallen. Die Umstellung von entwickelten Anwendungen auf eine neue Version sollte jedoch stets mit geringfügigen Quellcode Anpassungen möglich sein¹.

3 Philosophien

Wie bereits erwähnt liegt der Ursprung des Frameworks in der Journalisten Branche. Dies beeinflusste die Entwicklung stark. Der offizielle Slogan des Projekts lautet: „The Web framework for perfectionists with deadlines“. Dies ist auf die strikten Fristen zurückzuführen, mit denen die Entwickler bei der Entwicklung ihrer Web Anwendungen zu kämpfen hatten. Die Wahl der Programmiersprache fiel auf Python, welche für ihre Möglichkeiten der schnellen Software Entwicklung bekannt ist und es sich zum Ziel gesetzt hat möglichst einfach und leserlich zu sein. Trotz dieser zeitlichen Engpässe wollten die Erfinder sich nicht mit behilf Lösungen, im Bezug auf Web Technologien zufrieden geben. Alle Software Bibliotheken wurden deswegen eigens für Django neu entwickelt, anstatt bereits vorhandene einzusetzen. Die Entwickler bezeichnen sich selbst als wählerisch bzw. als Perfektionisten².

Die Wahl von Python als Sprache setzt sich konsequent durch das gesamte Projekt durch. Alle benötigten Anwendungen, Werkzeuge und sogar Konfigurationsdateien sind in Python implementiert. Dies ermöglicht größte Flexibilität und vereinfacht den Entwicklungsprozess.

Das Framework ist so gestaltet, dass sich sämtliche Komponenten durch Alternativen oder eigens entwickelte Module auswechseln lassen. Django fällt somit

¹ <https://docs.djangoproject.com/en/1.3/internals/release-process/>, letzter Zugriff: 10.05.2011

² <https://code.djangoproject.com/browser/django/trunk/docs/faq/general.txt>, letzter Zugriff: 10.05.2011

auch in die Kategorie der in Fachkreisen als „Glue On“ bekannten Frameworks. Als Beispiel kann man hier die „Templating“-Sprache nennen, die mitgeliefert wird. Die Entwickler nennen explizit Alternativen zu ihr und erklären die Einbindung in das System.

Für die Anwendungsentwicklung mit Django wurden Design-Philosophien und Richtlinien zusammengetragen. Diese können in der offiziellen Dokumentation³, oder in [1] Seite 275 nachgelesen werden.

4 Architektur

Django folgt dem für Webframeworks typischen MVC-Architektur. Allerdings mit kleineren Abwandlungen des klassischen Modells, auf die im Folgenden näher eingegangen wird. So bevorzugen die Entwickler auch den Begriff MVT (Model View Template). Eine vom Klienten ankommende Anfrage, wird vom Webserver verarbeitet, an den URL Mapper weitergegeben. In diesem ist mittels regulärer Ausdrücke festgelegt, welche entsprechende Funktionen der Views aufgerufen werden. Diese Funktionen verarbeiten die verschiedenen für die Anfrage benötigten Komponenten. Die Darstellung der entsprechenden Antwort geschieht über Templates.

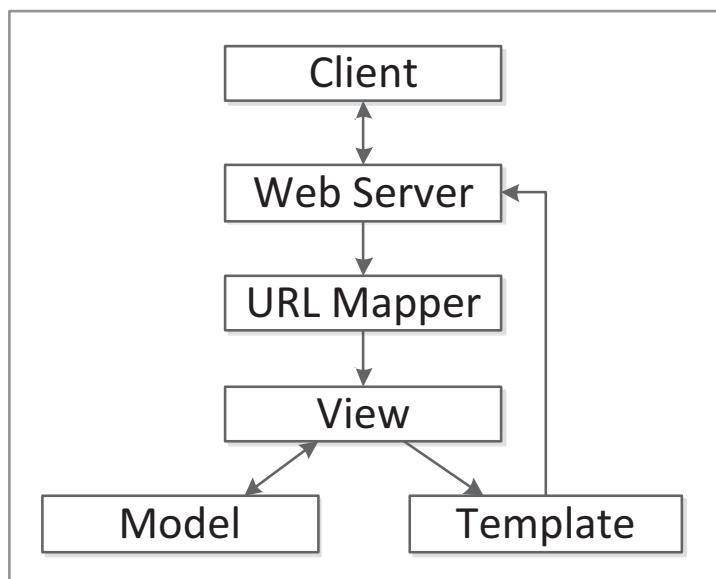


Abbildung. 1. Djangos MVT Architektur.

³ <https://docs.djangoproject.com/en/1.3/misc/design-philosophies/>, letzter Zugriff: 10.05.2011

Modelle dienen der Abstraktion, der von der Anwendung benötigten Daten. Im Allgemeinen werden sie in der „models.py“ Datei festgehalten. Durch den Objekt-Relationalen-Mapper werden die Daten, in entsprechende Tabellen in der Datenbank übertragen. Es ist daher nicht nötig mit SQL Befehlen auf die Datenbank zuzugreifen.

Die klassische View des MVC Modells ist also die Funktion, die vom URL-Mapper aufgerufen wird, da sie beschreibt welche Daten für die Anfrage relevant sind. Templates beschreiben, wie diese Daten für den Nutzer repräsentiert werden. Der Controller ist im Grunde die Funktionalität des Frameworks selbst. Da diese dafür verantwortlich ist welche Funktionen für welche Requests verarbeitet werden.

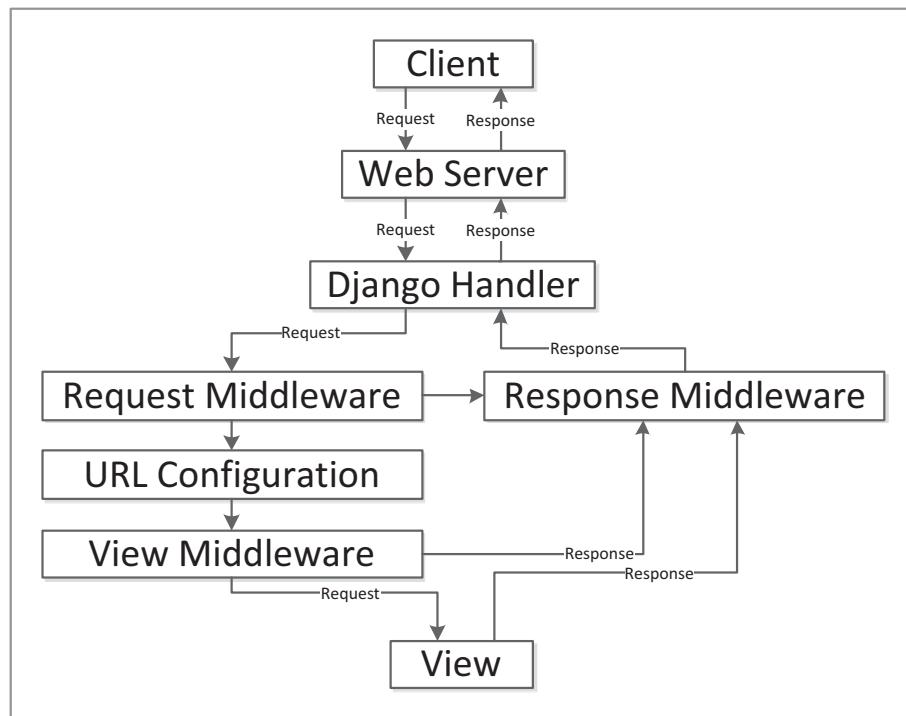


Abbildung. 2. Ablauf einer Anfrage und deren Antwort.

Wie in Abbildung 2 zu sehen ist, durchläuft eine typische Anfrage jedoch auch noch unterschiedlich Middleware Module und die entsprechende Antwort wird von den installierten Context Modulen mit Informationen bereichert.

5 Bestandteile

Django ist intern in verschiedene Bereiche unterteilt. Im Folgenden gebe ich eine Übersicht über die verschiedenen Segmente und Beschreibe kurz deren Verwendungszweck, bevor ich näher auf die Kern- und Optionalen Entwicklungsbibliotheken eingehe.

Tabelle 1. Gliederung des Django Projektes in seine verschiedene Bereiche mit entsprechenden Erklärungen.

Sektion	Beschreibung
Binaries	Anwendungen, die das Erstellen und verwalten von Projekten und Anwendungen erleichtern.
Configuration	Beinhaltet Standard Konfigurationen, für unter anderem Projekte und Anwendungen.
Contrib	Umfasst Standard-Implementationen häufig verwendeter Software Patterns. Die jedoch optional einsetzbar sind.
Core	Standard-Implementationen häufig verwendeter Software Patterns, die in praktisch jeder Web Anwendung zum Einsatz kommen.
Database	Datenbank Backends, Modelle und Dienstprogramme.
Dispatch	Signal Dispatcher, der es erlaubt Anwendungen zu benachrichtigen, wenn bestimmte Aktionen eintreten.
Forms	Werkzeuge zur automatischen Formular Erzeugung und deren Verarbeitung.
HTTP	Ein HTTP Parser.
Middleware	Verschiedene Gerüste, die zwischen die ankommenden Anfragen und Antworten an den Klienten zwischen geschaltet werden können.
Template	Djangos hauseigenes Templating System.
Template Tags	Zusätzliche Auszeichnungsschlagworte für das Templating System.
Tests	Bibliotheken, die das Testen von Django Anwendungen ermöglichen.
Utilities	Eine große Anzahl von Bibliotheken, die nicht speziell die Entwicklung von Webanwendungen adressieren, aber von Nutzen sein können.
Views	Module zur Erstellung der Präsentationsschicht einer Webanwendung.

2.1 Core Module

Wie aus Tabelle 1 bereits hervorgeht stellen die sogenannten „core“ Module die Grundlage fast aller in Django entwickelten Web Anwendungen dar. Aus diesem Grund möchte ich eine kurze Übersicht über ihre Funktionsweisen geben.

Tabelle 2. Djangos Core Module mit Beschreibungen.

Modulname	Beschreibung
Cache	Djangos Caching Module, welches verschiedene Möglichkeiten bietet, die Resultate rechenintensiver Vorgänge zu speichern.
Files	Ermöglicht die Handhabung von Dateien.
Handlers	Handhabt die Kommunikation zwischen dem Web Server und den Anwendungen.
Mail	Dieses Modul vereinfacht das Versenden von Emails und stellt Unterstützung für Plattformen die kein SMTP unterstützen bereit.
Management	Stellt Projekt- und Anwendungs-Verwaltungs Befehle bereit, welche auf einfache Weise erweitert werden können
Serializers	Mit diesen Modulen ist es möglich Django Objekte in andere Formate zu „übersetzen“. Es ist beispielsweise ein JSON und ein XML Übersetzer vorhanden.
Servers	Django bringt einen eigenen kleinen Web Server mit sich, der allerdings nur für Testzwecke eingesetzt werden sollte.
Context Processors	Die Kontext-Prozessoren ermöglichen die Weitergabe zusätzlicher Werte an die Templates.
Exceptions	Dieses Modul enthält die Definitionen für Django spezifische Exceptions, die vom Framework oder Benutzer geworfen werden können.
Paginator	Ermöglicht die einfache Aufteilung von Daten auf mehrere Webseiten, die durch generierte Links miteinander verknüpft sind.
URL Resolvers	Konvertiert angeforderte URLs zu den zugehörigen Callback Funktionen der Präsentationsschicht.
Validators	Stellt Funktionen zur Überprüfung von URLs, Emails und anderen Dingen zur Verfügung.
XHeaders	Gibt Möglichkeiten zur Hand, die sogenannten X-Header von HTML Dateien auszulesen.

2.1 Contrib Module

Neben den Core Modulen stellt Django einige Module bereit, die von externen Entwicklern beigesteuert wurden.

Tabelle 3. Djangos Contrib Module mit Beschreibungen.

Modulname	Beschreibung
Admin & Admin Documentation	Ein Automatisch Administrationsinterface, welches die Metadaten der Modellschicht und ein Interface zur Erstellung und Manipulation bereitstellt.
Auth	Authentifikations-System. Es verwaltet Benutzer, Gruppen, Rechte und Cookie basierte Nutzer-Sessions.
Comments	Einfaches und auf leichte Weise erweiterbares Kommentar-System.
Contenttypes	Hält die Inhalte der Präsentationsschicht aller installierten Applikationen fest.
CSRF	„Cross-Site Request Forgery“ Sicherheits-Modul.
Databrowse	Erstellt dynamisch generierte Webseiten, basierend auf den Daten einer Anwendung.
Flatpages	Ermöglicht die einfache Handhabung von Seiten mit reinen HTML-Inhalten.
Formtools	Eine Abstraktionsebene für die Formular Werkzeuge.
Gis	Eine Geoinformationssystem-Erweiterung für Django.
Humanize	Konvertiert Daten in von Menschen leichter lesbare Darstellungsformen.
Localflavor	Beinhaltet Funktionen, die spezielle Anpassungen der Darstellung von Daten an verschiedene Kulturen vornehmen können.
Markup	Stellt Template-Filter für verschiedene Auszeichnungssprachen bereit.
Messages	Benachrichtigungsfunktionen für Nutzer.
Redirects	Beinhaltet Webseiten Weiterleitungs-Funktionen.
Sessions	Unterstützung für Cookie-basierte anonyme Sessions
Sitemaps	Automatisiert die Kreation von Sitemap-XML-Dateien.
Sites	Assoziiert Objekte mit bestimmten Websites und ermöglicht ein einfaches Domain-Management.
Staticfiles	Handhabt die Verwaltung von statischen Dateien.
Syndication	RSS- und Atom-Syndikation-Feed Generierung.
Webdesign	Stellt Funktionen bereit, die speziell an Web Designer gerichtet sind.

6 Beispielprojekt

Um den typischen Ablauf der Entwicklung mit Django zu verdeutlichen, wird in diesem Kapitel eine einfache „To-do-Listen“ Anwendung gezeigt. Auf die Installation und Einrichtung von Django wird an dieser Stelle nicht eingegangen. Eine Anleitung ist unter anderem auf der Projekt Webseite⁴ zu finden und in [2] Seiten 49-55.

6.1 Projekt und Applikation erstellen

Zur Entwicklung einer Webanwendung stellt Django ein ausführbares Skript zur Verfügung, welches grundlegende Einstellungen festlegt und standardisierte Projekt erstellt. Ein neues Projekt, namens „example_project“, wird über die Kommandozeile mittels des Befehls

```
django-admin.py startproject example_project
```

erzeugt. Im Aktuellen Verzeichnis wird das Projekt mit Konfigurationsskripte für allgemeine Einstellungen („settings.py“) und zum Design der URLs („urls.py“) angelegt. Die Datei „manage.py“ ist ein weiteres Skript zur Verwaltung des Projektes. Mit ihm lässt sich auf folgende Weise die Grundstruktur der „To-Do-Liste“ Anwendung erzeugen.

```
manage.py startapp todo_app
```

Das Projekt muss anschließend mithilfe der „settings.py“-Datei konfiguriert werden. Unter anderem lassen sich dort Datenbank Anbindungen, verwendete Komponenten und Pfade zu statischen Dateien festlegen.

6.2 Das Anwendungsmodell

Das Modell enthält die Informationen über die Struktur der Daten, welche die Anwendung nutzen soll. Außerdem dienen Modelle als Abstraktionsebene zu den Datenbanken. Mithilfe von Objektrelationalen Abbildungen können die in ihnen niedergeschriebenen Python Klassen in SQL Befehle transformiert und auf die Datenbanken angewendet werden. Instanzen der Klassen entsprechen hingegen den Einträgen der Datenbanken.

Für unsere „To-Do“ Anwendung möchten wir eine Datenstruktur, welche einen Aufgabennamen, ein („erledigen bis“) Datum und einen Status besitzt. Ein entsprechendes Modell würde wie folgt aussehen:

⁴ <https://docs.djangoproject.com/en/1.3/topics/install/>, letzter Zugriff: 10.05.2011

```

TASK_STATUS_CHOICES = (
    ('Running', 'Running'),
    ('FINISHED', 'Finished'),
    ('ABORTED', 'Aborted'),
)

class Task(models.Model):
    """
    Task object model.
    """
    # Task title with a maximal length of 30 chars
    title = models.CharField(max_length=30)
    # Task end date, can be left blank
    ends_on = models.DateTimeField(null=True,
                                   blank=True)
    # Task
    status = models.CharField(max_length=30,
                             choices=TASK_STATUS_CHOICES)

    def __unicode__(self):
        """ Returns the unicode representation"""
        return self.title

```

Der SQL Quelltext zur Tabellen Erzeugung lässt auf der Kommandozeile mittels

```
manage.py sql todo_app
```

ausgeben. In diesem Fall, verfügt das Modell über keinen eigens festgelegten Primärschlüssel. Dieser wird aber von der Datenbank benötigt. Deshalb wird ein automatisch weiteres Feld vom Datentyp Integer zum SQL Statement hinzugefügt und zum Primärschlüssel deklariert.

```

TODO SQL STATEMENT
BEGIN;
CREATE TABLE „todo_app_task“ (
    „id“ integer NOT NULL PRIMARY KEY
    „title“ varchar(30) NOT NULL,
    „ends_on“ datetime,
    „status“ varchar(30) NOT NULL
);

```

Mithilfe des Befehls

```
manage.py syncdb
```

lässt sich das SQL Statement auf die Datenbank anwenden, und somit werden die entsprechenden Tablellen erzeugen.

6.3 Die View der Anwendung

Um festzulegen, welche Daten Nutzern der Anwendung sichtbar sind, müssen Views erstellt werden. Diese verarbeiten HTTP Requests. Eine typische View Funktion wählt Einträge aus der Datenbank aus, verarbeiten diese und liefert Ergebnisse über ein HTTP Response Objekt an den Klienten zurück.

Zur Ansicht eines über die „task_id“ ermittelbaren Tabellen-Eintrags, wird im Folgenden eine Methode definiert, die den entsprechenden Task ermittelt und über die „render_to_response“ aufruft. Diese Funktion verknüpft die Daten mit einem Template, rendert dieses und liefert diese Daten an den Klienten.

```
from django.shortcuts import render, redirect,
get_object_or_404
from todo_app.models import *

def view_task(request, task_id):
    # Get task with 'task_id' or return 404 error.
    task = get_object_or_404(Task, pk=task_id)
    # Render template
    template = 'todo/view.html'
    context = {
        'task_object' : task}
    return render(request, template, context)
```

6.4 Das URL Mapping

Damit die angelegte View erreichbar ist, muss die „url.py“ Datei bearbeitet werden. Mit ihrer Hilfe können durch reguläre Ausdrücke URLs gestaltet werden, welche dann auf die, sich in der View der Applikation befindenden Funktionen ge-„mapped“ werden können.

```
from django.conf.urls.defaults import *

# Enable the admin sites:
from django.contrib import admin
admin.autodiscover()

urlpatterns = patterns('',
    # -- Task urls --
    url(r'^view/(\d+)/$', 'todo_app.views.view_task'),
    # ----- Admin urls -----
    url(r'^admin/', include(admin.site.urls)),
    url(r'^admin/doc/',
        include('django.contrib.admindocs.urls')),)
```

Wäre das die Anwendung unter der Domain „todo.example.com“ erreichbar, so würde über „todo.example.com/view/3“ die Funktion „view_task“ aufgerufen werden und ihr wird neben „request“ Objekt die Zahl 3 als zweiter Parameter übergeben.

6.5 Der Administrator Bereich

Django verfügt über einen automatisiert erstellten Administrationsbereich. In diesem lassen sich alle, bei ihm registrierten, Klassen über CRUD (Quelle) System verwalten. Ist die Administrationsanwendung installiert und der mitgelieferte Webserver über den Befehl

```
manage.py runserver
```

gestartet, lässt er sich in unserem Beispielprojekt, im Falle der Standartkonfiguration, über die URL „127.0.0.1/admin/“ erreichen. Dort lassen sich nach Belieben „Tasks“ anlegen, betrachten, bearbeiten und löschen.



Abbildung. 3. Die Oberfläche der Adminanwendung mit registriertem Task Modell.

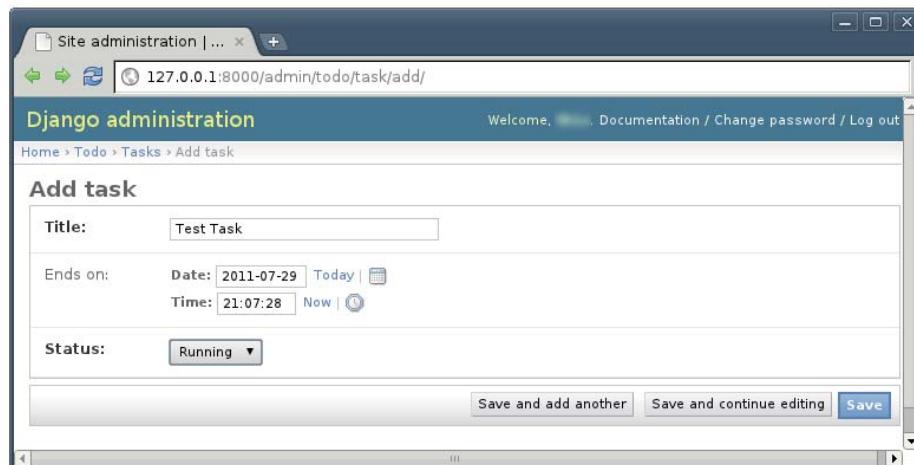


Abbildung. 4. Erstellung eines Tasks über die Admin-Oberfläche.

6.6 Das Template System

Damit die zuvor in der View angelegte Funktion etwas darstellen kann, fehlt noch ein Template. Es kümmert sich darum wie, bzw. in welcher Form, dem Nutzer die ausgegebenen Daten präsentiert werden. Django bringt ein eigenes Template System mit sich, das durch seine Simplizität besticht. Templates werden in einem zuvor festgelegten Verzeichnis gespeichert und entsprechend der Namen ihrer dazugehörigen Funktionen aufgerufen, wenn nichts anderes festgelegt wurde. Sie bestehen aus der HTML-Auszeichnungssprache, angereichert mit eingebauten Template Tags und Filtern. Außerdem können sie mit selbst implementierten Tags erweitert werden.

Nachdem über die Admin-Oberfläche „Task“ Einträge erstellt wurden können diese über die entsprechenden URLs betrachtet werden. Mit folgendem Template kann beispielsweise ein Task visualisiert werden:

```
{% if task_object %}
    <h1>Task: {{ task_object.title }}</h1>
    <p>Status: {{ task_object.status }}</p>
    <p>Finish until: {{ task_object.ends_on }}</p>
{% else %}
    This should not happen.
{% endif %}
```

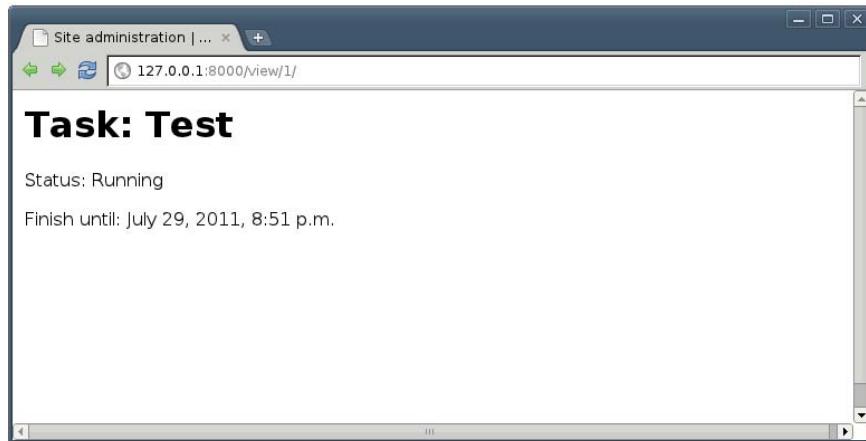


Abbildung 5. Darstellung des zuvor angelegten Tasks mit der „id“ 1.

Eine Übersicht über Djangos Template Tags wird in der offiziellen Dokumentation gegeben⁵. Eine ausführliche Anleitung liefert [3].

7 Fazit

Das Django Projekt ist das wohl bekannteste, der inzwischen beachtlichen Anzahl an Python Webframeworks⁶. Es besticht durch seinen großen Funktionsumfang, einer aktive Community und auch durch eine sehr gute Dokumentation. Im Vergleich zu, auf anderen Sprachen basierenden, Webframeworks hebt sich von seiner Konkurrenz durch die extrem schnelle Anwendungsentwicklung mit klarem, simplem Design ab. Zahlreiche erfolgreiche Praxiseinsätze⁷ beweisen die Stabilität und die Reife des Produkts.

References

1. Moore, D., Budd, R., Wright, W.: Professional Python Frameworks: Web 2.0 Programming with Django and Turbogears. John Wiley & Sons, (2007)
2. Kaplan-Moss, J.: The Definitive Guide to Django: Web Development Done Right. Apres, (2007)
3. Newman, Scott: Django 1.0 Template Development. Packt Publishing, (2008)
4. Bennett, J.: Practical Django Projects. Apres, (2008)
5. McGaw , J.: Beginning Django E-Commerce. Apres, (2009)

⁵ <https://docs.djangoproject.com/en/1.3/ref/templates/builtins/>, letzter Zugriff: 10.05.2011

⁶ <http://wiki.python.org/moin/WebFrameworks/>, letzter Zugriff: 14.06.2011

⁷ <http://www.djangosites.org/>, letzter Zugriff: 29.07.2011

ASP.NET
Eine Einführung in Version 4.0

Fabian Biberger

Otto-Friedrich Universität Bamberg

1 Grundlagen von ASP.NET

Es gibt die Legende, dass die erste Version von ASP.NET an einem Wochenende in den Winterferien 1998 von Scott Guthrie entwickelt wurde. Er selbst beziffert die Entwicklungszeit auf ungefähr eine Woche. Aus diesem Wochenprojekt, das in JAVA geschrieben war, entwickelte sich über einige Jahre hinweg die Microsoft Webentwicklungsplattform ASP.NET. Aus dem Namen kann man gut erkennen, das JAVA über die Zeit verloren ging, und das Projekt letztendlich mit C# auf der .NET-Plattform umgesetzt wurde (vgl. [19, S.4], [5]). Microsoft benötigte diese Umstellung auf eine modernere Plattform, um den Herausforderungen des schnellen Wandels der Internettechnologien gewachsen zu sein. In der vorliegenden Arbeit soll diese vorgestellt und die grundlegenden Konzepte sollen erklärt werden. Ziel ist es, einen grundlegenden Einstieg in ASP.NET zu geben und mit Hilfe der angegebenen weiterführenden Literatur eine tiefere Einarbeitung in die Technologie zu ermöglichen.

Um die im Folgenden verwendeten Versionsnummern besser einordnen zu können, gibt es zunächst eine kurze Übersicht über die Entwicklungsgeschichte von ASP.NET. Im Jahr 2002 erschien wie oben schon angemerkt die Version 1.0, zusammen mit dem komplett neuen .NET Framework. Es folgten die Evolutionsstufen 1.1, 2.0, 3.5, 3.5 Service Pack 1 und aktuell Version 4.0. Diese unregelmäßigen Versionsnummern geben die Versionierung parallel zur Hauptentwicklung von .NET wieder - die wichtigsten Weiterentwicklungen erhielt ASP.NET mit Version 2.0 (neue Steuerelemente, AJAX, Webparts, Benutzerverwaltung etc.) und mit der Version 3.5, in der das MVC-Pattern mit in den Hauptzweig von ASP.NET integriert wurde (vgl. [19, S.5], [1, S.811]).

Das Einführungskapitel stellt einige grundlegende Aspekte zu ASP.NET vor. Zum einen wird das .NET-Framework als Ganzes kurz vorgestellt, zum anderen werden typische Betriebsumgebungen und Entwicklungsumgebungen eingeführt. Das zweite Kapitel enthält die Einführung in Architektur, Programmierung und grundlegende Patterns, die in ASP.NET Anwendung finden. Der Hauptteil erklärt die Konzepte nach und nach am Beispiel einer einfachen Webapplikation. Das dritte und letzte Kapitel enthält eine Gegenüberstellung von Vor- und Nachteilen von ASP.NET und schlägt vor, in welchen Bereichen ASP.NET am Besten einzusetzen ist.

1.1 Das Microsoft .NET Framework

Um ASP.NET verstehen zu können benötigt man auch ein grundlegendes Verständnis für die Konzepte der Microsoft .NET Plattform. In Abbildung 1 sind diese übersichtsartig dargestellt. Die Basis der .NET-Plattform bildet die Common Language Runtime (CLR). Diese stellt eine virtuelle Maschine dar, wie sie auch in der JAVA-Programmiersprache Anwendung findet. Jedweder Programmcode wird in sogenannten Bytecode übersetzt, welcher dann auf dieser virtuellen Maschine ausgeführt wird. In .NET heißt dieser Bytecode Common Intermediate Language (CIL), in früheren Versionen von .NET wurde er Microsoft Intermediate Language (MSIL) genannt. Ähnlich wie in JAVA können

mehrere Programmiersprachen in CIL übersetzen. Eine Besonderheit von .NET ist allerdings, dass man Bibliotheken¹, die in anderen Programmiersprachen geschrieben wurden, in jeder anderen Sprache, die nach CIL übersetzt verwendet werden kann. Verantwortlich hierfür ist das Common Type System (CTS), welches die Datentypen kompatibel hält.

Auf diesen Basistechnologien basieren nun einige Frameworks und Bibliotheken, die die Programmierung für .NET erleichtern. Zuerst sind das natürlich die Basisklassen, die bis auf wenige Ausnahmen auch in ASP.NET verwendet werden können. Den Zugriff auf Daten erleichtert die nächste Schicht mit den Technologien ADO.NET, welches eine universelle Datenabstraktionsschicht anbietet, XML, das hinlänglich bekannt sein sollte, LINQ, eine universelle Datenabfragesprache und das EntityFramework, welches die Speicherung von Datenobjekten in allen möglichen Datenbasen erleichtert. In der Schicht darüber folgt nun ASP.NET, und die Technologien für Desktop-Anwendungen WinForms, Windows Presentation Foundation (WPF), Windows Communication Foundation (WCF) und Windows Workflow Foundation (WF). Über die Common Language Specification erhalten viele unterschiedliche Programmiersprachen Zugriff auf diese Bibliotheken. Insgesamt existieren etwa 50 Programmiersprachen für die .NET Plattform (vgl. [20]). Damit bietet das .NET-Framework sehr umfangreiche Möglichkeiten der Programmierung, auf die auch bei der Erstellung von Webanwendungen zurück gegriffen werden kann. Und sollten diese noch nicht ausreichen, hat man über die COM-Anbindung Zugriff auf viele weitere Bibliotheken aus der Microsoft-Welt (vgl. [1, S.37-44]).

1.2 Die Betriebsumgebung von ASP.NET

ASP.NET ist wie oben schon erwähnt eine von Microsoft entwickelte und vertriebene Technologie. Momentan gibt es keine offiziellen Versionen für Betriebssysteme anderer Hersteller. Des Weiteren ist es momentan nicht möglich, ASP.NET auf einem anderen Webserver neben den Internet Information Services (IIS) (vgl. [4]) zu betreiben. Ein ehemals verfügbares Apache-Modul wird nicht mehr weiterentwickelt (vgl. [18]). Man ist somit beim Einsatz des offiziellen ASP.NET auf eine Betriebsumgebung festgelegt. Mit dem Mono-Projekt (www.mono-project.com) existiert eine quelloffene Alternative, die versucht alle Funktionen von .NET nachzubilden. Für viele Anwendungsfälle ist die Implementierung inzwischen sehr gut geeignet, allerdings ist der Funktionsumfang immer ein wenig geringer als in der offiziellen Version, da die Entwicklung prinzipbedingt nur zeitlich verzögert stattfinden kann.

So wird man in der Praxis also zumeist mit einem Windows-Server und den IIS konfrontiert sein. Die .NET Plattform an sich ist kostenfrei. Allerdings fallen für Betriebssystem und eventuell für Datenbankserver Kosten an.

¹ Der Einfachheit halber werden im Rahmen dieser Arbeit die Begriffe *Bibliothek* und *Assembly* synonym verwendet. Der Autor ist sich der Unterschiede aber durchaus bewusst.

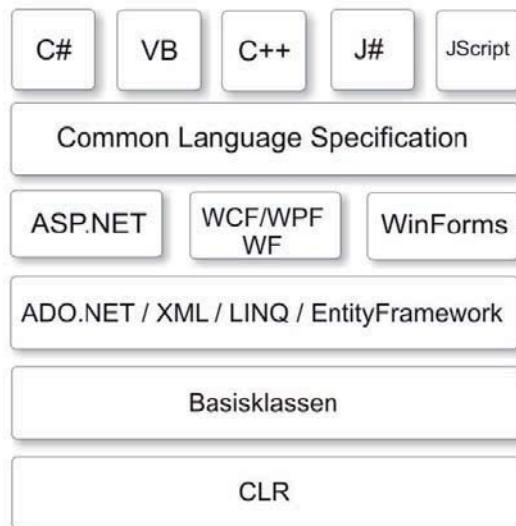


Abbildung 1: Überblick über die .NET-Plattform

1.3 Entwicklungsumgebungen für ASP.NET

Microsoft bietet für die Entwicklung mit ASP.NET das Visual Studio an, welches in unterschiedlichen Varianten genutzt werden kann (Express, Professional, Premium, Ultimate). Diese IDE ist für den professionellen Einsatz gut geeignet und bietet viele Möglichkeiten zur Verwendung in Firmen. Die Expressversion (vgl. [13]) ist für Einsteiger gut geeignet. Es existiert auch eine Installationsdatei, die alle benötigten Programme mitinstalliert (Datenbankserver etc.). Zur Verwendung in Umgebungen, die nicht komplett auf Microsoft-Technologie setzen wollen oder denen der Einsatz von Visual Studio schlichtweg zu teuer ist bietet sich #develop (siehe <http://sharpdevelop.net/opensource/sd/>) an, eine OpenSource-Entwicklungsumgebung für Windows, die auf die Standard .NET-Tools zugreift.

Bewegt man sich in einer mono-Umgebung, gibt es mit monodevelop (siehe <http://monodevelop.com/>) eine stark an #develop angelehnte Lösung, mit der ASP.NET Applikationen entwickelt werden können.

2 ASP.NET im Detail

Der Hauptteil dieser Arbeit enthält vier Kapitel. In diesen vier Kapiteln werden anhand eines durchgehenden Beispiels nach und nach die grundlegenden Architektur- und Programmierkonzepte von ASP.NET erklärt. Bei dem Beispiel handelt es sich um eine Applikation, die auf Wunsch über ein Formular oder über eine URL ein Bild an den Besucher ausliefert, dass genau zu einer ebenfalls angegebenen Größe passt. Die wichtigsten Teile des Beispiels finden sich im Anhang in einer Gesamtübersicht. Kleine Codeausschnitte werden im Fließtext erläutert.

Bei der Programmierung von ASP.NET Anwendungen sind zwei Grundprinzipien zur Architektur zu unterscheiden. Das erste Paradigma, das seit Version 1.0 Anwendung findet sind die sogenannten WebForms. Diese nehmen den Hauptteil der Erläuterungen ein, da sie sich gut eignen, um die Basiskonzepte von ASP.NET zu erläutern. Die zweite Art von Webanwendungen sind die Modell-View-Controller (MVC) Seiten. Diese nahmen in der vierten Ausgabe von ASP.NET Einzug und liegen mittlerweile in Variante 3 vor. Zur Entwicklung von neuen Webanwendungen wird es bevorzugt eingesetzt, da es sich auch in anderen Frameworks (Ruby on Rails, Django, CakePHP usw.) als Basis für Webapplikationen bewährt hat.

Zu Beginn soll kurz auf die Verarbeitung einer Seitenanfrage eingegangen werden, um besser nachvollziehen zu können, an welchen Stellen Eingriff in diese möglich ist und wann welche Objekte zur Programmierung zur Verfügung stehen.

2.1 Der Lebenszyklus einer Serveranfrage

In 1.2 wurde erläutert, dass eine ASP.NET Applikation im Normalfall zusammen mit den IIS betrieben wird. Es besteht eine enge Verbindung zwischen diesen Applikationen, und so kann man mit ASP.NET auch die Verarbeitung der IIS manipulieren.

Trifft eine neue Anforderung beim Server ein, wird die Anfrage von den IIS entgegen genommen. Sie werten die Art der Anfrage über die Dateiendung aus und stellen fest, an welche Stelle die Anfrage weiter geleitet werden soll. Im Fall einer ASP.NET Seite wird der Aufruf normalerweise über die Endung *.aspx* initiiert. Die Services leiten dann die Anfrage an die *aspnet_isapi.dll* weiter. Wie der Name der Datei andeutet, handelt es sich dabei um eine sogenannte ISAPI-Erweiterung (Internet Services API). Die ISAPI-Schnittstelle ist eine maschinennahe Schnittstelle für den Zugriff auf HTTP-Anfragen. Diese wird von ASP.NET genutzt, um auf die Anforderung von Seiten zuzugreifen und Anfragen zu beantworten. Beim Eintreffen einer Anfrage werden folgende Klassen erzeugt oder bei Existenz die entsprechende Referenz erzeugt:

HttpContext enthält die kompletten Umgebungsdaten einer Serveranfrage. Insbesondere die Klassen Request, Response, Application, Server und Cache, sind Eigenschaften dieser Oberklasse. Auf diese Klassen wird zu einem späteren Zeitpunkt noch einmal näher eingegangen.

HttpApplication bildet die Hülle einer ASP.NET Applikation. Hier können Ereignisse verarbeitet werden, die den Gesamtlebenszyklus einer Seite betreffen. Die Datei *global.asax*, die Teil jeder ASP.NET Applikation ist, enthält eine Implementierung dieser Klasse und dort können auch die Ereignisse mit eigenen Methoden überschrieben werden. Welche Ereignisse bei der Behandlung einer Anfrage auftreten, ist in Listing 1.1 dargestellt. Diese Klasse wird global beim ersten Aufruf der Applikation erzeugt und dann an den jeweiligen HttpContext übergeben.

Um diesen Vorgang zu verdeutlichen wird im Anwendungsbeispiel auf Application Ebene die URL der Anfrage manipuliert und umgeleitet:

Listing 1.1: Überschreiben einer URL in der Datei global.asax.cs

```

1 //Abfrage des ersten Segments der URL-Daten:
2 string width = Request.Url.Segments[1];
3
4 string newUrl = string.Format("/{0}/Default.aspx?width={0}&height={1}", width,
   height);
5 Context.RewritePath(newUrl);

```

Man erkennt den Zugriff auf die Basisklassen Request und Context. Besonders ist, dass diese Verarbeitung nicht auf eine Seite beschränkt ist, sondern auf Applicationsebene stattfindet. Im Beispiel wird hiermit eine URL der Form

image/{width}/{height}

zur Seite *Default.aspx* mit den Parametern *width* und *height* umgeleitet.

Bisher wurde die Dimension der Applikation betrachtet. Interessanter für die Programmierung von ASP.NET Seiten ist allerdings das Umfeld einer Seite. Auch diese durchläuft bei ihrem Aufruf eine Abfolge von Ereignissen:

Start Um Kontrolle über den ganzen Lebenszyklus einer Seite zu haben benötigt man auch Zugriff auf deren Aufruf (Request) und deren Ergebnis (Response). Diese Eigenschaften werden im Ereignis Start initialisiert.

Initialisierung Erstellen der Steuerelemente

Laden Setzen der Eigenschaften der Steuerelemente, Auswertung der Daten.

Validierung Steuerelemente werden auf Basis von Regeln auf ihre Richtigkeit geprüft.

Ereignisbehandlung Auswertung der Steuerelementereignisse, bei Postback (näheres hierzu in 2.3)

Rendern Erstellen des HTML-Quelltextes der Seite

Entladen Seite wird entladen, die Daten sind in den Ausgangstrom geschrieben, das Seitenobjekt wird aufgelöst.

Wie auch in die Ereignisse der Gesamtapplikation kann auch in jedes Ereignis der Seitenverarbeitung eingegriffen werden. In Listing 1.2 ist dargestellt, wie man einen String in den Ausgangsstrom schreibt.

Listing 1.2: Überschreiben des Ereignisses Page_Load

```
1 protected void Page_Load(object sender, EventArgs e)
2 {
3     Response.Write("Dieser String wird in den Ausgangsstrom geschrieben.");
4 }
```

Wo dieser Quelltext in der Struktur einer ASP.NET Anwendung stehen muss und wie allgemein mit Quelltext umgegangen wird, ist im nächsten Kapitel beschrieben.

2.2 Quelltext einer ASP.NET Seite

Im Rahmen dieser Arbeit wird für die Darstellung von Quellcode und in Beispielen die Programmiersprache C# verwendet. Das hat den Hintergrund, dass diese Sprache eine für Programmierer recht einfache Syntax aufweist, welche sehr nahe an der von Java ist. Man sollte die Beispiele also auch verstehen können, wenn man die Programmiersprache nicht beherrscht.

ASPX-Seiten bilden die Grundeinheit von WebForms und können auch im Rahmen von MVC eingesetzt werden. An späterer Stelle aber noch mehr hierzu. Der Header einer solchen Datei ist in Listing 1.3 dargestellt.

Listing 1.3: Beispielheader einer ASP.NET Seite

```
1 <% Page Language="C" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
   Inherits="WebApplication1.WebForm1" %>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
   org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3 <html xmlns="http://www.w3.org/1999/xhtml">
4 <head runat="server">
5   <title>Seitentitel</title>
6 </head>
```

Mit *Language* wird die verwendete Programmiersprache innerhalb der Seite angegeben. In diesem Fall ist das C# - jede .NET Programmiersprache hat hier ihr eigenes Kürzel. Wird hier nichts angegeben, wird automatisch VisualBasic.NET verwendet. Mit *CodeBehind* wird festgelegt, ob der Datei eine Code-Datei hinterlegt wurde, oder ob der Code innerhalb dieses Dokuments definiert ist. Dies ist der Fall, wenn das Attribut nicht gesetzt wird. *AutoEventWireUp="true"* sorgt dafür, dass Ereignisse automatisch an Funktionen mit dem Schema:

OBJEKT VARIABLE _EREIGNISNAME (PASSENDE PARAMETERLISTE)

gebunden werden (vgl. [19]). Außerdem erbt eine ASP.NET Webseite immer von der Klasse Page. In diesem Fall erbt schon die Klasse in der CodeBehind-Datei (*WebApplication1.WebForm1*) von Page, weswegen in der .aspx-Datei nur noch

auf diese verwiesen werden muss (mit dem Attribut *Inherits*).

Anschließend folgt ein Standard-Header für XHTML-Dateien. Dieser hat primär nichts mit ASP.NET zu tun. Welchen Mime-Type die Seite besitzt, kann über ein zusätzliches Attribut im ASP.NET Header, *ContentType* festgelegt werden. Vorsicht ist beim Einsatz einiger Steuerelemente geboten, deren Ausgabe nicht XHTML konform ist (vgl. [17]). Weitere Content-Typen können natürlich auch verwendet werden (HTML etc.). Die Manipulation des MIME-Typs ist ebenfalls über das Response-Objekt möglich. Das zeigt das Beispiel, dass als Ergebnis keine weitere HTML-Seite sondern ein Bild auslieferiert. Die betreffende Code-Zeile lautet:

Listing 1.4: Setzen eines MIME-Typs über das Response-Objekt

```
1 //Neuen ContentType setzen -> jpeg
2 Response.ContentType = "image/jpeg";
```

In der Seitenstruktur folgen nun die (X)HTML-Tags des Seitenanfangs. Besonders hierbei ist, dass alle Tags, deren Elemente vom Server bearbeitet werden sollen mit dem Attribut *runat="server"* gekennzeichnet werden müssen. In diesem Fall wird der *head* der Seite auf dem Server um zusätzliche Attribute ergänzt.

Möglichkeiten zur Einbindung von Code in .aspx-Seiten Die erste Möglichkeit zur Verwendung von Codeschnipseln innerhalb einer Seite ist das mit dem *runat="server"*-Attribut versehene script-Tag:

Listing 1.5: Einbindung von Code über das *<script>*-Tag

```
1 <script language="C#" runat="server">
2   //code
3 </script>
```

Code der hier definiert ist, wird ausgeführt und eventuelle Ausgaben an der entsprechenden Stelle in den *Response*-Stream geschrieben. Die verwendete Programmiersprache kann hier wieder mit dem *language*-Attribut eingestellt werden.

Die nächste Möglichkeit bieten die %-Tags:

Listing 1.6: Einbindung von Code über das <%-Tag

```
1 <%
2   //code
3 %>
```

Die bei zu exzessivem Einsatz allerdings die Lesbarkeit des Codes stark erschweren. Es existieren vier Spezialarten dieses Tags:

<%@ ... %> Server-Direktiven (vgl. [12])

<%= ... %> Schreibt den Rückgabewert einer Funktion in den Ausgabestrom der Seite (ersetzt ein *Response.Write();*) an dieser Stelle des Codes (vgl. [1, S.389f])

<%: ... %>	Wie %, nur dass die Ausgabe mit HTML Enkodierung versehen wird (vgl. [1, S.389f])
<%# ... %>	Ausdruck zur Bindung dynamischer Daten (vgl. [1, S.390])
<%\$... %>	Ausdruck zur Bindung statischer Daten (vgl. [1, S.390f])

Der Einsatz dieser Konstrukte empfiehlt sich bei sehr kurzen Informationsschnipseln, die in eine Seite eingefügt werden sollen. Sie tragen zu einer schöneren Trennung von Code und Design bei.

Bei Verwendung von viel Programmcode empfiehlt sich insgesamt also die Verwendung der *CodeBehind*-Eigenschaft des Dokuments. Dann kann eine separate Datei zur Programmierung verwendet werden. Dort können auch Funktionen zur Ereignisbehandlung platziert sein.

2.3 Das Konzept der WebForms

Seit ASP.NET Version 1.0 stellten WebForms das Grundparadigma dar, mit dem Seiten erstellt wurden und wird auch immer noch häufig eingesetzt. Das Prinzip kommt schon im Namen zum Ausdruck: jede Seite enthält ein HTML-Formular. Das Beispiel enthält eine einfache Maske mit 2 Eingabefeldern und einem Button. Der Quelltext ist vereinfacht in Listing 1.7 zu sehen, die Ausgabe in Abbildung 2. An dieser Stelle sei darauf hingewiesen, dass zum Layout von WebForms in Visual Studio auch ein grafischer WYSIWYG-Editor zur Verfügung steht, mit dem einfach Formulare, aber auch ganze Webseiten entworfen werden können.

Listing 1.7: Beispiel einer WebForm

```

1 <body>
2   <form id="Form1" runat="server">
3     <asp:Label ID="l1" Text="Höhe" runat="server"></asp:Label>
4     <asp:TextBox ID="height" runat="server"></asp:TextBox>
5
6     <asp:Label ID="l2" runat="server" Text="Breite"></asp:Label>
7     <asp:TextBox ID="width" runat="server"></asp:TextBox>
8
9     <asp:Button ID="BildAnfordern" runat="server" onclick="BildAnfordern_Click"
10    Text="Bild anfordern!" />
11   </form>
12 </body>

```

Herzlich Willkommen!

Höhe

Breite

Abbildung 2: Ausgabe von Listing 1.7

Man erkennt auch hier wieder das Attribut `runat="server"`, welches darauf hinweist, dass die Verarbeitung der Form vom Server beeinflusst wird. Das Standardverhalten ruft die Seite im Falle eines *Submits* über einen POST-Request erneut auf. Das ist der sogenannte *Postback*.

Dieser kann mit Hilfe der Eigenschaft `IsPostBack` des Page-Objektes abgefragt werden. Bei einem "normalen" Aufruf ist der Wert `false` und beim Aufruf über den POST-Request des Clients ist der Wert `true`. Das erleichtert den Umgang mit Formularen auf dem Server erheblich, da eine einfache Fallunterscheidung möglich ist. Darüber hinaus können innerhalb des Postbacks Ereignisse von Steuerelementen abgehandelt werden. Eine grafische Veranschaulichung des Postback findet sich in Abbildung 3.



Abbildung 3: Postback

Da der Postback ein wichtiges Element innerhalb der WebForms darstellt, findet sich in Listing 1.8 ein kurzes Beispiel, wie man den Postback abfängt.

Listing 1.8: Abfrage des Postbacks im Ereignis Page_Load

```

1 protected void Page_Load(object sender, EventArgs e)
2 {
3     If(IsPostBack) {
4         Response.Write("Die WebForm wurde abgeschickt!");
5     }
6 }
```

Das Formular im Beispiel verwendet Steuerelementereignisse, um den Klick auf den Button abzufangen dieses Listing findet sich im nächsten Kapitel.

Steuerelemente Steuerelemente bilden einen wichtigen Teil von ASP.NET. Sie liegen in einige unterschiedlichen Ausprägungen vor:

Server-Steuerelemente Server-Steuerelemente sind keine HTML-Formular-Elemente, sondern eher Steuerelemente, wie sie auch in anderen Windows-Technologien zum Einsatz kommen. Zum Beispiel kann man eine Baumansicht wie aus der Seitenleiste des Windows-Explorer bekannt auch innerhalb einer ASP.NET Seite verwenden. Exemplarisch sind in Listing 1.7 zwei Textboxen, zwei Labels und ein Button zu sehen. Serversteuerelemente werden von ASP.NET in HTML-Code umgewandelt (vgl. [10]).

HTML-Steuerelemente Man kann in ASP.NET auch die HTML-Formular-Steuerelemente verwenden, und sie dann per *runat="server"* als Eigenschaft des Page-Objektes verankern. Somit kann man auf sie genauso wie auf Server-Steuerelemente zugreifen (vgl. [1, S.416ff.]).

Benutzersteuerelemente Natürlich ist man nicht an die beiden oben genannten Arten von Steuerelementen gebunden, und kann selbst eigene definieren. Hierfür stehen zwei Möglichkeiten zur Verfügung, einmal die User Controls, welche die Kombination von mehreren Server-Steuerelementen erlauben und mit eigenen Methoden und Eigenschaften versehen werden können, und andererseits die Custom Controls, welche eine komplett eigene Implementierung der Control- oder der WebControl-Klassen darstellen. Erstere sind deutlich einfacher zu erstellen (vgl. [8]).

Webparts Webparts erlauben die nutzerseitige Manipulation von Seiten durch den Benutzer und stellen dafür Speichermöglichkeiten und Steuerelemente bereit (vgl. [9]).

Zum Abschluss des Kapitels über Webforms soll noch einmal das Beispiel zusammengefasst werden. Die Datei *Default.aspx* erhält entweder über ein Formular oder über ihre URL zwei Parameter übermittelt (*width* und *height*) und gibt daraufhin ein Bild aus, dass sie über eine Bibliothek abfragt. An Hand dieses Beispiels wurde vorgestellt, wie man in den Verarbeitungszyklus der Gesamtapplikation und der Seite eingreift. Außerdem, wie man Code in eine *.aspx*-Datei einbindet und wie man sie über CodeBehind an eine *.cs*-Datei bindet. Außerdem wurden Steuerelemente, Steuerelementereignisse und der Postback demonstriert. Damit ist die Darstellung der WebForms abgeschlossen und nun soll auf die Implementierung des MVC-Patterns in ASP.NET eingegangen werden.

2.4 Die ASP.NET MVC-Implementierung

Das Model-View-Controller-Pattern ist Bestandteil der meisten modernen Webframeworks (vgl. Ruby on Rails, CakePHP, Spring etc.). Es propagiert die Trennung von Datenhaltung, Anwendungslogik und Präsentation. Mit dem .NET-Framework 3.5 wurde in ASP.NET auch eine Implementierung dieses Konzeptes eingebaut. Auf die Grundlagen des Patterns soll hier nicht eingegangen werden, weswegen an dieser Stelle nur die Grundlagen und Besonderheiten von MVC in ASP.NET erläutert werden sollen.

Convention over Configuration Das Prinzip Convention over Configuration ist ebenfalls aus anderen Webframeworks bekannt und bedeutet, dass ein System mit den Default-Einstellungen lauffähig ist und beispielsweise Klassen-, Datei- und Pfadnamen standardmäßig sinnvoll gesetzt sind (Eine einführende Beschreibung findet sich zum Beispiel in [21]).

Auch in ASP.NET findet dieses Prinzip Anwendung. Der Programmcode wird automatisch in die drei Ordner Models, Views und Controllers aufgeteilt. Dateien, die einen Controller beinhalten, werden immer mit *Controller* am Ende des Dateinamens benannt erwartet. Eine ähnliche Konvention gibt es für den zum Controller gehörigen View. Dieser muss sich in einem Verzeichnis unterhalb von *Views* befinden, das genauso heißt wie der vordere Teil des Dateinamens des Controllers. Wenn der Controller also *HomeController.aspx* heißt, dann muss das View-Verzeichnis *Home* sein. Models dürfen hingegen frei benannt werden. Diese Benennungsschemata sind in Abbildung 4 dargestellt.

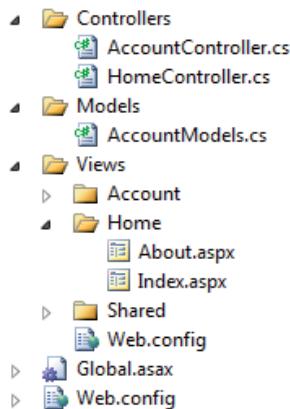


Abbildung 4: Ordnerstruktur einer ASP.NET MVC-Anwendung

Hand in Hand mit dieser Ordnerstruktur geht auch das URL-Routing. Darunter versteht man die Formatierung der URL einer Seite in einem ASP.NET Projekt. Per Default lautet die Aufteilung:

```
controller/action/id
```

Der Aufruf von `home/index/3` würde im Controller `home` die Funktion `index` mit dem Parameter 3 aufrufen. Durch das Überschreiben der Methode `RegisterRoutes` in der Datei `global.aspx` kann diese Standardeinstellung aber auch geändert werden (vgl. [15]).

Das Model Die Model Implementierung von ASP.NET unterscheidet sich von denen in anderen Frameworks. Als Basis für die Abbildung des Models bietet zum Beispiel Ruby on Rails die Möglichkeit an, über einen sogenannten objektrelationalen Mapper die Daten in einer Datenbank zu persistieren. ASP.NET bietet hier eine deutlich höhere Flexibilität. Man hat die Möglichkeit das ADO.NET Entity-Framework zu verwenden, bei dem es sich um einen objektrelationalen Mapper handelt, der mit unterschiedlichen Datenbanken zusammenarbeiten kann (mit allen, die das ADO.NET Framework unterstützen). Daraüber hinaus kann man selbst für die Persistenz der Klassen sorgen und dabei auf andere Technologien zugreifen (LINQ-to-SQL, Objektorientierte Datenbanken etc.). Das EntityFramework bietet dabei noch drei unterschiedliche Möglichkeiten an, wie ein Model erstellt werden kann:

Model first Mit einem in Visual Studio enthaltenen Designer lassen sich Klassendiagramme, wie aus UML bekannt, direkt als Model einer MVC-Anwendung modellieren.

Code first Will man zuerst den Code des Models implementieren und sich dann Gedanken über dessen Persisterung machen, ist auch das möglich. Bestehende Klassen können im Entity Framework abgespeichert werden.

Database first Will man eine MVC-Anwendung auf eine bestehende Datenbank aufbauen, oder mit einer optimierten Datenbank beginnen, ist auch der Weg von der Datenbank zum Model hin möglich (vgl. [11]).

Die Verwendung der Controller und Views unterscheidet sich nicht besonders stark von den Implementierungen anderer Web-Frameworks, weswegen an dieser Stelle auf weitere Erläuterungen verzichtet werden soll. In Abbildung 5 ist überblicksartig ein kleines MVC-Beispiel dargestellt, es enthält keine Model-Bindung. Ein Objekt des Models `Lehrveranstaltung` wird im Controller `LehrveranstaltungController` erzeugt und über `View(model)` an den View übergeben. In diesem kann dann über `Model.{Eigenschaftsname}` zugegriffen werden.

Razor Am Ende der Beschreibung des MVC-Patterns in ASP.NET soll die neue View-Technologie Razor erklärt werden, die in MVC Version 3 eingeführt wurde. Es handelt sich dabei um eine neue Syntax für die Erstellung von Views. Man versucht von den oben genannten Arten weg zu kommen, mit der Code in .aspx-Seiten eingebettet werden kann und führt dafür das universale „@“ ein, welches an dessen Stelle tritt. Zum besseren Verständnis an dieser Stelle ein kurzes Code-Beispiel:

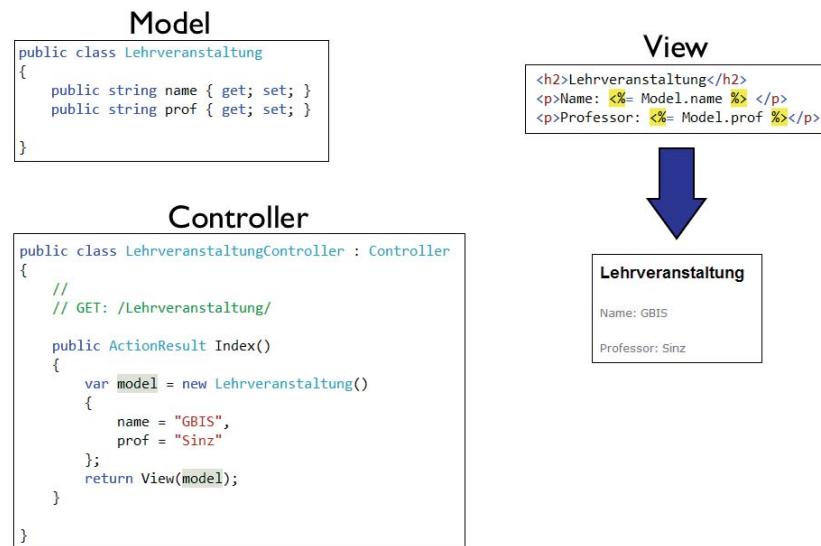


Abbildung 5: Beispiel einer mit MVC erstellten Seite

Listing 1.9: foreach-Schleife in einer .aspx-Datei

```

1 <% foreach(Product p in products){ %>
2   <li><%= p.Name %> ($<%=p.Price %>)</li>
3 <% } &>
```

Listing 1.10: foreach-Schleife in einer Razor-Datei

```

1 @foreach(Product p in products){
2   <li>@p.Name ($@p.Price)</li>
3 }
```

Mehrzeilige Code-Blöcke können einfach mit @{...} eingefügt werden. Man erkennt die sehr knappe Syntax von Razor gut und damit auch die großen Vorteile. Es ist sehr einfach Templates zu erstellen, die durch Webdesigner erstellt werden können und damit die Trennung zwischen Anwendungslogik und Design weiter zu vereinfachen. Sogenannte Master-Seiten existieren für normale ASP.NET-Seiten genauso wie für Razor, mit letzterem ist die Übersichtlichkeit aber wie gesehen, deutlich besser. Neben der besseren Lesbarkeit des Codes wurde die Engine zur einfacheren Erlernbarkeit und zur Testbarkeit entwickelt (zu Razor siehe [3]). In der Testbarkeit liegt auch ein Vorteil von MVC gegenüber WebForms im allgemeinen. WebForms lassen sich nur von Hand und nicht automatisiert testen.

MVC-Anwendungen hingegen erlauben den Einsatz von Unit-Tests für die Controller und mit Razor eben nun auch für die Views, was die Qualitätskontroll bei Webanwendungen stark vereinfacht.

Ob sich Razor gegen den normalen .aspx-Ansatz durchzusetzen vermag ist noch nicht abzusehen, allerdings scheint der Ansatz sinnvoll und einfach genug, als dass Entwickler und Designer das neue Konzept in Erwägung ziehen werden.

2.5 Zusammenfassung weiterer ASP.NET-Technologien

Zwischen der Vorstellung der Haupttechnologien kamen immer wieder andere Frameworks, Bibliotheken und Schnittstellen zur Sprache, die zum Abschluss des Hauptteils nochmals kurz zusammengefasst werden sollen.

ADO.NET Ist seit Frameworkversion 1.1 Teil von .NET und bildet eine Datenabstraktionsschicht, die sowohl mit XML-Daten als auch mit relationalen Datenbanken (mit OLE DB- oder ODBC-Treiber) umgehen kann. Es bietet die Möglichkeit von der konkreten Datenbank unabhängige Datenhaltungsschichten zu implementieren (weitere Informationen finden sich in [6]).

Masterseiten Die Masterseiten sind die Implementierung einer Template-Engine für ASP.NET. Eine .aspx- oder Razor-Seite kann in ihrem Header eine Masterseite definieren, in die sie eingesetzt werden soll und in der zugehörigen Masterseite gibt es ein Serversteuerelement <*asp:ContentPlaceHolder*>, an dessen Stelle der Inhalt dann eingefügt wird (vgl. [19, S.714ff]).

AJAX in ASP.NET Eine Technologie, die bislang noch nicht erwähnt wurde, die aber im Zusammenhang mit Web 2.0 nicht unerwähnt bleiben darf ist AJAX. Wie jedes Web-Framework bietet auch ASP.NET die Möglichkeit zur Verwendung von asynchronen Serveranfragen. Weitere Informationen finden sich beispielsweise in [2] oder [7].

3 Fazit

Abschließend wird in dieser Arbeit auf die Vor- und Nachteile des ASP.NET Frameworks eingegangen. Insbesondere soll dabei erläutert werden, für welche Einsatzzwecke die Technologie geeignet ist und in welchen Bereichen man lieber darauf verzichten sollte. Zunächst soll auf die Vorteile eingegangen werden.

3.1 Vorteile von ASP.NET

Der größte Vorteil von ASP.NET ist gleich im Namen erkennbar: es ist Teil des großen .NET Frameworks und kann auf alle darin enthaltenen Funktionen zugreifen. Somit gibt es für fast alles schon vorgefertigte Lösungen und Bibliotheken, die man verwenden kann. Gute Möglichkeiten bietet das vor allem bei der Integration in Firmenumgebungen, in denen Web-Services auf .NET-Basis eingesetzt werden und allgemein in Umgebungen, in denen auf .NET Technologie gesetzt wird. Eine Firma, die zum Beispiel Workflow-Lösungen auf dieser Basis verwendet, kann viele Synergieeffekte erzielen, wenn es auch die zugehörige Webapplikation zur Auswertung der Analysedaten in ASP.NET realisiert.

Darüber hinaus kann man auch auf COM-Schnittstellen zugreifen, was zum Beispiel die Manipulation von Microsoft Office Dateien in Web-Anwendungen ermöglicht - oder eben deren Erstellung auf Grund eingestellter Parameter auf einer Homepage.

Des Weiteren sind die Möglichkeiten zur Datenhaltung durch ADO.NET in ASP.NET extrem flexibel. Die Abstraktionsschicht ermöglicht saubere Trennung innerhalb der Applikationen und damit auch die Trennung von Datenbankdesignern und Applikationsentwicklern, wie es vor allem bei größeren Projekten sinnvoll ist.

Zuletzt ist es sinnvoll anzumerken, dass ASP.NET sich an vielen Stellen so programmiert wie eine Windows-Desktopanwendung. Hat man schon Entwickler aus diesem Hintergrund zur Verfügung, fällt es ihnen sicherlich leichter auf ASP.NET umzusteigen, als auf eine völlig neue Technologie.

3.2 Nachteile von ASP.NET

Der Vorteil, den die .NET Bibliothek mit sich bringt, erkauft man sich beim Einsatz durch die Plattformabhängigkeit beim Betriebssystem. Will man immer den vollen Funktionsumfang nutzen, muss man die Implementierung von Microsoft einsetzen, welche nur auf Windows-Betriebssystemen lauffähig ist. Mit Mono existiert zwar eine plattformunabhängige Alternative, allerdings kann deren Entwicklung nur zeitverzögert stattfinden.

Betrachtet man die Bücher oder die MSDN-Artikel zu ASP.NET tritt ein weiterer Nachteil zu Tage. Wo andere Frameworks schlank und einfach beschrieben werden können, weil sie es auch sind, umfassen die meisten Bücher über ASP.NET mehr als 1000 Druckseiten. Die Einarbeitung in ASP.NET gelingt nicht besonders schnell, sondern erfordert einigen Zeit- und Leseaufwand.

Wie auch im Hauptteil schon an mancher Stelle anklingt, ist das Konzept der

WebForms zwar immer noch ein wichtiger Teil von ASP.NET, allerdings ist die Architektur an vielen Stellen nicht mehr zeitgemäß und der Trend geht klar zum neueren Ansatz MVC. Das ist sicherlich kein Nachteil der Plattform an sich, aber viele Anwendungen werden sich nur schwer migrieren lassen, was eine Legacy Unterstützung notwendig macht, und somit das Framework aufbläst.

Ob man den letzten Punkt als Nachteil oder als Vorteil sieht, liegt im Auge des Betrachters. Die Release-Zyklen von ASP.NET sind deutlich langsamer als die der anderen Frameworks, was verhindert, dass Innovationen zeitnah Einzug finden, was aber auch eine Sicherheit bei der Verwendung von Bibliotheken mit sich bringt.

Wägt man diese Vor-und Nachteile gegeneinander ab, ergibt sich ein Vorteil von ASP.NET überall dort, wo Windows als Serverplattform zum Einsatz kommt und wo komplexe Projekte implementiert werden sollen. An Stellen, wo schnell und ohne viel Einarbeitungszeit eine Webapplikation erstellt werden soll, kann man zu kleineren und agileren Plattformen raten.

4 Weiterführende Literatur

Sehr empfehlenswert zur Einarbeitung in ASP.NET ist das Buch von Matthias Fischer und Jörg Krause (vgl. [1]), das sich auch an komplexe Zusammenhänge wagt und dabei immer klar strukturiert und verständlich bleibt.

An Quellen im Netz muss die sehr hilfreiche MSDN von Microsoft (vgl. [14]) gelobt werden, die zu vielen Themen einen guten Einstieg bietet. Darüber hinaus ist auch die offizielle Seite zu ASP.NET zu erwähnen, die vor allem mit guten Videotutorials für einige Themen aufwarten kann (vgl. [16]). Zum Abschluss eine Übersicht über hilfreiche Webseiten:

- <http://www.asp.net/> - Offizielle asp.net-Homepage
- <http://msdn.microsoft.com> - Nachschlagewerk für Microsoft Programmierer
- <http://weblogs.asp.net/scottgu/> - Blog des Chefentwicklers von ASP.NET
- <http://www.4guysfromrolla.com> - Unabhängige Seite mit unterschiedlichen Ressourcen zu ASP.NET
- <http://www.stackoverflow.com> - Frage und Antwortseite für unterschiedliche Programmiersprachen
- <http://www.asp.net/downloads> - Download von ASP.NET
- <http://www.icsharpcode.com/OpenSource/SD/Download/> - Download von #develop
- <http://www.mono-project.org> - Homepage des Mono-Projekts
- <http://www.monodevelop.org> - Homepage von monodevelop

Literatur

1. Matthias Fischer and Jörg Krause. *ASP.NET 4.0: Konzepte und Techniken zur Programmierung für Websites*. Hanser, München, 2010.
2. Jana Frank and Patrick A. Lorenz. *ASP.NET 3.5 mit AJAX*. Number Ill. Hanser, München, 1. aufl. edition, 2008.
3. Scott Guthrie. Introducing "razor" – a new view engine for asp.net. <http://weblogs.asp.net/scottgu/archive/2010/07/02/introducing-razor.aspx> (31.07.2011).
4. Microsoft. Accessing data with ado.net. [http://msdn.microsoft.com/en-us/library/e80y5yhx\(v=VS.71\).aspx](http://msdn.microsoft.com/en-us/library/e80y5yhx(v=VS.71).aspx) (31.07.2011).
5. Microsoft. Ajax. <http://www.asp.net/ajax> (31.07.2011).
6. Microsoft. Asp.net user controls overview. <http://msdn.microsoft.com/de-de/library/fb3w5b53.aspx> (31.07.2011).
7. Microsoft. Asp.net web parts overview. <http://msdn.microsoft.com/de-de/library/hhy9ewf1.aspx> (31.07.2011).
8. Microsoft. Asp.net-webserver-steuerelemente. <http://msdn.microsoft.com/de-de/library/bb386416.aspx> (31.07.2011).
9. Microsoft. Creating an entity framework data model for an asp.net mvc application. <http://www.asp.net/entity-framework/tutorials/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application> (31.07.2011).
10. Microsoft. Directive syntax. [http://msdn.microsoft.com/en-us/library/xz702w3e\(v=vs.71\).aspx](http://msdn.microsoft.com/en-us/library/xz702w3e(v=vs.71).aspx) (31.07.2011).
11. Microsoft. Download and install asp.net for free. <http://www.asp.net/downloads> (31.07.2011).
12. Microsoft. Internet information services. <http://www.iis.net/> (31.07.2011).
13. Microsoft. Msdn. <http://msdn.microsoft.com/> (31.07.2011).
14. Microsoft. Mvc framework and application structure. [http://msdn.microsoft.com/en-us/library/dd410120\(VS.98\).aspx](http://msdn.microsoft.com/en-us/library/dd410120(VS.98).aspx) (31.07.2011).
15. Microsoft. Scott guthrie. <http://msdn.microsoft.com/en-us/ee832729> (31.07.2011).
16. Microsoft. www.asp.net. <http://www.asp.net/> (31.07.2011).
17. Microsoft. Xhtml standards in visual studio and asp.net. <http://msdn.microsoft.com/en-us/library/exc57y7e.aspx> (31.07.2011).
18. William A. Rowe. mod_aspdotnet. <http://sourceforge.net/projects/mod-aspdotnet/files/> (31.07.2011).
19. Holger Schwichtenberg. *Microsoft ASP.NET 3.5 mit Visual C# 2008 - Das Entwicklerbuch: Grundlagen, Techniken, Profi-Know-how*. Microsoft, Unterschleißheim, 1 edition, 2008.
20. Wikipedia. List of cli languages. http://en.wikipedia.org/wiki/List_of_.NET-languages (31.07.2011).
21. www.springsource.org. Convention over configuration. <http://static.springsource.org/spring/docs/2.0.x/reference/mvc.html#mvc-coc> (31.07.2011).

Anhang

Listing 1.11: Default.aspx

```

1 <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
   Inherits="MISemBeispiel.GetImage" %>
2
3 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
   org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4
5 <html xmlns="http://www.w3.org/1999/xhtml">
6
7 <head runat="server">
8   <title></title>
9 </head>
10 <body>
11
12 <h1>Herzlich Willkommen!</h1>
13
14 <form id="form1" runat="server">
15   <asp:Table ID="Table1" runat="server" Height="114px" Width="200px">
16     <asp:TableRow runat="server">
17       <asp:TableCell runat="server"><asp:Label ID="11" Text="Höhe" runat=
         ="server"></asp:Label></asp:TableCell>
18       <asp:TableCell runat="server"> <asp:TextBox ID="height" runat="
         server"></asp:TextBox></asp:TableCell>
19     </asp:TableRow>
20
21     <asp:TableRow runat="server">
22       <asp:TableCell runat="server"><asp:Label ID="12" runat="server"
         Text="Breite"></asp:Label></asp:TableCell>
23       <asp:TableCell runat="server"><asp:TextBox ID="width" runat="
         server"></asp:TextBox></asp:TableCell>
24     </asp:TableRow>
25   </asp:Table>
26
27   <asp:Button ID="BildAnfordern" runat="server" onclick="BildAnfordern_Click
      " Text="Bild anfordern!" />
28 </form>
29 </body>
30 </html>

```

Listing 1.12: Default.aspx.cs

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.UI;
6 using System.Web.UI.WebControls;
7 using ImageWork;
8 using System.IO;
9 using System.Drawing.Imaging;
10
11 namespace MISemBeispiel
12 {
13   public partial class GetImage : System.Web.UI.Page
14   {
15     protected void Page_Load(object sender, EventArgs e)
16     {
17       try
18       {
19         //Umgang mit dem _GET Request:

```

```

20             if (Convert.ToInt32(Request.QueryString["width"]) > 0 &&
21                 Convert.ToInt32(Request.QueryString["height"]) > 0)
22             {
23                 //Aufruf der Methode zum Senden des Bildes an den Client
24                 sendPicture(Convert.ToInt32(Request.QueryString["width"]),
25                             Convert.ToInt32(Request.QueryString["height"]));
26             }
27         }
28         catch (Exception ex)
29         {
30             //Bei Fehler wird auf die aktuelle Datei zurückgeleitet
31             Server.Transfer("~/Default.aspx");
32         }
33     }
34     protected void BildAnfordern_Click(object sender, EventArgs e)
35     {
36         //Aufruf der Methode zum Senden des Bildes an den Client
37         sendPicture(Convert.ToInt32(width.Text), Convert.ToInt32(height.
38             Text));
39     }
40     private void sendPicture(int width, int height){
41         //Aktuelle Header der Response löschen
42         Response.ClearHeaders();
43         //Neuen ContentType setzen -> jpeg
44         Response.ContentType = "image/jpeg";
45         //Holen eines Bildes
46         ImageR img = new ImageR(width, height, ".\\images");
47         //Resize des geholten Bildes
48         System.Drawing.Bitmap newImg = (System.Drawing.Bitmap)img.
49             resize();
50         //Schreiben des Bildes in den Ausgabestrom von Response
51         newImg.Save(Response.OutputStream, System.Drawing.Imaging.
52             ImageFormat.Jpeg);
53     }

```

Listing 1.13: Global.asax

```

1 <%@ Application Codebehind="Global.asax.cs" Inherits="MISemBeispiel.Global"
   Language="C#" %>

```

Listing 1.14: Global.asax.cs

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.Security;
6 using System.Web.SessionState;
7
8 namespace MISemBeispiel
9 {
10     public class Global : System.Web.HttpApplication
11     {
12
13         protected void Application_Start(object sender, EventArgs e)
14         {
15         }
16

```

```
17     protected void Session_Start(object sender, EventArgs e)
18     {
19
20     }
21
22     protected void Application_BeginRequest(object sender, EventArgs e)
23     {
24         if (Request.Url.Segments.Length == 4)
25         {
26             //Aufteilung der URL
27             string page = Request.Url.Segments[1].Substring(0, Request.Url
28                 .Segments[1].Length - 1);
29             string width = Request.Url.Segments[2].Substring(0, Request.
30                 Url.Segments[2].Length - 1);
31             string height = Request.Url.Segments[3];
32
33             //Prüfen der Parameter
34             if (page == "image" && Convert.ToInt32(width) > 0 && Convert.
35                 ToInt32(height) > 0)
36             {
37                 //Weiterleiten der korrekten Anfrage
38                 Context.RewritePath(string.Format("~/Default.aspx?width
39                     ={0}&height={1}", width, height));
36             }
37             else
38                 //Ausgabe eines Fehlers
39                 Context.AddError(new Exception("Bitte versuchen sie es
40                     erneut ihre Anfrage enthielt ungültige Parameter"));
41         }
42
43     protected void Application_AuthenticateRequest(object sender,
44         EventArgs e)
45     {
46
47
48     protected void Application_Error(object sender, EventArgs e)
49     {
50
51
52     protected void Session_End(object sender, EventArgs e)
53     {
54
55
56     }
57
58     protected void Application_End(object sender, EventArgs e)
59     {
60
61     }
62 }
63 }
```


Google Web Toolkit

Kathrin Massny

Kloster-Banz-Straße 2, 96052 Bamberg
kathrin.massny@googlemail.com

Abstract. Das Google Web Toolkit ist ein Framework zu Erstellung von Web-Anwendungen. Mit seiner Architektur und seiner Art des Programmierens möchte Google es neuen Durchbruch im Bereich der Webanwendungsentwicklung positionieren. Dieses Paper stellt das GWT vor, erläutert seine Funktionsweisen und Besonderheiten und versucht einen Einstieg in die Erstellung von Web-Anwendungen mit dem Framework zu geben.

Keywords: Architektur, Remote Procedure Calls, Ajax, Widgets, Panels, Events, Compiler, Web-Anwendungen

1 Einleitung

Diese Arbeit beschäftigt sich mit dem Framework¹ „Google Web Toolkit“ (GWT). Sie soll dem Leser näher bringen, was das GWT überhaupt ist und was es von anderen gebräuchlichen Frameworks zur Entwicklung dynamischer Websites abhebt. Auf Grund dessen ist die Arbeit in mehrere Kapitel eingeteilt, die auf die jeweiligen hervorstegenden Funktionalitäten des Frameworks eingehen.

Das folgende Kapitel soll dem Leser einen Blick auf GWT im Ganzen geben und die nötigen Grundlagen für die nachfolgenden Kapitel schaffen. Diese werden angeführt von einem Kapitel über die Kommunikation zwischen Client und Server im GWT. Darauf folgt ein Kapitel über die Erstellung einer Web-Anwendung im GWT und welche Komponenten hierfür unerlässlich sind. Zum Abschluss werden herausragende Funktionen nochmal hervorgehoben und das Framework anhand Vorteile und Nachteile evaluiert.

2 Das Google Web Toolkit

Dieses Kapitel soll einen ersten Eindruck des Google Web Toolkits verschaffen. Es soll Aufschluss darüber geben, was das GWT ist und was es, im Vergleich zu

¹ modernes Rahmenwerk, welches dem Programmierer den Entwicklungsrahmen für die Anwendungsprogrammierung zur Verfügung stellt [4]

anderen Frameworks in der Entwicklung von Web-Anwendungen, so anders macht. Zudem wird auf die Architektur des GWT eingegangen. Einerseits um die Funktionsweise des Frameworks zu erläutern und andererseits um eine Grundlage für die folgenden Kapitel zu schaffen, in welchen auf einige der wichtigen Komponenten im Detail eingegangen wird.

2.1 Was Google Web Toolkit ist

Wirft man einen Blick auf die Entwicklung von Web-Anwendungen, stellt man fest, dass man es mit recht vielen Programmiersprachen zu tun bekommt. Betrachtet man die Client-Seite, stößt man auf HTML-Code. Zusätzlich dazu findet man aber XML- und JavaScript-Code, hinter welchen sich die Logik der Anwendung versteckt [1]. Geht man dann weiter auf die Server-Seite trifft man auf Java, PHP oder Ruby. Kommen hier noch Datenbanken hinzu, erweitert sich das Sprachkontingent um SQL. Betrachtet man die Summe der Sprachen die man für die Entwicklung einer Web-Anwendung benötigt, bekommt man schnell den Eindruck eines sehr komplexen und aufwändigen Unterfangens, besonders wenn man diese Sprachen nicht beherrscht. Als Erschwernis kommt dazu, dass die Web-Anwendungen heutzutage nicht mehr statischer Natur sind, sondern kontinuierlich nach mehr Dynamik streben und so versuchen, traditionelle Desktop-Anwendungen zu ersetzen [1]. Als treffende Beispiele kann man hier die große Anzahl von Browser-Games nennen, die es zumindest soweit geschafft haben, kleine Spielchen für zwischendurch von der Platte zu verbannen und immer und überall verfügbar zu sein. Für die Business-Seite bietet Google mit Google Docs und Google Kalender eine platzsparende und von überall abrufbare Office-Lösung mit der sämtliche Basics eines Office-Pakets abgedeckt sind.

Um die Komplexität aus der Erstellung solcher Web-Anwendungen herauszunehmen, die durch die Anzahl der verschiedenen Programmiersprachen zu Tage kommt, hat Google das GWT entwickelt und Version 1.0 während der JavaOne Konferenz² 2006 in San Francisco [1], der Öffentlichkeit als OpenSource-Framework vorgestellt, welches eine völlig neue Dimension der Web-Anwendungs-Entwicklung eröffnen soll. Googles Anreiz für die Entwicklung des GWT war die Eröffnung einer völlig neuen Dimension der Web-Anwendungs-Entwicklung. Die Idee war, ein Framework zu entwickeln, welches es ermöglicht den Client und den Server in derselben Programmiersprache schreiben zu können und den Client-Code anschließend in valides JavaScript zu konvertieren [7]. Das Google Web Toolkit verinnerlicht diese Idee, denn zum Schreiben der Server- und der Client-Seite benötigt man lediglich Java-Kenntnisse. Seit der Version 1.0 aus dem Jahr 2006 hat Google die bestehenden Funktionalitäten des GWT durch verschiedene Versionen hindurch verbessert und teilweise um Neue erweitert und schließlich im Mai 2011 die aktuelle Version 2.3 veröffentlicht.

Das Google Web Toolkit ist sogar in zwei Ausführungsalternativen verfügbar. Hat man schon viel mit Java und Eclipse gearbeitet, bietet es sich an, das GWT in Eclipse zu installieren. Es ergänzt die IDE um sämtliche notwendigen Funktionen, die es zur Webanwendungs-Erstellung braucht und man kann darin wie gewohnt seinen Java-

² <http://www.oracle.com/javaone/index.html>

Code schreiben. Alternativ dazu kann man sich das GWT auch als Standalone downloaden und seine Webanwendung via Texteditor und GWT-Befehlen durch die Windows-Konsole erstellen. Letzteres erfordert jedoch einiges mehr an Einarbeitungszeit, da die Bedienung hier alles andere als intuitiv ist.

2.2 Das Google Web Toolkit im Detail

In diesem Teil des Kapitels wird auf die Grundbausteine des Google Web Toolkit eingegangen um zum einen Aufschluss über die Architektur des Frameworks zu geben und zum anderen die einzelnen Teilbereiche grundlegend zu erläutern. Hintergrund ist, dass man im Anschluss deren Stellung und Aufgaben innerhalb des GWT und ihre Verbindungen zueinander vor Augen hat, um auf die Folgenden Kapitel eingehen zu können.

Es bietet sich an, das GWT in zwei Hauptbausteine zu teilen, die Entwicklungs- und die Anwendungsseite, da diese Sichtweise für den Einstieg am verständlichsten ist und man dadurch auch schon einen Blick auf den Lebenszyklus einer GWT-Anwendung bekommt.

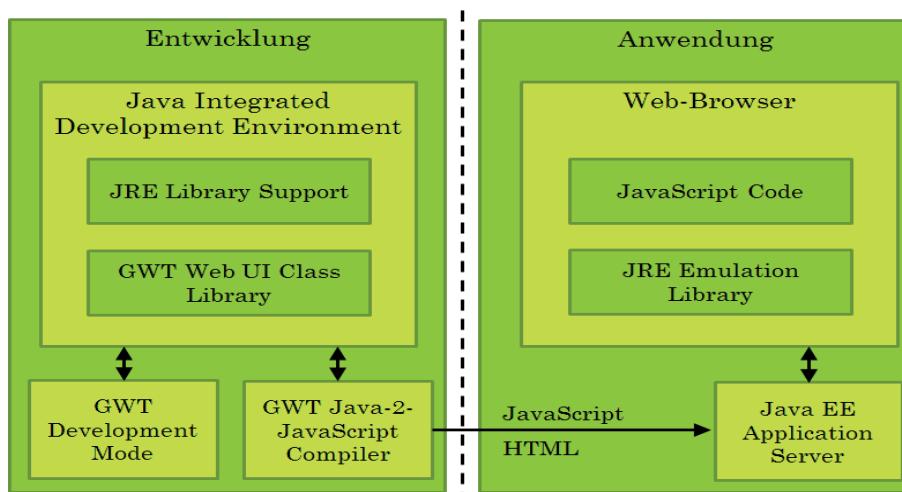


Fig. 1. GWT Architektur [Grafik basierend auf 4]

Wie man in der Grafik bereits sehen kann, besteht das GWT aus mehreren Teilkomponenten. Konzentriert man sich zunächst auf die Entwicklungsseite, erkennt man die drei Hauptbausteine des GWT, die **Java IDE**, den **Development Mode** und den **Java-to-JavaScript-Compiler**. Beginnen wir zunächst mit der Java IDE, der integrierten Entwicklungsumgebung basierend auf der Sprache Java: in diesem Bereich befasst man sich ausschließlich mit dem Schreiben einer Anwendung, und zwar in Java. Nutzt man das GWT als Standalone, rüstet die IDE es zusätzlich mit Compiler, Texteditor und Debugger aus. Installiert man sich das GWT als Eclipse-Plugin sind diese Komponenten bereits vorhanden. Die volle Mächtigkeit von Java als Programmiersprache bekommt man durch den **Java Runtime Environment Library**.

Support, welcher es einem ermöglicht sämtliche Klassen von verfügbaren Java-Paketen zu benutzen. Mit der **Web UI Class Library** stellt das GWT dem Entwickler sämtliche standardmäßige Benutzeroberflächen zur Verfügung. In dieser Bibliothek findet man alle gängigen Arten von Benutzeroberflächen, mit der sich eine Anwendung interaktiv gestalten lässt. Dazu gehören z.B. Buttons, Textfelder, Schaltflächen oder Auswahlfelder [3]. Hat man seine Anwendung fertig geschrieben, führt einen der nächste Schritt zu zwei Bausteinen, die das GWT für die Anwendungsentwicklung so interessant machen.

Der erste Baustein steht eng mit der Java-Programmierung in Verbindung: der Debugger. Hier kann man den Komfort gegenüber JavaScript ausschöpfen, denn der Java-Debugger ist um einiges ausgereifter als der JavaScript-Debugger. Nutzt man das GWT als Eclipse-Plugin, kann man für seine Anwendung den Debugger wie für jeden anderen Java-Code einsetzen. In der Architektur ist er als ein Teil des **Development Mode** anzusiedeln. Dieser Modus ermöglicht es einem, seine Anwendung ausführen zu können ohne sie in JavaScript überführen zu müssen. Startet man eine Anwendung im Development Mode, führt die Java Virtual Machine den Code als kompilierten Java Bytecode aus [2]. Mit Hilfe eines Plugins, welches für jeden gängigen Browser verfügbar ist, kann man seine Anwendung in seinem favorisierten Browser aufrufen. Dafür verwendet man die URL, die einem vom GWT übergeben wird, wenn man seine Anwendung im Development Mode laufen lässt. Ein Punkt, der das Testen und Verbessern von Anwendungen im GWT so schnell und einfach macht, ist, dass Änderungen im Code durch eine Aktualisierung des Browsers sofort angezeigt werden [2]. Gerade wenn man weiß, wie das Testen von Web-Anwendung in anderen Frameworks verläuft, kann man diese Methode als äußerst komfortabel erachten.

Hat man den finalen Stand seiner Anwendung erreicht, kann man diese nun in JavaScript umwandeln lassen. Dafür wurde das Google Web Toolkit mit dem Java-to-JavaScript-Compiler ausgestattet. Da der Compiler es erst möglich macht die Anwendung für den späteren Gebrauch lauffähig zu machen, kann man ihn als wichtigste Komponente des GWT bezeichnen. Er transformiert die in Java geschriebene Anwendung in eine gleichwertige JavaScript-Anwendung [2]. Da Java und JavaScript nicht 1:1 zu übersetzen sind, steht der Compiler in enger Verbindung mit der **JRE Emulation Library** von der Anwendungsseite. Diese Bibliothek gibt an welche Klassen & Methoden aus welchen Paketen in JavaScript überführbar sind um eine voll funktionsfähige Anwendung zu erhalten. Ein Beispiel für eine dieser Restriktionen ist, dass JavaScript für Gleitkommazahlen lediglich einen 64-bit typ unterstützt. Dies kann, gegenüber Java, in der Anwendung von arithmetischen Operationen zu Präzisionsverlust im Ergebnis führen [5]. Auch die Exceptions in Java und JavaScript sind unterschiedlich, so dass man darauf achten muss, dass diese im Kompilierten Zustand genauso greifen wie im Development Mode [5]. Um diese Restriktionen schon bei der Programmierung beachten zu können, stellt Google eine GWT JRE Emulation Reference³ zur Verfügung, in welcher sämtliche Pakete, Methoden und Datentypen aufgelistet sind, die das GWT automatisch übersetzen kann [2]. Falls man doch noch problematischen Code in seiner Anwendung stehen

³ <http://code.google.com/intl/de-DE/webtoolkit/doc/2.0/RefJreEmulation.html>

hat, bekommt man vom Compiler eine Fehlermeldung mit Hinweisen auf die problematischen Stellen.

Jedoch kann der Compiler noch mehr als nur den Java-Code in JavaScript zu übersetzen. Zum Beispiel kann man vor dem Kompilieren festlegen wie detailliert der JavaScript-Code am Ende aussehen soll. Hierfür gibt es drei verschiedene Stile: *obfuscated*, *pretty* und *detailed*. Standardmäßig wird der JavaScript-Code in der *obfuscated*-Variante kompiliert, da er den Code auf die kleinstmögliche Größe komprimiert und so Speicherkapazität spart. In der nachfolgenden Grafik kann man sehen, dass der Code jedoch soweit runterkomprimiert wurde, dass er unlesbar ist und man aus ihm nicht schließen kann, wofür die Methode gedacht ist und was sie konkret ausführt [3]:

```

1
2   function f(a){if(a.match(/^\w+:\/\/)){}else{var b=m.createElement(_);b.src=a;ab=a=e(b.src)}return a}
3

```

Fig. 2. Compiler-Stil obfuscated

Wählt man den Stil *pretty*, sieht der Code, den wir uns eben im *obfuscated*-Stil betrachtet haben, wie folgt aus:

```

1
2   function ensureAbsoluteUrl(url) {
3     if (url.match(/^\w+:\/\/)) {
4       ...
5     } else {
6       var img = $doc_0.createElement('img');
7       img.src = url + 'clear.cache.gif';
8       url = getDirectoryOfFile(img.src);
9     }
10    return url;
11  }

```

Fig. 3. Compiler-Stil pretty

Wie man erkennen kann, hat die Methode nun einen Namen und man weiß was sie übergeben bekommt. Ebenso ergeben die einzelnen Variablen und Zuordnungen mehr Sinn und man kann sich schon eher vorstellen was diese Methode ausführt. Der Stil *detailed* ist zum Stil *pretty* nahezu identisch. Unterschiede gibt es nur noch in der Tiefe der Details, indem es z.B. einen vollständigen Klassennamen angibt und keine Abkürzungen, wie sie bei *pretty* verwendet werden [3].

Abgesehen von dieser Einstellung, kann man den Compiler-Vorgang optimieren. Beispielsweise kann man einstellen, dass lediglich in der Anwendung verwendeter Code kompiliert und alles andere außen vor gelassen wird [5]. Erstellt man sich also eine Klasse mit einer großen Anzahl verschiedener Methoden, nutzt in seiner Anwendung allerdings nur eine Handvoll, wird eben nur diese Handvoll kompiliert und man erspart sich dadurch einen großen Speicherplatzverbrauch [5].

Hat der Compiler schließlich alles fertig übersetzt, wandern wir auf die Anwendungsseite, indem dieser die fertige JavaScript-Anwendung und den HTML-

Code der Seite an den Server schickt. Dieser wiederum, steht mit den Browsern, die die Anwendungen ausführen, in Verbindung. Weiterhin finden wir auf der Anwendungsseite die JRE Emulation Library, die dem Compiler vorgibt welchen Java-Code er in JavaScript umwandeln kann.

3 Die Client-Server-Kommunikation

Da wir nun wissen, was das Google Web Toolkit ist und wie es im Groben funktioniert, möchte ich in diesem Kapitel näher auf den Punkt der Kommunikation zwischen Client, also dem Browser, und dem Server, der alle Anwendungsinformationen bereithält, eingehen.

Zunächst ist die Frage zu klären, warum die Client-Server-Kommunikation denn einen so wichtigen Stellenwert innerhalb des GWT's einnimmt. Die Antwort hierauf ist recht einfach zu beantworten: jedes Mal wenn ein Benutzer eine Interaktion auf dem Browser ausführt, muss dieser ggf. Daten an den Server schicken und bekommt von diesem wiederum neue Seiteninformationen zurück. In Zeiten statischer Anwendungen und Webseiten, musste der Browser nach jeder Interaktion die gesamte Webseite neu laden, um Inhaltsänderungen anzeigen zu können. Das heißt, eine Anfrage mit Änderungswerten wurde an den Server geschickt, diese dort verarbeitet und als Antwort zurückgeschickt [6] (siehe Fig. 4). Diese Verarbeitungsweise war recht Zeitraubend und hatte einen recht hohen Bedarf an Bandbreite.

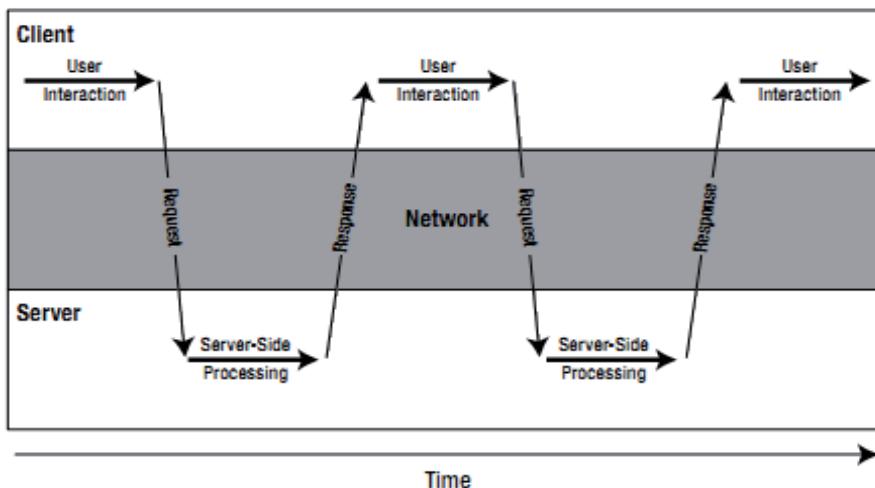


Fig. 4. Statische Client-Server Kommunikation [6]

Heute läuft die Kommunikation, dank **Ajax (Asynchronous JavaScript and XML)**, viel flüssiger und ressourcenschonender, da der Browser nur noch nachlädt, was tatsächlich verändert werden muss [8] (siehe Fig. 5). GWT-Anwendungen basieren genau auf diesem Prinzip der Kommunikation und bieten, je nachdem

welche Sprache der Server im Backend unterstützt, zwei verschiedene Arten der Client-Server-Kommunikation [2].

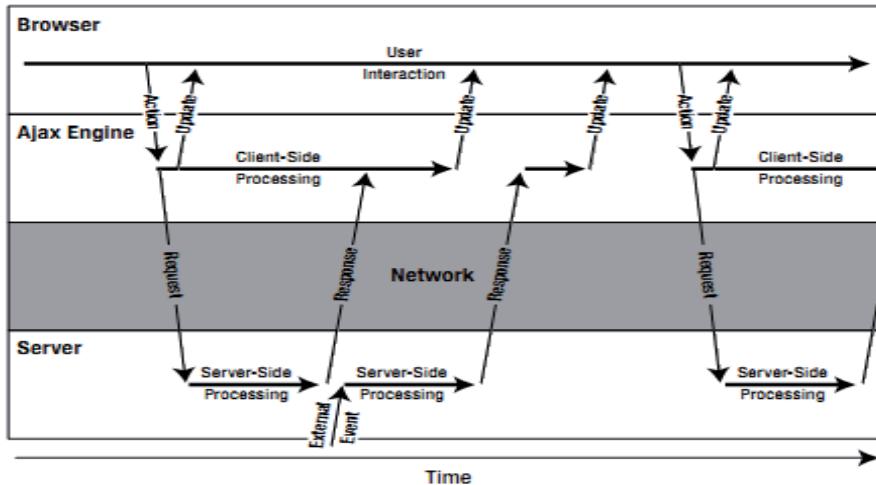


Fig. 5. Asynchrone Client-Server Kommunikation [6]

Zum einen kann man die Standard-Ajax-Kommunikation nutzen. Die Kommunikation zwischen Browser und Server wird hier über ein spezielles JavaScript-Objekt namens XMLHttpRequest getätigt [3]. Es ist ein Interface, welches den Clients ermöglicht über HTTP-Aufrufe Anfragen zu senden und zu empfangen [7]. Diese Art der Kommunikation ist sinnvoll, wenn man mit älteren Servern, die kein Java unterstützen, kommunizieren muss oder mit Servern, die auf anderen Sprachen wie Ruby oder PHP basieren [3]. Die andere Kommunikationsmöglichkeit richtet sich an die Java-Anwendungsserver und ist die für das GWT geeignetere Variante, da sie die Client- und Server-Seite durch die gemeinsame Sprache vereint. Es nennt sich **Remote Procedure Call (RPC)** und ist auf der Basis-Ajax-Kommunikation aufgebaut. Es erlaubt dem Programmierer, Kommunikationsanfragen mit Hilfe von Methoden innerhalb eines Java-Interfaces aufzurufen [6]. Die folgenden Absätze des Kapitels gehen näher auf die Erstellung eines solchen RPC innerhalb des GWT ein und werden diese durch Code-Beispiele etwas erläutern. Der Code stammt aus dem Default-Code, der zu Demonstrationszwecken in jedem neuen Eclipse-GWT-Projekt erstellt wird. Die Anwendung fordert den User auf, in ein Textfeld seinen Namen einzutragen und abzuschicken. Als Antwort bekommt dieser anschließend ein Begrüßungsfenster mit Informationen dazu, welche Daten an den Server gesendet wurden, und was der Server geantwortet hat.

3.1 Definition eines Service-Interface

Das Service-Interface stellt im übertragenen Sinne eine Vereinbarung zwischen Client und Server dar. Diese gibt ihnen an welche Methoden aufgerufen werden

können und welche Werte sie zurückverlangen können [6]. Jedes Service-Interface muss vom Interface RemoteService ableiten, da dieses dem Compiler den Hinweis gibt, dass es sich um ein RPC-Interface handelt und nicht um eine gewöhnliche Java-Datei [6].

```

1 package de.seminar.GWT.client;
2
3 import com.google.gwt.user.client.rpc.RemoteService;
4 import com.google.gwt.user.client.rpc.RemoteServiceRelativePath;
5
6
7 public interface GreetingService extends RemoteService
8 {
9     String greetServer(String name) throws IllegalArgumentException;
10 }
11

```

Fig. 6. Code eines Service-Interface auf der Client-Seite

Da das Interface von der Client-Seite der Anwendung aus arbeitet, muss es bei den anderen Client-seitigen Codes platziert werden. In Fig. 6 sieht man nun das Interface GreetingService welches den Aufruf der Methode greetServer spezifiziert.

3.2 Service Implementation

Auf der Server-Seite nutzt der GWT-RPC Servlets. Servlets sind Teil der Java EE-Spezifikation⁴ und machen dynamische Inhalte im Web verfügbar [6]. Sie sind Erweiterungen von Web-Containern und machen es für diesen möglich Anfragen zu empfangen und ausführbare Antworten für diese zu erstellen. Im GWT ist Tomcat⁵ dieser Web-Container [2]. Um einen Service auf der Server-Seite auch als Servlet erkennbar zu machen, muss diese von der Klasse RemoteServiceServlet ableiten. Außerdem muss es das Service-Interface der Client-Seite implementieren um alle Methoden verfügbar zu haben [2]. Das RemoteServiceServlet ermittelt alle aufzurufenden Methoden und führt diese auch aus.

In Fig. 7 sieht man nun die Implementierung des Interface und was die Aufgabe der Methode ist. Sie überprüft die Eingabe, die der User beim Aufrufen der Anwendung macht und gibt ggf. eine Fehlermeldung zurück. Im weiteren Verlauf holt sich die Anwendung Informationen zum Server und zum benutzten Browser des Users und speichert sie in Variablen. Mit dem return-Befehl gibt der Server dem Nutzer eine persönliche Begrüßung aus und gibt an welchen Server er für die Anwendung benutzt und mit welchem Browser der Nutzer gerade wohl im Netz unterwegs sein könnte.

⁴ <http://www.prozesse-und-systeme.de/j2eeIntroAndCriticism.html>

⁵ In Java geschriebener Servlet-Container der eine Umgebung von Java-Code auf Webservern bereitstellt - <http://tomcat.apache.org/>

```

1 package de.seminar.GWT.server;
2
3 import de.seminar.GWT.client.GreetingService;
4 import de.seminar.GWT.shared.FieldVerifier;
5 import com.google.gwt.user.server.rpc.RemoteServiceServlet;
6
7
8 public class GreetingServiceImpl extends RemoteServiceServlet implements GreetingService {
9
10    public String greetServer(String input) throws IllegalArgumentException {
11        if (!FieldVerifier.isValidName(input)) {
12            throw new IllegalArgumentException("Name must be at least 4 characters long"); }
13
14        String serverInfo = getServletContext().getServerInfo();
15        String userAgent = getThreadLocalRequest().getHeader("User-Agent");
16
17        input = escapeHtml(input);
18        userAgent = escapeHtml(userAgent);
19
20        return "Hello, " + input + "!<br><br>I am running " + serverInfo
21                + ".<br><br>It looks like you are using:<br>" + userAgent;
22    }
23
24    private String escapeHtml(String html) { }
25
26
27
28
29
30 }
```

Fig. 7. Code der Service-Implementation auf der Serverseite

3.3 Implementierung der Client-Seite

Um das Servlet nun auch nutzen zu können, benötigen wir schließlich noch eine asynchrone Version unseres bereits bestehenden Service-Interface. Der Unterschied zum ersten Interface sind die asynchronen Rückgabewerte [5]. Im Moment hat der Client keinen Anhaltspunkt, was er mit den Werten, die er vom Server zurückbekommt, tun soll. Diesen Anhaltspunkt soll das asynchrone Interface beinhalten um diesen ausführen zu können wenn die Anfrage an die Service-Methode zurückkommt [6]. Dieser Anhaltspunkt wird durch einen clientseitigen CallbackHandler realisiert. Dieser wird in der JavaScript-Engine registriert um nach Beendigung der Servermethode den User benachrichtigen zu können [7]. Das Interface muss das `com.google.gwt.user.client.rpc.AsyncCallback`-Paket importiert haben, da der Callback-handler von diesem Typ sein muss [7]. Im Falle unseres Beispiels, erscheint bei erfolgreicher Durchführung RPC ein Fenster mit allen Informationen, die im Servlet festgelegt wurden (siehe Fig. 9).

```

1 package de.seminar.GWT.client;
2
3 import com.google.gwt.user.client.rpc.AsyncCallback;
4
5 public interface GreetingServiceAsync {
6     void greetServer(String input, AsyncCallback<String> callback) throws IllegalArgumentException;
7 }
8
```

Fig. 8. Code des asynchronen Service-Interface auf der Client-Seite

In einer XML-Datei, die sich im selben Ordner wie die HTML-Datei befindet, werden alle schließlich alle Servlets und die dazugehörigen Servlet-Klassen eingetragen.

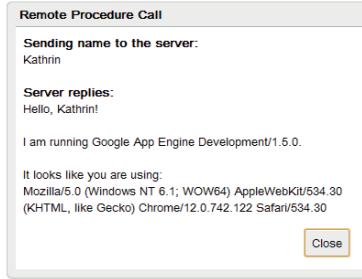


Fig. 9. Ergebnis des Remote Procedure Calls der Methode greetServer

4 Die wichtigsten Bestandteile einer GWT – Anwendung

In diesem Kapitel werfen wir einen Blick auf die wichtigsten Bestandteile einer Web-Anwendung im GWT. Da Web-Anwendungen sehr stark auf Benutzer-Interaktion aufbauen, wird sehr viel auf die **Web User Interface Library** eingegangen. Ihre Inhalte sind es, die Interaktivität einer Anwendung erst möglich machen. Um die Erklärungen ein bisschen beispielhafter darzustellen, wird es an angemessenen Stellen Code-Beispiele geben.

4.1 EntryPoint

Jede Anwendung braucht einen Startpunkt der vorgibt, was wie ausgeführt wird. In Java-Anwendungen wird dies durch die Main-Methode realisiert. Im GWT wird diese Main-Methode wiederum durch den **EntryPoint** realisiert. Die Klasse, die den Einstiegspunkt einer Anwendung bietet, muss das Java-interface **EntryPoint** implementieren und die damit verbundene Methode **onModuleLoad()** bereitstellen [7]. In dieser Methode werden alle Startaktivitäten der Anwendung festgelegt. Im Beispiel-Code startet die Klasse Pairs ein Memory-Spiel indem es ein Spielbrett erstellt (siehe Fig. 11).

```

11  public class Pairs implements EntryPoint
12  {
13
14      Button startButton = new Button("Let's Go");
15
16      public void onModuleLoad()
17      {
18
19      }
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34 }
```

Fig. 10. Code der Einstiegsklasse Pairs mit EntryPoint und onModuleLoad () - Methode [9]

4.2 Widgets

In Fig. 10 kann man sehen, dass in der Klasse ein Button definiert wird. Solche Buttons gehören im GWT zu den Kontrollelementen namens **Widgets**. Sie stellen die

Komponenten einer Web-Anwendung dar, die für den Benutzer auf dem Browser sichtbar sind. Zu ihnen gehören Buttons, Textboxen, Listboxen und viele weitere [3]. Ohne solche Widgets wäre eine Web-Anwendung nutzlos, denn durch sie wird die Interaktivität einer Anwendung realisiert [3]. Alle im GWT verfügbaren Widgets sind durch die **Web User Interface Library** implementierbar und können in der Google Widget List⁶ inklusive Erläuterungen eingesehen werden.

Man ist jedoch nicht ausschließlich auf diese vorgegebenen Widgets beschränkt. Alle Widgets aus der Web UI Library lassen sich nach Belieben verändern. Beispielsweise kann man sie durch eine CSS-Datei neu designen.

Um Widgets interaktiv einsetzen zu können, müssen sie allerdings auf Aktionen vom Browser reagieren können. Aus diesem Grund wird für jedes Widget definiert, auf welche Aktion es reagieren soll und was es tun soll, wenn eine solche Aktion ausgelöst wird. Für diesen Vorgang stellt das GWT EventHandler zur Verfügung, die den Widgets angehängt werden. Im Falle des Buttons handelt es sich um einen ClickHandler der zum Button hinzugefügt wird und im weiteren Verlauf in der Methode onClick(ClickEvent event) festlegt was der Button zu tun hat wenn der Nutzer mit der Maus auf ihn klickt (siehe Fig.11). Welche Eventhandler und Events für welches Widget zu Verfügung stehen, kann man den Widget-Informationen in der Widget Gallery entnehmen.

```

11  public class Pairs implements EntryPoint
12  {
13
14      Button startButton = new Button("Let's Go");
15
16      public void onModuleLoad()
17      {
18          RootPanel.get("content").add(startButton);
19
20          startButton.addClickHandler(new ClickHandler() {
21
22              public void onClick(ClickEvent event)
23              {
24                  DialogBox dialog = new DialogBox();
25                  Spielbrett spiel = new Spielbrett();
26                  dialog.setText("Pairs");
27                  dialog.setModal(false);
28                  dialog.center();
29                  dialog.show();
30                  dialog.add(spiel);
31              }
32          });
33      }
34  }

```

Fig. 11. Anfügen eines EventHandlers mit Anweisungsmethode für den Event [9]

4.3 Panels

Neben der Erstellung von Interaktionsmöglichkeiten, bietet die Web UI Library auch eine Möglichkeit zum Gestalten der Anwendung. Dies wird dem Entwickler durch Panels ermöglicht. Durch Panels können Widgets in der Anwendung da positioniert

⁶ <http://code.google.com/intl/de-DE/webtoolkit/doc/latest/RefWidgetGallery.html>

werden, dass die Anwendungsfunktionalität sinnvoll bereitgestellt werden kann [3]. Zudem kann durch sie gewährleistet werden, dass bestimmte Komponenten der Anwendung erst auftauchen wenn sie tatsächlich gebraucht werden.

In Fig. 11 kann man sehen, dass dort bereits ein Panel eingefügt wurde: das RootPanel. Es dient als Schnittstelle zwischen Anwendung und zur im Browser angezeigten Webseite [3]. Aus diesem Grund gibt es in jeder Anwendung mindestens ein RootPanel. Wie man im Code sehen kann ruft das RootPanel via getRequest eine ID namens „content“ auf. Diese ID verweist in diesem Fall auf ein div-Tag in der HTML-Datei der Webseite. Durch .add(startButton) wird der vorher generiert Button der Schnittstelle hinzugefügt und so die Anwendung in die Webseite eingebunden.

```

11  public class Spielbrett extends Composite
12  {
13      FlowPanel panel = new FlowPanel();
14      int karten = 8;
15      Karte ersteKarte=null;
16      int pairs = 0;
17      public Spielbrett(){
18          panel.setPixelSize(400,400);
19          initWidget(panel);
20          newGame();
21      }

```

Fig. 12. Beispiel eines weiteren Panels [9]

Im Beispiel von Fig. 12 sehen wir ein weiteres Panel, ein FlowPanel. In diesem Panel werden die Komponenten von links oben bis nach rechts unten nebeneinander eingereiht. Für ein Memory-Spiel das perfekte Panel um das Spielbrett mit 4 x 4 Karten darzustellen. Mit dem Befehlt .setPixelSize(400,400) wird die Größe des Panels festgelegt. Die Felder des Panels werden wie bei einem Array mit feld[i] angesprochen.

Welche Panels das GWT noch zur Verfügung stellt, kann ebenfalls in der Widget Gallery⁷ nachgelesen werden.

4.4 Composite

Auch die Klasse Composite kann als Panel bezeichnet werden. Es ist jedoch ein sehr besonderes Panel und erhält auf Grund dessen einen eigenen Absatz. In Fig. 12 mag bereits aufgefallen sein, dass die Klasse Spielfeld von der Klasse Composite ableitet. Zum besseren Verständnis, wozu das Composite da ist, eine Erklärung am Beispiel des Memory-Spiels (siehe Code Fig.11). In der Klasse Pairs wird in der Anwendungsmethode für den Button ein Widget DialogBox erstellt in welcher das Spielfeld des Memory erscheinen soll. Nun besteht das Spielfeld selbst wiederum auch aus Panels und Widgets, einem Widget kann allerdings nur ein Widget hinzugefügt werden. Hier springt die Klasse Composite nun ein. Leitet man eine Java-Klasse von Composite ab, umhüllt diese die Java-Klasse so, dass ihre Methoden

⁷ <http://code.google.com/intl/de-DE/webtoolkit/doc/latest/RefWidgetGallery.html#Panels>

von außerhalb nichtmehr erkennbar sind [7]. Die Klasse wird dadurch wie ein Widget behandelt und kann so in die DialogBox eingebettet werden. Der Inhalt des neuen Widgets wird mit Hilfe des Befehls `initWidget(panel)` festgelegt. Dadurch wird aus dem FlowPanel, welches das Spielfeld darstellt, das neue Widget [7]. Auf diese Weise können sämtliche Java-Klassen einer Anwendung in Widgets umgewandelt und ungehindert in die Anwendung eingebettet werden.

5 Schlusswort

Nachdem das Google Web Toolkit nun allgemein vorgestellt wurde und die wichtigsten und herausragendensten Funktionen des Frameworks näher betrachtet wurden, möchte ich im abschließenden Teil der Arbeit noch einmal auf die Vor- und Nachteile des GWT eingehen und ein persönliche Einschätzung zu diesem Framework abgeben.

Zunächst möchte ich auf die Punkte eingehen, die allgemein als Nachteile angegeben werden und welche ich selber als nachteilig erachte. Zum einen ist da die äußerst lange und komplizierte Einarbeitung in das Framework. Diese tritt vor allem auf wenn, man das GWT als Standalone nutzen möchte. Einführende Literatur deckt für diese Variante meist nur die Versionen 1.4 und 1.5 ab. Möchte man mit der neusten Version 2.3 arbeiten, hat man recht wenig Literatur und das Google Tutorial zur Verfügung. Dieses ist jedoch eher weniger für Einsteiger geeignet. Für jeden, der neu ins GWT einsteigen möchte, sind die Tutorials aus [9] äußerst zu empfehlen. Zudem ist das GWT eher dafür geeignet neue Projekte und Anwendungen zu schreiben. Ein bereits bestehendes Projekt um GWT-Anwendungen zu erweitern könnte durch die Unterschiede zu anderen Frameworks und die Grundlage von Java etwas komplizierter werden.

Was ich bei der Arbeit mit dem GWT sehr positiv empfunden habe, war wie Einbindung in Eclipse und der äußerst komfortable Development Mode. Dieser könnte zwar etwas schneller beim Start sein, jedoch macht er dieses Manko durch die Aktualisierungsfunktion wieder wett. Sehr angenehm war auch die Anwendungsentwicklung in Java. Gerade wenn man Neuling im Bereich der Webanwendungserstellung ist und eine schon bekannte Sprache nutzen kann, kann dies von Vorteil sein, da man sich lediglich mit der Anwendung selber beschäftigen muss und nicht noch zusätzlich mit einer neuen Sprache. Für Entwickler, die Anwendungen seit je her in JavaScript entwickeln ist das GWT eher ungeeignet. Für Entwickler von Software in Java könnte es interessanter sein, jedoch glaube ich nicht, dass es denn von Google erwünschten Status einer neuen Ära in der Entwicklung von Webanwendungen erreichen wird.

Abschließend kann man sagen, dass das GWT ein recht umfangreiches Tool ist, welches einige Zeit Einarbeitung fordert, um wirklich gute komplexe Web-Anwendungen damit entwickeln zu können, jedoch hat es mir nach einigen Anlaufschwierigkeiten doch Spaß gemacht damit zu arbeiten.

6 Literaturverzeichnis

- [1] Burnette, Ed (2006), "Google Web Toolkit – Taking the pain out of Ajax", The Pragmatic Programmers, LLC, www.pragmaticprogrammer.com (2006)
- [2] Google Code, Google Web Toolkit 2.3 Tutorial, <http://code.google.com/intl/de-DE/web toolkit/doc/latest/tutorial/>
- [3] Hanson, Robert; Tacy, Adam (2007), "GWT im Einsatz – Ajax-Anwendungen entwickeln mit dem Google Web Toolkit", Carl Hanser Verlag München Wien (2007)
- [4] Quelle: IT-Wissen – Das große Online-Lexikon für Informationstechnologie: <http://www.itwissen.info/definition/lexikon/GWT-Google-web-toolkit-Google-Web-Toolkit.html>
- [5] Kereki, Federico (2010), "Essential GWT: Building for the Web with Google Web Toolkit 2", Addison-Weasly Professional (2010)
- [6] Smeets, Bram; Uri, Boness; Bankras, Ronald (2008), "Beginning Google Web Toolkit: From Novice to Professional", Apress, www.apress.com (2008)
- [7] Sowa, Hans; Radinger, Wolfgang; Marinschek, Martin (2008) "Google Web Toolkit – Ajax-Anwendungen einfach und schnell entwickeln" dpunkt.verlag Heidelberg (2008)
- [8] Steyer, Ralph (2007), „Google Web Toolkit – Ajax-Applikationen mit Java“, entwickler.press, ein Imprint der Software & Support Verlag GmbH, www.entwickler-press.de (2007)
- [9] Code wurde auf Basis des Video-Tutorials „GWT-Tutorial #2-Unser erstes Projekt“, von Tobias Sell erstellt. Video & Code können hier betrachtet werden: <http://selfisch.de/?p=8>