



Technische Universität Braunschweig
Institut für Wirtschaftsinformatik

Teamprojekt SoSe 2009

Entwicklung einer mobilen Webapplikation



Dominik Schürmann
Philipp Wille
Sergej Dechand

10. Juni 2010

*Dieses Werk ist unter einem Creative Commons Namensnennung-Keine kommerzielle Nutzung 3.0 Deutschland Lizenzvertrag lizenziert. Um die Lizenz anzusehen, gehen Sie bitte zu <http://creativecommons.org/licenses/by-nc/3.0/de/> oder schicken Sie einen Brief an Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Projektvorstellung	1
1.2	Vorgehensweise	1
2	Technische Voraussetzungen	3
2.1	Android	3
2.1.1	Betriebssystem	3
2.1.2	Geräte	3
2.1.3	Internetbrowser	4
2.2	iPhone	4
2.2.1	Betriebssystem	5
2.2.2	Geräte	5
2.2.3	Internetbrowser	6
2.3	WebKit Engine	7
3	Web-Standards	8
3.1	W3C Best Practices	8
3.1.1	Mobile Web Best Practices	8
3.1.2	W3C Validator	9
3.1.3	Kritik	9
3.2	DotMobi Top-Level-Domain	10
3.2.1	dotMobi Domain	11
3.2.2	Webtools	11
3.3	Apple iPhone Recommendations	13
3.3.1	Safaris funktionale Einschränkungen auf dem iPhone	13
3.3.2	Fokussierte Mobilität und Benutzerwahrnehmung	13
3.3.3	Vermittelte Leichtigkeit	14
3.3.4	Konkrete Tipps zur Umsetzung	15
4	Evaluation	16
4.1	Beurteilung der Standards	16
4.2	Umsetzung	16
5	UI Toolkits	17
5.1	iWebKit	17
5.1.1	Überblick	17
5.1.2	Aussehen	17
5.1.3	Lizenz	17
5.1.4	Bewertung	18
5.2	iUi	19
5.2.1	Überblick	19
5.2.2	Aussehen	19

5.2.3	Lizenz	19
5.2.4	Bewertung	20
5.3	Eigenes Toolkit	21
5.3.1	Überblick	22
5.3.2	Aussehen	22
5.3.3	Lizenz	22
5.3.4	Bewertung	23
5.4	Entscheidung	24
6	Design	25
6.1	Grundlagen	25
6.2	Navigation	25
6.3	Seitenaufteilung	26
6.4	Vorschau	26
7	Prototyp	27
7.1	Auftretende Probleme	27
7.2	Entscheidung	28
7.3	Umsetzung	28
8	Zusammenfassung	30
9	Abbildungsverzeichnis	31
10	Literatur	31
11	Glossar	38
12	Anhang	41
12.1	Auszug vom Prototypen	41
12.1.1	index.html	41
12.1.2	style.css	43
12.1.3	android.css	44

1 Einleitung

Im Rahmen dieses Teamprojekts des Instituts für Wirtschaftsinformatik werden wir die Möglichkeiten und Standards hingehend einer mobilen Webapplikation evaluieren und einen Prototyp entwickeln. In unserem Konzept wollen wir aufzeigen, was durch Einsatz von bereits vorhandenen Standards und unserem individuellen Verständnis eine solche mobile Webapplikation auszeichnet. Diese Erkenntnisse werden wir in das Projekt einfließen lassen, das den Praxisbezug zu diesem Konzeptes darstellt.

1.1 Projektvorstellung

Die Webapplikation soll das Gelände der Technischen Universität Braunschweig (TU BS) auf einer „OpenStreetMap“ basierenden Karte für ein mobiles Endgerät zur Verfügung stellen und diese Karte mit einer Wiki kombinieren, in der Studierende Informationen hinterlegen können. Diese Informationen können beispielsweise aktuelle Raumbelagungen sein, oder auch aktuelle Neuigkeiten sowie Eventankündigungen zu Plätzen und Räumen. Aus anderen Quellen verfügbare Daten wie Raumbelagungen, sollen möglichst als Initialbefüllung der Wiki dienen und von Studierenden auf dem neusten Stand gehalten werden. Wir überlegen uns in unserem Teamprojekt gezielt, wie eine solche Webanwendung für mobile Applikationen optimiert werden kann, welche Webstandards für den mobilen Bereich existieren und entwickeln einen Design-Prototypen, der sinnvolle Standards und Usability vereinen soll.

Die serverseitige Verarbeitung ist für unser Teamprojekt irrelevant, da wir uns erstmal nur auf die clientseitige Bedienung konzentrieren und auf dem Server ein beliebiges Web Framework wie z.B. Ruby on Rails oder Grails auf Java Basis laufen kann.

Auf dem mobilen Gerät liegt das besondere Augenmerk auf der Konzeption einer eingängigen Benutzeroberfläche, die für Projekte im Rahmen des Studierendenportals der TU BS weiterverwendet werden kann.

1.2 Vorgehensweise

Im Vorhinein zeigen wir kurz die technischen Voraussetzungen auf, also die aktuell schon in der Praxis eingesetzten Technologien, um dann in die Sichtung schon vorhandener Web-Standards überzugehen. Wir beschränken uns hierbei auf drei ausgewählte für unser Projekt relevante Standards.

Nach dieser Übersicht werden wir in der Evaluation Schlüsse aus den Web-Standards für unser Projekt ziehen und deren Umsetzung diskutieren.

In den nächsten beiden Kapiteln geht es dann um das Design der Applikation, das Mithilfe eines „User Interface Toolkits“ (UI Toolkits) einfacher für mobile Endgeräte optimiert werden kann. Nach der Ergründung der sich durch die Toolkits ergebenden Möglichkeiten, stellen wir sie zur Diskussion und wägen ihre Vor- und Nachteile gegeneinander ab. Am Ende der Toolkit Evaluierung soll schließlich ein Favourit stehen, den wir dann zur Umsetzung des Prototypen heranziehen.

Hiernach werden wir eine Analyse des strategisch günstigen Aufbaus der Webseite

im Bezug auf Seitenaufteilung und Navigation anführen bevor wir zur Entwicklung des Prototypen kommen.

2 Technische Voraussetzungen

Um einen Überblick über die relevanten technischen Voraussetzungen im Hinblick auf mobile Webapplikationen zu bekommen, werden in diesem Kapitel die beiden aktuell wichtigsten Vertreter der mobilen Betriebssysteme und die zumeist eingesetzte Rendering-Engine für mobile Webseiten vorgestellt. Wir werden hierbei vordergründig auf die Aspekte bezüglich unseres Themas eingehen um ein Grundverständnis der eingesetzten Technik zu vermitteln.

2.1 Android

Bei Android handelt es sich um ein Betriebssystem für mobile Endgeräte, das federführend von Google entwickelt wurde und unter einer freien Lizenz (Apache-Lizenz 2.0) verändert und verwendet werden darf. So wurde es in Deutschland erstmalig in einer Kooperation von Google mit T-Mobile auf dem Handy „T-Mobile G1“ vertrieben. Da es frei verwendet werden kann haben viele Hersteller weitere Endgeräte mit dem Android Betriebssystem angekündigt und ihre eigenen Ideen mit eingearbeitet.

2.1.1 Betriebssystem

Das Betriebssystem setzt auf dem Linuxkernel 2.6 mit einigen Userspace Programmen auf, unterstützt echtes Multitasking und bietet ein umfangreiches „Application Programming Interface“ (API) um Programme für Android in Java zu entwickeln. Auf dem System kommt eine besondere Laufzeitumgebung für Java Programme zum Einsatz, die sich „Dalvik“ nennt. Aus dieser Laufzeitumgebung kann komfortabel auf darunterliegende Bibliotheken, wie Webbrowser-Engine sowie Grafikbibliotheken, und kernernahe Programme zugegriffen werden indem einfache Schnittstellen zur Verfügung stehen, die eine Abstraktionsschicht für die Gerätefunktionen darstellt. [Andb]

Um Applikationen für Android zu entwickeln genügt es, das freie „Android-Software Development Kit“ (Android-SDK) [Andc] herunterzuladen, das von einem Android-Emulator bis hin zur Dokumentation alles enthält was man benötigt. Da das komplette System freie Software ist, also unter einer von der Free Software Foundation als frei anerkannten Open Source-Lizenz steht, unterstützt es natürlich die unterschiedlichsten Hardwareelemente.

Bei der Art der Eingabe wäre das zum Beispiel Multitouchbildschirm, Tastatur oder auch Trackball. Es bleibt dem Hersteller des Gerätes überlassen welche Hardware er mit dem Betriebssystem kombiniert, vorausgesetzt er lässt dafür einen Treiber für Linux schreiben. Das Betriebssystem selber entwickelt sich somit immer weiter, da Google, viele Firmen, die sich in der Open Handset Alliance zusammengefunden haben, und freie Programmierer den offenen Quellcode verbessern.

2.1.2 Geräte

Da Android nicht an eine feste Produktlinie eines Herstellers gebunden ist, gibt es eine ganze Reihe von schon erschienenen Geräten und angekündigten Geräten, die nicht nur

Handys umfasst, sondern auch Netbooks, Küchengeräte, Fernbedienungen und sogar Bilderrahmen [Anda].

Stellvertretend wird hier auf das „T-Mobile G1“ eingegangen, das von der Hardwareausstattung stellvertretend für die schon erschienenen Android Handys steht. Das „T-Mobile G1“ ist von der Firma „High Tech Computer Corporation“ (HTC) als erstes Android Handy auf dem Markt erschienen und beinhaltet alle wichtigen Geräteeigenschaften eines modernen Handys. Es unterstützt sowohl „Global System for Mobile Communications“ (GSM) als auch „Universal Mobile Telecommunications System“ (UMTS), was für ein schnelles Internetvergnügen unterwegs unabdingbar ist. Desweiteren unterstützt es Wi-Fi und Bluetooth als drahtlose Übertragungsmethoden und hat ein „Global Positioning System“ (GPS) Empfänger sowie einen Kompass, Beschleunigungssensoren und einen Lagesensor verbaut. Als Eingabemethoden bieten sich entweder die ausschiebbar Tastatur, der Touchscreen oder der Trackball an. Das Herz des Gerätes bilden ein mit max. 384 MHz getakteter ARM-Prozessor, 192 MB RAM und 256 MB Speicher, der mit einer microSD erweitert werden kann. Die Schwächen bei diesem ersten Android Gerät liegen in der relativ schlechten 3.2 Megapixel VGA-Kamera und der geringen Akkulaufzeit. Die Bildschirmauflösung und somit auch die Darstellung von Webseiten beträgt 480 x 320 Pixel, ist also identisch mit der des iPhones [HTCa]. Wie durch unterschiedliche Hersteller angekündigt, werden die Nachteile dieses ersten Android Handys durch die noch folgenden androidbasierten Mobilgeräten der Vergangenheit angehören.

2.1.3 Internetbrowser

Der Internetbrowser, der beim Android Betriebssystem mitgeliefert wird, basiert, so wie der des iPhones, auf der WebKit Browserengine, auf die in einem späteren Kapitel eingegangen wird. Der Browser kann über den Touchscreen bedient werden, unterstützt aber, wohl aufgrund von anfänglicher Angst vor Patentklagen seitens Apple [Ven], kein Multitouch. Bei dem T-Mobile G1 ist dieser Browser weitestgehend unverändert, wohingegen bei einem im Oktober 2009 erscheinenden Handy, dem HTC Hero, eine echte Flash-Unterstützung sowie Multitouch implementiert wurde [HTCb]. Es gibt auch alternative Browser wie den Browser „Steel“, der bestimmte Fingergesten unterstützt, und den Opera Mini. Desweiteren hat Opera angekündigt, in naher Zukunft eine überarbeitete Version ihres mobilen Browsers exklusiv für die Android Plattform herauszubringen.

2.2 iPhone

Das iPhone von Apple ist ein vollwertiges mobiles Endgerät. Neben den Funktionen eines modernen Handys, wie das Telefonieren, das Versenden von SMS und einer eingebauten Kamera [App09e], kann der Benutzer in der Standardausstattung auf den iCal Kalender, das Adressbuch, den iTunes Musik-Player, einen Mailclient und den Safari Internetbrowser des Apple MacBooks zurückgreifen [App09d]. Das zentrale Konzept des iPhones zeigt sich in dieser Funktionalität: Eine neue mobile Endgeräteklasse, die unsere erweiterten täglichen Bedürfnisse nach Entstehung des Web 2.0 an die sich geänderte Realität anpasst.

2.2.1 Betriebssystem

Auf dem iPhone läuft das von Apple entwickelte iPhone OS [App09z]. Es ist aus dem „Mac OS X“ Betriebssystem hervorgegangen und sehr stark daran angelehnt. Nicht zuletzt wird das anhand der identischen Benennung seiner Programme, den „Applikationen“, deutlich. Als Betriebssystem fungiert das iPhone OS als Schnittstelle zwischen der Hardware des Geräts und der Software, etwa den Applikationen. Während Teile des Systems, wie etwa das zugrundeliegende „Darwin“ Betriebssystem, open source sind [App09a], bleibt der Quellcode für das komplette System, wie auch bei Mac OS X, unveröffentlicht.

Das einzige vom Betriebssystem unterstützte Eingabegerät ist das sog. „Multi-Touch Display“ [App09e], ein berührungssensitives Display, das bis zu 12 Berührungen gleichzeitig auswerten kann [App09-27]. Bei Bedarf kann so auch ein Softwarekeyboard zur Texteingabe gerendert werden, das der Benutzer mit Tippgesten bedienen kann. Im Zentrum des User Interfaces steht der sog. „Home Screen“ [App09b]. Auf ihm werden die Icons von allen installierten Applikationen angezeigt, auf die durch Fingertippen zugegriffen werden kann. Da das iPhone OS für Endgeräte mit kleinem Display konzipiert ist, kann nur ein Programm gleichzeitig angezeigt werden. Um zwischen Programmen zu wechseln sind also entweder Links zwischen den Programmen oder der Umweg über den Home Screen notwendig.

2.2.2 Geräte

iPhone OS läuft auf zwei unterschiedlichen Geräteklassen: Dem iPhone und dem iPod touch. Die Endgeräte dieser Klassen unterscheiden sich grundlegend in ihrer Hardware. Während in jedem iPod touch Wi-Fi und Kopfhörerausgang eingebaut sind, sind in den iPhones zusätzlich Bluetooth, GSM EDGE, Lautsprecher, Lautstärkeregulierer und eine Kamera verbaut. Nur mit den iPhones kann also telefoniert werden. Aktuell gehören fünf Endgeräte diesen beiden Geräteklassen an, deren jeweilige Ausstattung in Abbildung 1 aufgeführt ist.

Momentan sind zwei unterschiedliche iPod touch Geräte erhältlich: iPod touch First Generation [App07] und iPod touch Second Generation [App09-29]. Beide haben ein 3.5 Zoll LCD Glas-Display mit einer Auflösung von 480x320 Pixeln, eine PowerVR MBX Lite 3D Grafikkarte, 128 MB DRAM und sind mit 8, 16 oder 32 GB Speicher erhältlich. Gemeinsam ist ihnen auch der 620 MHz Prozessor, der allerdings in der ersten Generation auf 412 MHz, in der zweiten auf 533 MHz runtergeregelte ist. Die zweite Generation erweitert die erste außerdem um eingebaute Lautsprecher, Lautstärkereglern an der Außenseite und Bluetooth. Die Gehäuse sind aus Aluminium mit einer Glasscheibe über dem Multi-Touch Display.

In der Gruppe der iPhones gibt es momentan drei Endgeräte. Das „iPhone“ [App08], das „iPhone 3G“ [App09c] (3rd Generation) und das „iPhone 3GS“ [App09e] (3rd Generation Speed). Wie auch die iPod touch Geräte haben alle iPhones ein 3.5 Zoll LCD Glas-Display mit einer Auflösung von 480x320 Pixeln. Die ersten beiden Geräte haben außerdem eine PowerVR MBX Lite 3D Grafikkarte und 128 MB DRAM, wie auch die

	Original	3G	3GS	Touch	Touch 2G
Speaker & physical volume control	Yes	Yes	Yes	No	Yes
Flush-mounted headphone jack	No	Yes	Yes	Yes	Yes
Remote volume control using the iPod earbuds with remote and mic ^[17]	No	No	Yes	No	Yes
Wi-Fi	Yes	Yes	Yes	Yes	Yes
Bluetooth	Yes	Yes	Yes	No	Yes
EDGE network	Yes	Yes	Yes	No	No
3G network	No	Yes	Yes	No	No
Assisted-GPS	No	Yes	Yes	No	No
Digital compass	No	No	Yes	No	No
2.0 mpx still camera	Yes	Yes	No	No	No
3.0 mpx still and video camera	No	No	Yes	No	No
Safari, email, multimedia, maps	Yes	Yes	Yes	Yes	Yes
Voice Control	No	No	Yes	No	No
Nike+iPod	No	No	Yes	No	Yes

Abbildung 1: Ausstattung der iPhone und iPod touch Geräte [Wik09d]

iPod touch, einen 620 MHz Prozessor, runtergeregelt auf 412 MHz, und eine 2 Megapixel Kamera. Das iPhone 3GS hat eine PowerVR SGX Grafikkarte, 256 MB DRAM, einen 833 MHz Prozessor, runtergeregelt auf 600 MHz und eine 3 Megapixel Videokamera.

Die Hardware-Grundausstattung jedes iPhones umfasst Wi-Fi, eine USB 2.0 Schnittstelle, GSM/GPRS/EDGE Mobilfunknetze und Bluetooth 2.0. Das iPhone 3G erhielt zusätzlich mit „Assisted GPS“ Zugang zum GPS-System und kann sich auch mit UMTS / HSDPA verbinden. Das iPhone 3GS erweitert die Hardware um besseres HSDPA, Voice Control, ein Magnetometer und Bluetooth 2.1. Während das Gehäuse des iPhones noch zum Teil aus Aluminium besteht, wie die der iPod touch, sind die Nachfolgemodelle komplett aus Plastik, mit Ausnahme der Glasscheibe über dem Display. Jedes iPhone erschien mit einer neuen Version des Betriebssystems. Während das iPhone und der iPod touch First Generation noch auf iPhone OS 1.x liefen, erschienen das iPhone 3G und der iPod touch Second Generation mit iPhone OS 2.x, bzw. das iPhone 3GS mit iPhone OS 3.x [Wik09c]. Während ein Betriebssystem-Update für die iPhones kostenlos ist, müssen iPod touch Besitzer dafür bezahlen.

2.2.3 Internetbrowser

Der Standardbrowser im iPhone OS ist der Apple Safari [App09-28]. Im Gegensatz zu vielen anderen mobilen Endgeräten ist dieser nur geringfügig in seiner Funktionalität

beschnitten. So können zum Beispiel keine Drittanbieter-Plugins installiert werden, wie Flash oder Java [App09g]. Ansonsten rendert Safari Internetseiten aber genauso wie auf dem MacBook über WebKit [App09v], allerdings in maximal acht Tabs. Es ist außerdem möglich, alternative Browser teilweise kostenfrei über den App Store zu installieren. Beispiele sind der kostenfreie „Mango Browser“ [App09-30] und der kostenpflichtige „VanillaSurf“ [App09-31]. Bekanntere mobile Browser wie den Opera Mini gibt es allerdings zur Zeit für das iPhone nicht.

2.3 WebKit Engine

Wie in den Kapiteln von Android und iPhone schon erwähnt wurde, handelt es sich bei Webkit um eine freie HTML-Rendering Bibliothek, auf deren Grundlagen viele moderne Browser wie z.B Safari (Apple) oder Chrome (Google) aufbauen. WebKit wurde von Apple entwickelt und Teils unter GPL [Fou] und Teils unter BSD [BSD] Lizenz gestellt. Grundsätzlich besteht WebKit aus 2 Komponenten, einmal WebCore für die Darstellung von HTML und CSS und einmal JavaScript Core, der Parser für JavaScript. Es ist nicht zwingend notwendig beide Komponenten zu nutzen. Google hat für Chrome nur den WebCore genommen und für JavaScript eine eigene Engine namens V8 [Chrb]. Android und iPhone dagegen verwenden beide Webkit sowohl für die Darstellung als auch für JavaScript Verarbeitung.

Ein Auszug der wichtigsten Features von WebKit [Webb]:

- CSS 2.1 und CSS3 Support für Darstellung und Formatierung
- Möglichkeit zum Stylen von Formularelementen
- DOM Inspektor
- Drosera, ein JavaScript Debugger
- Enhanced Rich Text Editing
- verschiedene XML Technologien wie XPath, SVG oder XSLTProzessor
- verschiedene Plugins wie MPlayer, Adobe Flash Player etc.

3 Web-Standards

Um einen Einblick in vorhandene Standards und Empfehlung im Bereich mobile Webapplikationen zu bekommen stellen wir drei relevante Standards vor: Als erstes werden wir auf die vom World Wide Web Consortium erstellten Empfehlungen, die W3C Best Practices eingehen. Danach werden wir die Domainendung „.mobi“ untersuchen und die Apple iPhone Recommendations zusammenfassen.

3.1 W3C Best Practices

Das W3C oder auch World Wide Web Consortium [W3Ca] ist für die Entwicklung und Standardisierung von Webtechniken verantwortlich. Es besteht aus 349 Mitgliedern, darunter u.a Microsoft, IBM und Red Hat [W3Cd]. Dabei ist das W3C keine international anerkannte Organisation und ist eigentlich auch nicht berechtigt, Standards zu setzen, dennoch haben es viele dieser Standards zu eine ISO Norm geschafft, wie z.B XML unter dem Namen ISO 8879:1986 Standard Generalized Markup Language (SGML) [W3Ce]. Der Gründer der W3C ist Tim Berners-Lee, der für die Erfindung des World Wide Webs (WWW) bekannt ist. Das WWW macht es erst möglich, Dokumente auf Entfernung abzurufen, wofür man im Endeffekt Standards braucht. Einige dieser standardisierten Webtechniken sind (X)HTML, XML und CSS, die heute nicht mehr wegzudenken sind. Bei der Entwicklung dieser Standards wurde versichert, dass die Techniken frei von gebührenpflichtigen Patenten bleiben [W3Cb]. Um ihren Standards keinen offiziellen Charakter zu geben, nennt das W3C diese „W3C-Recommendations“ und „Best Practices“.

Bis es zu einer W3C-Recommendation kommt durchläuft muss es vorher die Stadien Working Draft, Last Call Working Draft, Candidate Recommendation und Proposed Recommendation durchlaufen. Erst die finale Version hat wirklich Relevanz für die Websiteentwickler, während die vorherigen Stadien nur wichtig für die Entwickler der auf die Standards aufsetzende Software sind.

3.1.1 Mobile Web Best Practices

Wir begrenzen uns im Rahmen unseres Projektes nur auf einige dieser Standards und Best Practices, nämlich XHTML für die semantische Ausgabe der Websites, CSS für das Design und JavaScript für die clientseitige Verarbeitung der vom Server übergebenen Daten. Dabei beziehen wir uns auf die am 29. Juli 2008 offiziell herausgegebenen Mobile Web Best Practices 1.0, Basic Guidelines [W3Cc], welche im finalem Stadium vorliegt.

Hier stellen wir eine Zusammenfassung mit den für uns relevanten Empfehlungen vor. Primär geht es in dem Dokument um die Unterschiede zwischen dem Desktop Browser und mobilen Browsern. Das Dokument beginnt mit der Aussage, dass die meisten Websites auf Desktop Software und Bildschirmgröße ausgerichtet sind. Da normale Websites auf den mobilen Geräten aufgrund der zu kleinen Bildschirmgröße und eingeschränkten Eingabegeräten kaum bis gar nicht bedienbar sind, wäre es sinnvoll, eine komplett neue Website oder Ansicht für die mobilen Geräte zu entwickeln. Außerdem stellen die Bandbreite und der verursachte Traffic ein großes Problem dar, da einerseits die Techniken,

wie UMTS, flächendeckend verfügbar sind und andererseits hierbei die Kosten für Traffic ziemlich teuer sein können. (In Deutschland gibt es aktuell keine volumenunbegrenzte Tarife). Außerdem sind die Latenzzeiten viel höher, was dazu führen kann, dass die Navigation auf der Website zu langsam wird.

Als Zielgruppe bei mobilen Geräten definiert das W3C folgende Eigenschaften:

- Bildschirmbreite von mindestens 120 Pixel
- Browsereigenschaften wie:
 - XHTML Support
 - UTF8 Support
 - JPEG Unterstützung
 - GIF/PNG Unterstützung
- mindestens 256 Farben
- mindestens CSS 1 Unterstützung, CSS 2 Optional
- HTTP 1.0 Support
- Script (z.B JavaScript) Support im Clienten nicht unbedingt notwendig

3.1.2 W3C Validator

Desweiteren bietet das W3C einen Validator [W3Cf] für das Überprüfen der Website an. Es gibt sowohl eine mobile (MobileOK Checker) als auch eine Desktop (Markup Validation Service) Version. Für unser Projekt ist allerdings nur die mobile Version relevant. MobileOK Checker überprüft die Website auf die mobile Tauglichkeit sowie auf die in der Recommendations empfohlenen Best Practices. Eine Beispielsuntersuchung an der Website von TV Movie findet man in der Abbildung 2

Die Abbildung zeigt einerseits die Syntax Fehler, die auf der Seite gefunden werden und andererseits Performance Verbesserungs Tipps oder auch nur Warnungen bei Browserunterschieden.

3.1.3 Kritik

Trotz der einheitlichen Standards steht das W3C immer wieder unter Kritik. Eines der Kritikpunkte ist, dass dem W3C vorgeworfen wird, von den großen Unternehmen wie z.B Apple oder Microsoft dominiert zu werden, wie z.B in dem HTML5 Audio und Video Codec Streit zugunsten der großen Unternehmen entschieden wurde und nicht zugunsten der Community [Hic]. Ursprünglich sollte es in HTML5 ein Audio und Video Codec geben, d.h jeder Browser kann ohne Zusatzprogramme Audio und Video abspielen, was aber zu dem Konflikt geführt hat, dass die Mitglieder wie Adobe, Apple dagegen geklagt haben, da sie ihre patentierten Codecs durchsetzen wollten. Letzendlich wurden die Codecs aus der Spezifikation entfernt. Mozilla und Google ignorieren dies aber und

W3C W3C mobileOK Checker
Is your Web site mobile-friendly?

Address:

Result: 67/100

Page Size: 17.9KB document: 9.3KB - images: 8.6KB

Network: 12 requests document: 1 - images: 11

Type	URI	Redirects
image	http://cell-tvt.122.207.net/b/ss/cell-tvt/5.1/4686190?pf..._lpp&h1=pageName&D=...	1
image	http://mobtvtlo.iwbox.de/cq4bin/iw/CP/op1001.tvt?r=&d=6527872	1
document	Resource under test	0
image	http://m.tvtoday.de/op/tvt/img/2/Poq5eTZXGeM285q4OT8sq/icon_tipp.gif	0
image	http://m.tvtoday.de/op/tvt/img/B/KMnqlluAM_mSeE7CKyPcw/TVT2.png	0
image	http://m.tvtoday.de/op/tvt/img/K/PjcnzKXG.TpBO5qRm9Wv7A/143531.jpg/80.-.jpg	0
image	http://m.tvtoday.de/op/tvt/img/S/b-lopEtmnS3InqKHyUQ/change.png/10.-.png	0
image	http://m.tvtoday.de/op/tvt/img/X/7AczdN5B0hukCC6XfRQ/icon_film.gif	0
image	http://m.tvtoday.de/op/tvt/img/./kt78vUUh6TIN9m-SC-kkq/tv.png/10.-.png	0
image	http://m.tvtoday.de/op/tvt/img/v/ls9LAaeR1HZXQ08vijRw/1036872357.jpg/80.-.jpg	0

Abbildung 2: W3C Validator angewand auf die mobile Version von TV Movie [W3Cf]

setzen trotzdem auf diese Spezifikation (Firefox [Moz] und Chrome [Chra] enthalten in der aktuellen Version diese Codecs).

Ein weiterer Kritikpunkt ist das Nichteinhalten der Standards einiger Mitglieder wie z.B. Microsoft. Der Internet Explorer unterstützt nur einen Teil [ACI] der veröffentlichten Standards der W3C und nutzen somit ihre Monopolstellung aus.

3.2 DotMobi Top-Level-Domain

mTLD, Mobile Top Level Domain, meist mit dotMobi abgekürzt, ist ein Unternehmen, das seit Ende 2006 die Top-Level-Domain .mobi anbietet, seitdem sie Mitte 2005 die Rechte an dieser Endung von der „Internet Corporation for Assigned Names and Numbers“ (ICANN) erhalten haben. Die ICANN verwaltet als privatrechtliche Non-Profit-Organisation die Top-Level-Domains im Internet. Die Nutzung von Domains mit der Endung .mobi ist von dotMobi mit einigen Anforderungen an die technische Gestaltung der darauf verweisenden Webseiten verbunden, um für den Endnutzer zu gewährleisten, dass jede Webseite mit der Endung .mobi für mobilen Endgeräten passend dargestellt wird. Die Firma dotMobi, die durch einige namenhafte Sponsoren, wie Microsoft oder Google, unterstützt wird, hat einige Hilfsmittel, wie eine „dotMobi Mobile Web Developers Guide“ und diverse Webtools herausgebracht, um bei der Entwicklung von mobilen Webseiten richtungsweisend zu wirken. Im Folgenden werden die wesentlichen Eigenschaften der Nutzung einer dotMobi Domain aufgezeigt und die Kritik an einer solchen, für mobile Geräte beschränkten Domain, herausgearbeitet. Am Schluss werden

ein paar Webtools von dotMobi vorgestellt, die als Hilfestellung für die Entwicklung von mobilen Webapplikationen dienen können.

3.2.1 dotMobi Domain

Laut der Firma dotMobi kommen auf einen gekauften PC vier gekaufte Handys und somit auch ein potenziell großer Markt an Webseiten, die speziell für diese mobilen Geräte optimiert sein müssen [dot07]. Nach deren Verständnis muss eine mobile Webseite navigationstechnisch anders aufgebaut sein, als eine herkömmliche Webseite. Somit muss die Webseite sowieso komplett neuen Anforderungen gerecht werden und sollte dann eine eigene Webpräsenz unter der Endung .mobi sein, um den Endnutzern damit eindeutig aufzuzeigen, dass die Webseite für mobile Geräte optimiert ist. Die Webseiten müssen den Anforderungen von dotMobi genügen und somit hinsichtlich der Auflösung, der Eingabemethodik und anderen Kriterien, die in einem Paper [dot07] auf deren Seite zusammengefasst wurden, angepasst sein. Kritiker sagen nun, dass diese Anforderungen auch ohne eine eigene Domainendung umgesetzt werden können, indem zum Beispiel aufgrund der Auflösung oder des User-Agents, der beim Aufrufen einer Webseite mitgesendet wird, die Webseite mit unterschiedlichen Stylesheets ausgestattet wird und somit für die jeweiligen Geräte optimiert ist. Das Vorgehen bei einer separaten Webseite unter einer .mobi-Domain würde dem ursprünglichen Designkonzept von Webseiten widersprechen, da somit zwei unterschiedliche Inhalte verwaltet werden müssten.

3.2.2 Webtools

Unter der Kategorie mobiThinking [moba] bietet dotMobi Artikel zum Thema Best Practices und Showcases, die aber leider ein wenig veraltet sind und meist nicht auf die neueren Formfaktoren des iPhone und ähnlichen Geräten angepasst sind. Unter mobiForge hingegen findet man aktuelle Themen und wirklich hilfreiche Artikel zum Design von Webseiten für mobile Endgeräte im Blog unter dem Punkt „Designing“ aktuelle Artikel zum Thema Android.

Interessant wird es unter mobiReady [mobb]. Hier gibt es einen Page, Markup und Site Test, ähnlich den Validatoren des W3C, die hilfreich sind, wenn es um das Testen von Webseiten in Bezug auf deren Anzeige und Funktionsalität auf mobilen Endgeräten geht. So können Webseiten für mobile Geräte optimiert werden ohne ein Gerät zu besitzen. Wenn dort ein Account erstellt wird, können mehrere Seiten kostenlos zum Testen in eine Warteschlange gegeben werden, damit diese durchgetestet werden. Die Kriterien in dieser Evaluierung stellen die Übertragungsgröße der Webseite, die Auslassung von Frames und die Validierung nach einem XHTML Profil dar. Hiernach wird eine Note von 1 bis 5 vergeben, wobei 5 die Beste ist. Auf der Auswertungsseite der unterschiedlichen bewerteten Unterseiten lassen sich genauere Ergebnisse einsehen. Interessant hierbei ist unter anderem der zu erwartende durchschnittliche Preis für den Endnutzer pro Land bei einem Zeittarif und die erwartete Zeit der Übertragung in Sekunden bei Wi-Fi, UMTS und GPRS (siehe Abbildung 3). Hier lässt sich die Seite auch in einem Emulator der Handys Nokia N70, Samsung z105, Sony Ericsson k750i, Motorola v3i und Sharp GX-10

darstellen, die aber nicht die aktuellsten Entwicklungen der unterschiedlichen Hersteller darstellen.



Abbildung 3: Ergebnis von google.mobi durch mobiReady [mob]]

Eine weitere Hilfestellung beim Entwickeln einer angepassten Webseite bietet dotMobi durch ihren DeviceAtlas [Dev]. Sie bieten hiermit eine Lösung, die Unterschiede der mobilen Endgeräten in dem wirklich großen Angebot herauszufiltern und die Webseite aufgrund dieser Werte für die unterschiedlichen Geräte passend darzustellen. Dies wird durch eine API für Java, .NET, PHP, Python and Ruby möglich. Der Einsatz dieser aktuellen Datenbank für Mobilgeräte ist im Normalfall kostenlos wobei weitere Dienstleistungen, wie zum Beispiel Enterprise Support oder die Verbindung mit eigenen Daten, mit einer Lizenz dazu erworben werden können.

Ein letztes Tool stellt der Instant Mobilizer [Ins] dar, der aus einer bereits bestehenden Seite eine für mobile Geräte angepasste generiert, indem er Bilder verkleinert, Text anders formatiert und kleinere Anpassungen am Design vornimmt. Dass dies nur bedingt funktionieren kann, zeigt aber leider der Test mit ein paar bestehenden Seiten. Am Ende der „Sofortmobilisierung“ wird einem dann auch gleich der Kauf einer .mobi-Domain bei einem Partner vorgeschlagen, womit klar wird, dass der Mobilizer nur ein Kundenfänger ist. Im Endeffekt wird dann bei jedem Aufruf der dann gekauften .mobi Domain, die bestehende nicht-mobile Webseite durch den Instant Mobilizer für mobile Geräte angepasst und das Ergebnis angezeigt.

3.3 Apple iPhone Recommendations

Mit der Subdomain „http://developer.apple.com“ stellt Apple eine zentrale Anlaufstelle für Entwickler bereit, die auf Apple-Software basierende Programme schreiben wollen. Die Benutzer können Möglichkeiten zum Gedankenaustausch, Tutorials und offizielle Apple SDKs nutzen. Aus dieser Quelle stammen auch die *iPhone Human Interface Guidelines for Web Applications* [App09f]. Im Folgenden werden diese in einem für das Teamprojekt sinnvollen Rahmen ausgewertet.

3.3.1 Safaris funktionale Einschränkungen auf dem iPhone

Der Standard-Browser des iPhones ist eine Portierung des Safaris auf das iPhone OS. [App09y] Während die internen Funktionen des Safaris nicht beschnitten wurden, gibt das mobile Betriebssystem dem Benutzer allerdings Grenzen vor. Da auch das iPhone selbst Java nicht unterstützt, können auch im Browser keine Java-Applikationen ausgeführt werden. Das gleiche gilt für Flash und Plug-Ins von Drittanbietern. [App09g] Weitere Einschränkungen legt die Fingersteuerung des Touchscreens auf, die sich grundlegend von der Steuerung mit einem Mouse-Zeiger unterscheidet. Die Fingergesten dienen primär der Verschiebung des sichtbaren Bildschirmbereichs und können so nicht, wie ein Mouse-Zeiger für hover-Effekte verwendet werden. [App09h]

3.3.2 Fokussierte Mobilität und Benutzerwahrnehmung

Ein Benutzer, der eine Web-Applikation mit einem iPhone aufruft, befindet sich häufig in einer Situation, in der ihm kein anderes internetfähiges Gerät zur Verfügung steht. Im Gegensatz zu Standrechnern und sogar Laptops, ist er mit einem iPhone wirklich mobil. Häufig ist er gerade irgendwohin unterwegs und hat wenig Batterie, Zeit und Lust, sich mit komplexen Layouts oder unnötigen Inhalten zu beschäftigen. Für ihn zählt einzig schneller und einfacher Zugang zu den gewünschten Daten, ohne überzogenes Branding. Eine Web-Applikation für das iPhone sollte sich deshalb einer konkreten Hauptaufgabe widmen, die benutzerfreundlich und übersichtlich ausgelegt wird, wobei untergeordnete Aufgaben entweder in eine weitere Web-Applikation ausgelagert werden, oder komplett weggelassen werden sollten. [App09j]

Das iPhone vereinigt viele Features ehemals eigenständiger Geräte in einem Endgerät. Es dient sowohl als MP3-Player, Kalender, leichtgewichtiger Email-Client mit Internetzugang und Handy. Alle diese Applikationen haben zum Ziel, die Mobilität des Benutzer bei alltäglichen Aufgaben zu steigern. Eine Web-Applikation sollte dasselbe Ziel verfolgen. Dazu sollte sie sich möglichst nahtlos in die vorhandenen Applikationen einfügen und versuchen, diese nachzuahmen. Ein wichtiger Faktor ist die Verwendung von bekannten, den vorhandenen Applikationen nachempfundenen Steuerelementen, was dem Benutzer die Eingewöhnungszeit erspart. Sinnvoll ist außerdem die Loslösung von der Browsernavigation, um den Eindruck einer eigenständigen Applikation zu verstärken. Eine subtile Farbverwendung und diskretes Branding unterstützen und fördern ein aufgeräumtes und eingängiges Layout. Zooming in der Webapplikation sollte deaktiviert und nötiges Scrollen auf ein Minimum und die vertikale Achse begrenzt werden. [App09i]

Das iPhone ermöglicht dem mobilen Benutzer die verzahnte und parallele Ausführung vieler Applikationen. So wird er nicht erst die Musik ausstellen, wenn er eine E-mailbenachrichtigung bekommt und so möchte er auch nicht erst die Web-Applikation schließen. Schnelles und spontanes Wechseln zwischen den Applikationen ist normal und auch hier sollte eine Webapplikation gleichziehen. Ruft ein Benutzer nach einer längeren Zeitphase die Web-Applikation wieder auf, möchte er an genau der Stelle fortfahren zu arbeiten, an der er unterbrochen wurde. Das muss ihm entweder serverseitig durch regelmäßige Logs oder auf Seite des Clients durch geeignete Cookies ermöglicht werden. Ferner muss er auch in der Web-Applikation das Gefühl haben, mit den nativen Applikationen des iPhones vernetzt zu sein. Das kann durch Schnittstellen mit diesen Applikationen, wie speziellen Links, aber auch durch geeignete Datenstrukturen, wie eine mögliche Kalendereinbindung von Terminen, geschehen. [App09p]

3.3.3 Vermittelte Leichtigkeit

Die Hauptaufgabe des iPhone-Programmierers ist es, dem Benutzer die seiner Mobilität angemessene Leichtigkeit im Umgang mit der Applikation zu vermitteln. Dazu sollte er die Überreizung des Benutzers durch zu viele dargestellte Farben und Formen auf dem kleinen Display, wie etwa durch überzogene dekorative Elemente, vermeiden. [App09i] Vor allem sollten Bilder, Links und Adds sehr sparsam eingesetzt werden. [App09k] Fundierte und knappe Bezeichner oder eingängige Symbole ermöglichen dem Benutzer ferner eine intuitive Bedienung der Web-Applikation, was ihm Zeit und Ärger erspart. [App09n] Unnötige Interaktivität, wie vom Inhalt losgelöste Startpages, werden von ihm allerdings als störend wahrgenommen. [App09o]

Um dem User Menus weitestgehend zu ersparen, sollten sich die wichtigsten Informationen direkt auf der Hauptseite befinden. Diese sollten möglichst ohne unnötiges Scrollen und ohne überflüssige Informationen abrufbar sein. [App09l] Eine Möglichkeit, längere Texte zu kürzen, sind Zusammenfassungen, denen ein Link zum Volltext folgt. Auch die Anzahl der nötigen Bedienelemente sollte der Übersichtlichkeit halber minimiert werden. Ein trotzdem benötigtes Hauptmenu könnte durch einen einzelnen Link oder durch eine Tabbar ersetzt werden. Vorrang hat allerdings die Verkürzung des Clickstreams des User. [App09w]

Weiterhin trägt die konsequente Vermeidung größerer Freiflächen und Abstände zwischen den einzelnen Elementen der Web-Applikation zu ihrer Übersichtlichkeit bei. Dabei muss allerdings immer auf die spezifische Steuerung des iPhones mit den Fingerkuppen geachtet und den Bedienelementen genug Platz eingeräumt werden, um bequem und schnell benutzbar zu sein. [App09k] Den vielleicht größten Bottleneck stellt allerdings der User Input dar. Die geringe Größe des iPhones und das Fehlen einer Hardware-Tastatur macht es dem Benutzer schwer, auch unterwegs Texteingaben zu tätigen. Diese Schwierigkeiten sollten durch alternative Eingabemöglichkeiten, wie Selection-Lists oder Pop-Up Menus abgefangen werden, um aufkommendem Frust vorzubeugen. [App09m]

3.3.4 Konkrete Tipps zur Umsetzung

Um spezifischen CSS Code für das iPhone zu verwenden, kann die im CSS 3 Standard enthaltene „media query“ verwendet werden. [App09r] Da das iPhone einen, bis auf die oben genannten Ausnahmen voll funktionsfähigen, Safari Browser enthält, kann nicht anhand des Browser zwischen iPhone und Desktop Rechner oder Laptop differenziert werden. Allerdings kann hier die spezifische Größe des iPhone Displays herangezogen werden, um den Client als ein solches zu identifizieren. [App09s] Ferner wird die Standardbreite einer Website vom Safari auf 980px gesetzt oder einer festen „width“-Angabe im CSS Code angepasst. Da die Standardgröße für das iPhone zu groß ist, ist es also sinnvoll, die Breite der Seite selbstständig durch CSS festzulegen. [App09t]

Am einfachsten zu erfassen und zu bedienen ist das sogenannte Listenlayout, in dem der Inhalt einer Web-Applikation linksbündig in Form von Listen dargestellt wird. Dieses Layout wird auch von einem Großteil der nativen iPhone Applikationen verwendet und ist dem User schon bekannt. [App09u] Eine für Bildschirmausgabe optimierte Schriftart, wie beispielsweise Helvetica, und eine Schriftgröße zwischen 17-22 Pixeln, ermöglichen dem Benutzer die augenschonende Betrachtung der Web-Applikation. Betonungen und Hervorhebungen sollten in der gleichen Schriftart als fetter Text realisiert werden, was ebenfalls die Augen schont. Kursivschriften und Unterstreichungen von Links sollten vermieden werden, weil erstere zu unscheinbar sind und letztere den Text zusammengedrängt erscheinen lassen. [App09x] Ein nettes Feature ist das Bereitstellen eines speziell angepassten Icons, das per Web Clip auf dem Home Screen eingebunden werden kann. [App09q]

4 Evaluation

Aus den vorangegangenen Untersuchungen schon vorhandener Empfehlungen und Standards zeigen wir nun im Hinblick auf die aktuelle Technik eine Herangehensweise und Umsetzung der Erkenntnisse. Die Standards werden ausgewertet und hingehend ihrer Praxistauglichkeit beurteilt um dann daraus eine Umsetzungsstrategie für unser Projekt zu entwickeln.

4.1 Beurteilung der Standards

Die untersuchten Standards bieten eine gute Basis für mobile Webapplikationen. Wir beziehen uns dabei hauptsächlich auf die Recommendations von W3C, da wir uns dafür entschieden haben, dass wir eine Applikation für alle mobile Geräte anbieten wollen. Die iPhone Recommendations spielen hierbei aufgrund der Plattformunabhängigkeit eine untergeordnete Rolle, da wir davon nur die Empfehlungen davon betrachten, die keinen Einfluss auf die anderen Geräte und auf die W3C Spezifikation nehmen. Außerdem verzichten wir bei unserem Prototypen auf die dotMobi Domain, da diese jederzeit als Erweiterung in Betracht kommen kann und bei der Entwicklung der Applikation keine Relevanz hat.

Die Mobile Web Best Practices 1.0 von W3C bieten eine gute Zusammenfassung aller mobiler Standards und Best Practices.

4.2 Umsetzung

Bei der Umsetzung ist auf jeden Fall darauf zu achten, dass die mobile Webapplikation separat läuft, d.h. eine komplett eigene Ansicht für die mobile Version der Seite um weniger Daten zu übertragen. Dadurch kann man sich komplett auf die mobilen Geräte konzentrieren und vermeidet damit viele Fehler beim testen. Außerdem soll es nur eine einzige Applikation für alle mobile Geräte geben um Redundanz zu vermeiden. Für das Design der Website verwenden wir XHTML und CSS. Für die Navigation auf der Campuskarte setzen wir auf JavaScript, obwohl von W3C empfohlen wird darauf soweit wie möglich zu verzichten. Dies ist in unserem Fall allerdings notwendig, da sonst keine benutzerfreundliche Navigation auf der Campuskarte bzw. der dazugehörigen Wiki möglich ist. Die Inhalte der Campuskarte werden via AJAX nachgeladen und mit JavaScript verarbeitet und in der Website eingefügt. Mit Ausnahme der Campuskarte versuchen wir aber auf clienseitige Verarbeitung der Website aus Performancegründen zu verzichten. Um die Arbeit zu vereinfachen setzen wir auf auf JavaScript und User Interface (Ui) Toolkit, worauf im nächstem Kapitel näher eingegangen wird.

5 UI Toolkits

Nachdem wir uns ausführlich über vorhandene Toolkits informiert haben, die als grafische Basis für unsere mobile Webapplikationen in Frage kämen, gab es im wesentlichen drei Optionen. Zwei basierend auf vorhandenen UI Toolkits, die primär für das iPhone gedacht waren, und die dritte ist, ein eigenes Design zu entwickeln. Andere Toolkits disqualifizierten sich schon von vornherein, da sie entweder veraltet oder nicht ausreichend gut dokumentiert waren. In diesem Abschnitt werden wir nun auf die verschiedenen Aspekte der Toolkits, insbesondere deren Vor- und Nachteile, eingehen und schließlich eine Auswahl treffen.

5.1 iWebkit

iWebkit wirbt unter anderem mit dem Slogan „iWebkit is a great tool because it is very easy to use, extremely fast, compatible and extendable.“ [iWea]. Ob es diesem Slogan gerecht wird wird in den nächsten Kapiteln evaluiert.

5.1.1 Überblick

iWebkit ist beim Schreiben dieses Grobkonzeptes in der Version 4.6.2 zum Download¹ freigegeben. Es baut größtenteils auf XHTML strict, CSS und JavaScript auf und versucht Ajax zu vermeiden, um den Code und die Handhabung so einfach wie möglich zu gestalten. Es wird hauptsächlich von Christopher Plieger und Johan van Wilsum entwickelt, die in regelmäßigen Abständen neue Versionen veröffentlichen.

5.1.2 Aussehen

Die Abbildung 4 zeigt eine mögliche Übersichtsseite, neutral in schwarz gehalten und im Android-Browser dargestellt. Auf der Abbildung 5 ist eine eventuelle Ansicht eines Wiki-Artikels zu sehen. So oder ähnlich kann eine mobile Applikation mit dem iWebkit aussehen, wenn keine weiteren Modifikationen vorgenommen werden und das Toolkit in seinem gegebenen Umfang genutzt wird.

5.1.3 Lizenz

Bei der Lizenz handelt es sich um die GNU LGPL v3, die GNU Lesser General Public License. Diese besagt kurz zusammengefasst, dass die Software, in diesem Fall iWebkit, für jeden beliebigen Zweck genutzt, vervielfältigt und beliebig verändert werden darf. Die Bedingung hierbei ist, dass die veränderte Versionen wieder unter der GNU LGPL v3 freigegeben wird.

Im Gegensatz zur GPL v3, darf bei der LGPL iWebkit aber auch als Bibliothek in einem proprietären Programm verwendet werden, ohne das gesamte Programm dann unter eine kompatible Lizenz stellen zu müssen. Die GNU LGPL ist somit die perfekte Lizenz

¹<http://iwebkit.net/downloads>

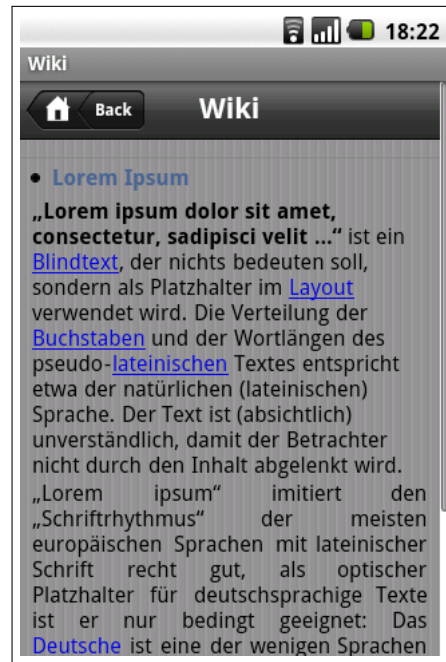
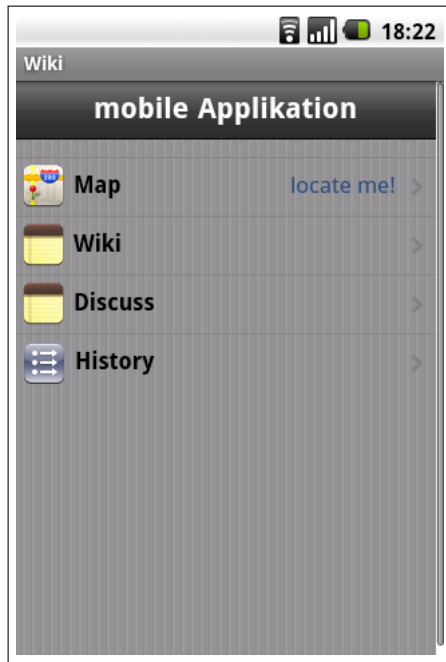


Abbildung 4: Übersichtsseite, iWebkit (Eigenentwicklung)

Abbildung 5: Wiki-Artikel, iWebkit (Eigenentwicklung)

für eine solche Bibliothek. Es wird damit sichergestellt, dass sie aktiv weiterentwickelt wird, aber auch in kommerziellen Webseiten eingesetzt werden kann.

5.1.4 Bewertung

Das Toolkit kann auf einer modernen übersichtlichen Webseite heruntergeladen werden, die auch ein Forum, aktuelle News rund um iWebkit und weitere Downloads, wie Plugins und Themes, zur Bibliothek bereithält. Dem Hauptentwickler kann sogar auf Twitter gefolgt werden, um an den aktuellen Entwicklungen teilzuhaben. Hier zeigt sich gleich eine der großen Stärken des Toolkits: Es ist aktuell, hat eine aktive Community und es gibt Plugins für 4 bekannte Frameworks/WebCMS namentlich Grails, Drupal, Wordpress und iForum. Das Downloadpaket beinhaltet eine Anleitung auf Englisch, das Toolkit selbst und eine Demonstrationsanwendung desselbigen. Die Anleitung geht von den simpelsten HTML Grundübungen bis hin zur Anwendung des Toolkits, ohne dessen Funktionsweise genauer zu beschreiben. Da die Bibliothek aber klar strukturiert wurde, ist leicht zu erkennen, welche Stylesheets für welchen Zweck gedacht sind.

Die Bibliothek wurde klar für auf Webkit basierende Browser konzipiert und somit funktionieren die erstellten Seiten im Safari und Google Chrome Browser tadellos, wohingegen der Firefox gerade die angepassten Formularelemente nicht richtig darstellen kann. Da es bei unserer Applikation hauptsächlich um die Darstellung auf mobilen Geräten geht, ist es wichtig, dass die Webseite auf den Referenzplattformen iPhone und Android

gut dargestellt wird.

Erwartungsgemäß haben unsere Tests gezeigt, dass dies auch der Fall ist, da beide Browser auf Webkit basieren und die gleichen Auflösungen haben. Der Opera Mini, ein weiterer weit verbreiteter Browser für mobile Geräte, verhält sich leider nicht ganz optimal. Es gibt einige Darstellungsfehler und bei den Formularelementen lassen sich die Checkboxes nicht auswählen. Nachdem wir ein kleines Beispieldesign mit dem Toolkit erstellt haben, zeigten sich die Stärken von iWebkit. Es ist wirklich sehr leicht zu bedienen und bietet einem die Grundfunktionalitäten die benötigt werden um ein sauberes Design für Webkit fähige Browser zu erstellen.

Alle weiteren Funktionalitäten, wie Ajax, müssten selbst programmiert und hinzugefügt werden. Dies kann aber auch von Vorteil sein wenn spezielle Ajax Funktionalitäten auf der Webseite benötigt werden. Es kann somit nicht zu Inkompatibilitäten zwischen den JavaScript Bibliotheken des Toolkits und der eigenen kommen.

5.2 iUi

iUi ist ein weiteres Toolkit für mobile Applikationen. Es konzentriert sich hauptsächlich auf den Safari-Browser und das iPhone, verspricht aber auch auf anderen mobilen Geräten lauffähig zu sein.

5.2.1 Überblick

iUi [iUiB] ist aktuell in der Version 0.20. Das Toolkit verspricht, dass es ohne Kenntnisse von JavaScript leicht in Standard-HTML einzubinden ist. Es lässt die Applikation dann im typischen iPhone Look and Feel aussehen, sowohl auf den iPhone, als auch auf fremden Geräten. Intern nutzt es CSS, HTML und JavaScript und verzichtet fast vollständig auf Ajax-Requests. Wie schon erwähnt, muss der Benutzer selbst nichts in JavaScript schreiben, sondern nur bestimmte CSS Klassen an Elemente hängen.

5.2.2 Aussehen

In Abbildung 6 und 7 ist ein Beispiel für ein iUi Menü zu sehen. Die Style Sheets sind allerdings für Webkit optimiert, sodass die Applikation nur auf iPhone oder Android ohne Einschränkungen funktionieren würden.

Für uns ist das allerdings kein Nachteil, weil einerseits iPhone Geräte weit verbreitet sind und andererseits die Einschränkungen nicht so groß sind. Ein Vergleich zwischen Safari auf dem iPhone und Opera auf dem Android ist in in Abbildung 8 bzw. in Abbildung 9 zu sehen.

5.2.3 Lizenz

Das iUi hat eine New BSD License [BSD], welche uns erlaubt, das Toolkit sowohl in freien als auch in proprietären Projekten zu verwenden.

Auszug aus der Wikipedia (Stand 9. Juni 2009):

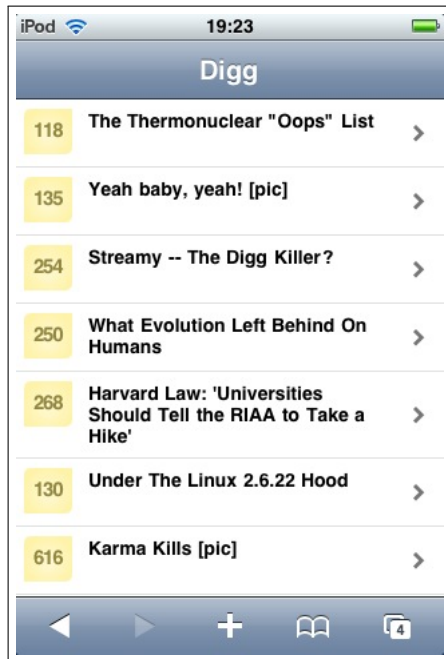


Abbildung 6: Beispiel einer iUi Seite [iUia]

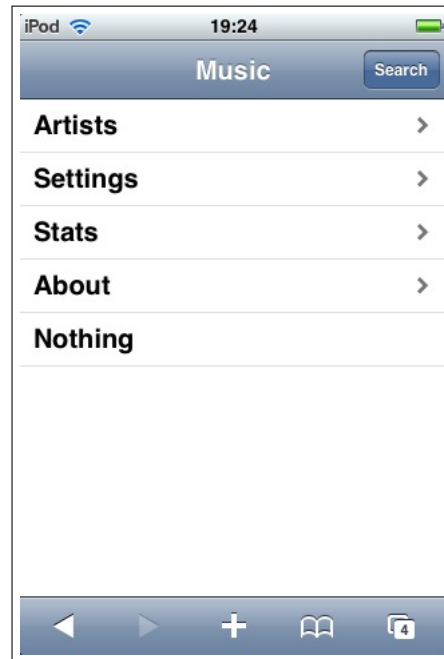


Abbildung 7: Weiteres Beispiel einer iUi Seite [iUia]

„Software unter der BSD-Lizenz darf frei verwendet werden und es ist erlaubt, sie zu kopieren, zu verändern und zu verbreiten. Einzige Bedingung der BSD-Lizenz ist, dass der Copyright-Vermerk des ursprünglichen Programms nicht entfernt werden darf. Somit eignet sich unter der BSD-Lizenz stehende Software auch gut als Vorlage für kommerzielle (teilproprietäre) Produkte.

Dieses Lizenzmodell unterscheidet sich von der GPL darin, dass es kein Copyleft enthält: Ein Programmierer, der ein unter einer BSD-Lizenz veröffentlichtes Programm verändert und dann verbreitet, ist nicht verpflichtet, den Quellcode seines veränderten Programms mitzuveröffentlichen. Er ist auch nicht verpflichtet, das Ergebnis seiner Änderungen wiederum unter der BSD-Lizenz zu veröffentlichen.“

Diese Lizenz würde sich somit für unser Teamprojekt eignen, egal ob wir es frei oder kommerziell (proprietär) vertreiben würden.

5.2.4 Bewertung

Das Toolkit bietet eine sehr gute Funktionalität für Webapplikationen auf mobilen Geräten. Auf der Projektseite können viele Beispiele und eine umfangreiche Dokumentation gefunden werden, nach der ein Webdesigner ohne Programmierkenntnisse das Layout anpassen könnte, ohne sich mit JavaScript zu beschäftigen.



Abbildung 8: Beispiel Geräte mit Web-Kit (Eigener Test)



Abbildung 9: Beispiel Geräte ohne Webkit (Eigener Test)

Zusätzlich existieren viele Programmierschnittstellen mit anderen Frameworks, z.B. Grails oder Ruby on Rails, mit einigen Beispielen dazu. Dadurch ist es nicht so einfach, Zusatzfunktionalität hinzuzufügen, ohne das Toolkit selbst umzuschreiben. Durch die BSD Lizenz ist es allerdings kein Problem das Toolkit zu erweitern, da der Code beliebig verändert werden darf, ohne die Änderungen freigeben zu müssen.

Insgesamt entsteht aber der Eindruck, dass das iUi Toolkit nicht ganz ausgereift und außerdem ziemlich überladen ist. Es ist z.B. nicht komplett Browserunabhängig und auch im Safari auf dem iPhone laufen die Seiten nicht ohne weitere Anpassungen. Außerdem setzt das Toolkit zuviel auf JavaScript Operationen, die besser vom Server übernommen werden sollten, denn in der Performance macht sich bemerkbar, dass der interpretierte JavaScript Code deutlich langsamer ist, als der serverseitig compilierte Code.

5.3 Eigenes Toolkit

Da leider keins der gefundenen Toolkits unseren Erwartungen und Anforderungen vollends entsprechen konnte, müssten bei der späteren Realisierung des Prototypen mit egal welchem Toolkit auf jeden Fall Anpassungen vorgenommen werden. Daher ist abzuwägen, ob es sinnvoller ist, ein komplexes Toolkit unseren Anforderungen anzupassen, oder ob wir besser auf einem simplen und übersichtlichen Toolkit aufsetzen und dieses um die fehlenden Funktionen erweitern sollten. Die letzte Möglichkeit soll im Folgenden näher betrachtet werden.

5.3.1 Überblick

Unser eigenes Toolkit setzt auf dem Universal iPhone UI Kit [UiU] (UiUI Kit) von Diego Martín Lafuente auf, dessen Ziel es ist, nur unter Verwendung von reinem HTML und durch WebKit erweitertes CSS, komplexe Interfaces für das iPhone zu erstellen. Er setzt dabei nicht, wie viele iPhone Apps, auf mit Ajax realisierte Übergangseffekte beim Wechsel von Seiten, sondern benutzt statische HTML-Links.

Das UI Toolkit unterstützt Kopfzeilen mit je einem optionalen Zurück- und Funktionsbutton, sowie zwei optionalen Funktionsbuttons am Seitenboden, die noch durch ein Button Panel mit vier Buttons ergänzt werden können. Es beinhaltet ein generelles Layout für Forms, mehrere für Listen und ein rudimentäres Konzept zur Einbindung von Bildern. Hervor sticht die konzeptionelle Umsetzung eines Chat Interfaces.

Damit das Toolkit unseren Anforderungen gerecht wird, müssen wir zusätzlich noch ein Konzept für Tabs am Seitenboden nach iPhone Vorbild erarbeiten und umsetzen, um mehr, aber trotzdem übersichtliche, Navigationsmöglichkeiten zu erhalten. Außerdem müssten wir die Ajax-Übergangseffekte selbst schreiben, bzw. mit einem geeigneten Javascript Framework verwirklichen.

5.3.2 Aussehen

In Abbildung 10 ist das mögliche spätere Design einer Wiki Seite zu sehen. Dabei ist die Tabbar am Seitenkopf zu Anschauungszwecken als Grafik eingefügt worden. Das übrige Layout wurde mit dem UiUI Kit erstellt.

Bei Abbildung 11 fand ein Wechsel in den Edit-Modus des Wikis statt. Die Überschrift steht hier in einem Input-Feld und der Text in einer Textarea. Die Tastatur ist nicht im Toolkit enthalten, sondern wird vom iPod Touch bereitgestellt.

5.3.3 Lizenz

Das UiUI Kit steht unter der GNU AGPL 3 [AGP] Lizenz. Sie erlaubt die uneingeschränkte Modifikation des Toolkits unter den Voraussetzungen, die in der GNU APGL 3 in Sektion 5 näher aufgeführt sind und deren Kernpunkte ich hier kurz zusammenfassen möchte:

- a) Das eigene Toolkit muss als modifiziert gekennzeichnet und mit dem entsprechenden Datum versehen werden.
- b) Das eigene Toolkit muss unter die GNU AGPL 3 gestellt werden und einen gut einsehbaren Hinweis darauf enthalten.
- c) Das eigene Toolkit muss als Ganzes unter die GNU AGPL 3 gestellt werden und darf nicht in Teilen anders lizenziert werden.
- d) Auf jedem möglichen User Interface muss sich ein Hinweis auf die Lizenz befinden. Da es sich bei dem Toolkit selbst um ein User Interface handelt, also genau einer.



Abbildung 10: Beispiel einer Wiki-Seite (Eigenentwicklung)



Abbildung 11: Editieren der Wiki-Seite (Eigenentwicklung)

Ferner muss die Lizenz nicht auf Programmteile übertragen werden, mit denen das eigene Toolkit eine sogenannte "Aggregation" bildet. Diese würde unserer Ansicht nach auch bei der Einbindung des Toolkits in das Studierendenportal vorliegen. Dies bleibt aber noch zu klären.

5.3.4 Bewertung

Das UiUI Kit um eigene Funktionen zu einem eigenen Toolkit zu erweitern ist natürlich arbeitsaufwendiger, als ein bereits bestehendes Toolkit zu verwenden. Dazu kommen eventuell später auftretende Lizenzprobleme, die mit der Einbindung des Toolkits in das Studierendenportal auftreten könnten. Vor einer erfolgreichen Implementierung müssten wir uns dagegen absichern und näher über die bestehende Rechtslage informieren.

Die Vorteile würden allerdings überwiegen: Durch die niedrige Komplexität des UiUI Kits ist es leicht, sich in den vorhandenen Quellcode einzuarbeiten. Dadurch ist das Toolkit schnell anpassbar und um die von uns vorgesehenen Funktionen, wie etwa die Tabs, erweiterbar. Desweiteren ist es sehr komfortabel in ein bestehendes Projekt zu integrieren, da es rein in HTML und WebKit/CSS, ohne Einsatz von Javascript, geschrieben ist. Das ist zu bevorzugen, da die Übergangseffekte mit Ajax ohnehin im Framework und nicht im Toolkit implementiert werden.

5.4 Entscheidung

Nach eingehender Betrachtung der Unterschiede zwischen den drei Endkandidaten haben wir uns abschließend gegen iUI und ein eigenes Toolkit und für iWebkit entschieden. iWebkit zeichnet sich den anderen beiden gegenüber vor allem durch die aktivste Community aus. Während sich iUI primär über Google Code [iUIb] organisiert, existiert für iWebkit eine eigene aktive Community-Seite [iWeb] mit zahlreichen Erweiterungen und Fixes. UiUI hat keine organisierte Community.

Bei der Überprüfung der drei Kandidaten auf ihre Erweiterbarkeit, schneidet das UiUI Kit immer noch am besten ab, da es die geringste Komplexität aufweist. Allerdings ist der iWebkit Source Code trotz seiner Komplexität übersichtlich, aufgeräumt und am besten dokumentiert. Er befindet sich außerdem noch immer in der aktiven Entwicklung seitens der Hauptentwickler, unterstützt durch die Community. Ebenso aktiv sind die Entwickler von iUI, allerdings ist der Source Code und die Dokumentation weniger überzeugend.

Den Ausschlag gibt allerdings das Design, das bei iWebkit am meisten überzeugt. iWebkit hat einfach die schönsten und saubersten Grafiken und ist das einzige Toolkit in der engeren Auswahl, das verschiedene Skins beinhaltet. Als Erweiterung lässt sich sogar das Design einer Tabbar finden, die allerdings manuell in das Kern-Toolkit eingebunden werden muss.

6 Design

Aus der Evaluation vorhandener Webstandards und Toolkits folgt in diesem Abschnitt nun ein konkretes Designkonzept, das sich an unseren Erkenntnissen orientiert. Zunächst wird dabei nochmal auf die Grundlagen, wie etwa die Besonderheiten einer Web-Applikation, eingegangen. Der folgende Abschnitt beschreibt eine auf mobile Web-Applikationen optimierte Navigation. Im nächsten Teil wird dann die geplante Seitenaufteilung anhand des Campuskartenprojektes erläutert. Der Abschnitt endet mit einer Vorschau des geplanten Prototypen.

6.1 Grundlagen

Im Gegensatz zu durchschnittlichen Webseiten sollten mobile Web Applikationen nicht durch ihre Fülle an Features auffallen, sondern durch die präzise und simple Hilfe bei der Erfüllung einer klar definierten und abgegrenzten Aufgabe.

Die Aufgabe dieses Projekts ist die Bereitstellung eines an Orte der realen Welt gebundenen Wikis. Dabei dient die verwendete Campuskarte in diesem Kontext als Navigationselement zwischen den Wikiseiten.

Dem Benutzer muss es möglich sein, Einträge anzulegen, zu verändern, zwischen ihnen zu navigieren und sich mit anderen sachbezogen über Änderungen auszutauschen. Mehr Features in dieser Applikation unterzubringen wäre überflüssig und würde das Ziel einer mobilen Web-Applikation verfehlen.

Bei der Umsetzung sollte außerdem Rücksicht auf den Benutzer und seine Gewohnheiten speziell in Bezug auf mobile Endgeräte gelegt werden. Darunter fällt der intelligente Einsatz von Cookies, um dem Benutzer den fließenden Wechsel zu und von anderen Applikationen zu der Web-Applikation zu ermöglichen, aber auch die Verwendung von speziellen Metatags, wie der „viewport“-Tag von Apple, mit dem unter anderem das Skalieren der Website im Safari des iPhones ausgeschaltet werden kann. Desweiteren sollte aufgrund der Bedienung über ein berührungssensitives Display von der Benutzung eines In-Place-Editors für das Wiki abgesehen werden. Dieser würde zwar zu mehr Übersichtlichkeit beitragen, könnte aber nur allzu leicht ungewollt ausgelöst werden, da auf einem Touch-Display neben dem Klicken auch über Fingertippen navigiert wird.

Um die Augen des Benutzers auf dem kleinen Display zusätzlich zu schonen, sollte das Design möglichst farbneutral ausgestaltet werden. Dazu gehört die Vermeidung von übermäßigem Branding, und bunten Adds. Zu viele Farben auf einem kleinen Display überlagern den Inhalt und fördern die Entstehung von Ungeduld und Frust beim Benutzer. Hervorhebungen und wichtige Passagen sollten deshalb auch klar ersichtlich gekennzeichnet werden, was durch die Verwendung von fetter Schrift und der Vermeidung von kursiven und unterstrichenen Passagen erreicht wird.

6.2 Navigation

Um eine Web-Applikation als solche erscheinen zu lassen, sollte eine vom Browser losgelöste Navigation vorhanden sein. Diese Navigation sollte zielführend und sparsam um-

gesetzt werden, um einerseits die Benutzerfreundlichkeit zu erhöhen und andererseits kostbaren Platz auf dem kleinen Display zu sparen. Am besten wird eine solche Navigation mit Hilfe einer Tabbar am unteren Ende der Seite, in Kombination mit eindeutigen und leicht zu erfassenden Symbolen oder kurzen prägnanten Bezeichnern, umgesetzt. Diese Methode ist gleichzeitig am platzsparendsten und am einprägsamsten für den Benutzer. Zusätzliche Navigationselemente und Buttons seitlich des Seitentitels sind zu empfehlen und bereits in vielen mobilen Applikationen, etwa des iPhones, fest verankert.

Um Verwirrung seitens des Benutzers zu vermeiden, sollte nur in vertikale Richtung gescrollt werden können, wie er es auch von einer durchschnittlichen Webseite gewohnt ist. Der Wechsel zwischen den verschiedenen Seiten sollte mit Ajax realisiert werden, einerseits um die Applikation dynamischer und somit mehr nach einer nativen Applikation erscheinen zu lassen, andererseits um eventuelle Performancegewinne durch Caching zu ermöglichen. Nur mit Ajax ist außerdem die Realisierung einer iPhone Web-Applikation unter Benutzung des „apple-mobile-web-app-capable“ Metatags möglich, der die Applikation im Vollbildmodus darstellt, da der Klick auf einen beliebigen Link ansonsten zum Verlassen des Vollbildmodus führen würde, um diesen im Safari zu öffnen.

6.3 Seitenaufteilung

Die Seitenaufteilung sollte nach dem „Head-Body-Foot“-Konzept umgesetzt werden. Dabei wird im „Head“ der Seitentitel und Actionbuttons, im „Body“ der Inhalt und im „Foot“ die Navigation angezeigt. Dieses Prinzip spart Platz und fördert die Übersichtlichkeit der Applikation. Während der Head und der Body scrollbar sein sollten, würde es sich anbieten, den Foot am Seitenboden zu verankern, um ständig auf die Navigation zugreifen zu können. Branding sollte auf den Head reduziert und auch dort minimiert werden, um die Übersichtlichkeit zu erhöhen.

Die wichtigsten Informationen sollten übersichtlich und kompakt auf einer Seite zusammengefasst und präsentiert werden, etwa auf der Hauptseite des Wikis. Desweiteren sollten keine übermäßig großen Zwischenräume oder sonstige leeren Flächen gelassen werden, um den Aufwand des vertikalen Scrollens für den Benutzer so gering wie möglich zu halten.

Die Campuskarte sollte fest verankert den Platz zwischen Head und Foot komplett ausfüllen. Wenn allerdings keine Navigation benötigt wird, sollte diese ausgeblendet werden, um dem Benutzer die größtmögliche Fläche zum Erkunden der Karte bereitzustellen. Eingabefelder sollten, soweit möglich, über drop-down Menus verwirklicht werden, um dem Benutzer das unbequeme Eingeben von Texten zu ersparen, so etwa bei Daten, Uhrzeiten oder jeglichen vorhersehbaren Daten.

6.4 Vorschau

Ein Prototyp, der nach den vorgegebenen Richtlinien umgesetzt wird, sollte also ungefähr so aussehen:

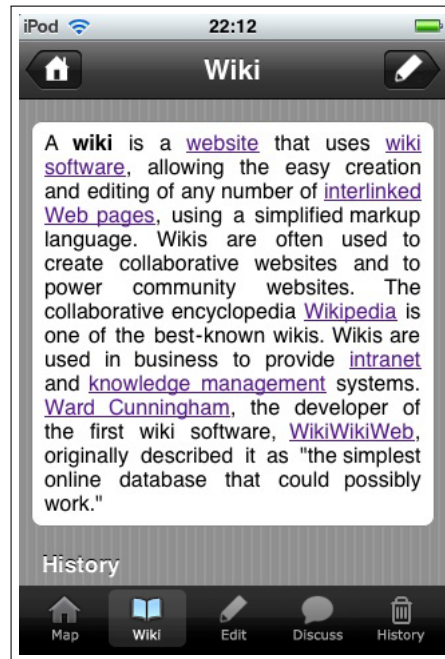


Abbildung 12: Ein Mockup des späteren Designs

7 Prototyp

In dem Prototypen versuchen wir die gewonnenen Erkenntnisse umzusetzen und entwickeln eine Webapplikation, die eine Vorabversion darstellt. Wir haben versucht die Seitenaufteilung und die Standards soweit umzusetzen wie das möglich war und sind dabei leider relativ früh auf Probleme gestoßen die in diesem Kapitel erläutert werden.

7.1 Auftretende Probleme

Leider ist auch in der aktuellen Version 4.6 von iWebKit immer noch keine Tabbar enthalten. Allerdings soll eine solche laut Hauptentwickler „Christopher“ schon mit Version 5.0 nachgereicht werden. [new09] Da das Design aber fundamental auf einer Tabbar aufbaut, muss sie vorläufig getrennt implementiert werden. Dabei treten allerdings unerwartete Probleme auf.

Um ein fest positioniertes Designelement wie eine Tabbar in einer HTML Seite auch beim Scrollen an ihrem Platz zu halten, wird klassisch dem CSS Attribut „position“ der Wert „fixed“ zugewiesen. Dieser bewirkt eine absolute Positionierung des Elements, die sich auch beim Scrollen nicht ändert. Obwohl WebKit diese Wertbelegung des Attributs im allgemeinen unterstützt, ist sie aber nicht im WebKit des mobilen Safari implementiert. [Flo09] Also muss die Tabbar anders fest positioniert werden. Dazu gibt es zwei grundlegend verschiedene Herangehensweisen:

Mit Hilfe von Javascript könnte die Position der Tabbar nach einem bei Laufzeit auftretenden onScroll Event neu berechnet und die Tabbar entsprechend mitverschoben

werden. Dazu existiert auch schon Beispielquelltext. [Mar09] [Ric08] Gut und ausführlich beschreibt Richard Herrera auf seinem Blog, wie das Problem mit Javascript zu umgehen ist und veröffentlicht seinen Code und ein Beispielvideo. [Ric08] Während er es schafft, die Usability zu erhalten und auch den Landscape Mode ermöglicht, ist sein Script leider sehr rechenintensiv und performt auf schwächeren Geräten nur sehr ruckelig und nicht hinnehmbar.

Die andere Herangehensweise arbeitet nicht auf Scriptebene, sondern verändert die Struktur des HTML Codes. Dabei wird die Tabbar am Seitenboden absolut positioniert und der Seiteninhalt per scrollbarem Blockelement über dieser eingefügt. So wird der gewünschte Effekt erzeugt und die Performance wird nicht geschmälert. Allerdings leidet die Usability stark, da das Blockelement nur ungewohnt mit zwei Fingern anstatt wie gewohnt mit einem Finger gescrollt werden kann. Dazu kommt, dass innerhalb des Blockelements keine onScroll Events mehr möglich sind.

7.2 Entscheidung

Beide Herangehensweisen bringen unbestritten viele Nachteile mit sich. Am Ende überwiegen allerdings die Vorteile, die der Ansatz eines scrollbaren Blockelements mit sich bringt. Zum Einen schränkt die ungewohnte Bedienung der Applikation ihre Usability weit weniger ein, als es die starken Performanceeinbußen tun, die das Skript mit sich bringt. Dazu kommt, dass mit dieser Möglichkeit weit weniger in den bereits bestehenden Code eingegriffen wird, als wenn das Javascript verändert werden würde.

Bei der Implementierung der Tabbar handelt es sich nur um eine Übergangslösung, die durch die Version 5.0 von iWebkit abgelöst werden wird. Der zusätzliche Code, der durch das scrollbare Blockelement anfällt, kann allerdings im Gegensatz zum Javascript Code sehr viel schneller wieder entfernt werden und greift weniger in die inneren Strukturen von iWebkit ein. Auch aus diesem Grund ist die zweite Herangehensweise zu bevorzugen.

7.3 Umsetzung

Da der Sourcecode des Prototypen zu groß ist um in vollständig in diesem Paper darzustellen befinden sich die wichtigsten dazu gehörenden Dateien im Anhang des Papers. Der komplette Sourcecode ist auf Nachfrage als Archiv verfügbar. Die Abbildungen 13, 14, 15 und 16 zeigen Screenshots vom Prototypen.

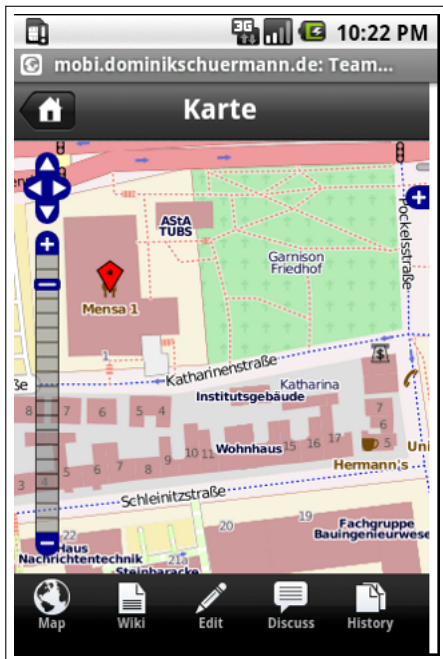


Abbildung 13: Prototyp: Karte

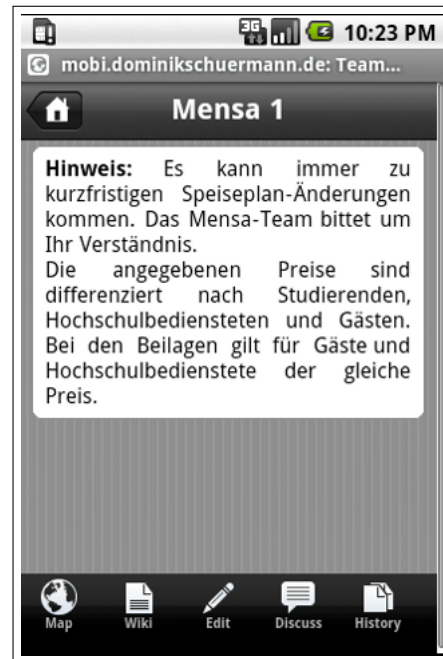


Abbildung 14: Prototyp: Wiki



Abbildung 15: Prototyp: Editieren



Abbildung 16: Prototyp: Diskussion

8 Zusammenfassung

Abschließend mussten wir feststellen, dass gerade bei der Umsetzung des Prototypen viele Probleme hinsichtlich der Webstandards auftraten, da der CSS-Standard auf dem iPhone und dem Android nicht vollständig implementiert wurde. Als Referenz gilt hier „The Great WebKit Comparison Table“ [Weba], in der die Implementierungen von CSS, HTML und JavaScript in den unterschiedlichen WebKit-Browsern verglichen werden. Der Prototyp zu unserem Paper stellt nur ein Prototyp hinsichtlich der Seitenaufteilung, des Designs und der Umsetzung von Webstandards dar. Somit wären die weiteren Schritte eine nutzbare mobile Webapplikation zu entwickeln um die Anforderungen an die Funktionalität zu implementieren.

9 Abbildungsverzeichnis

1	Ausstattung der iPhone und iPod touch Geräte [Wik09d]	6
2	W3C Validator angewand auf die mobile Version von TV Movie [W3Cf] .	10
3	Ergebnis von google.mobi durch mobiReady [mobb]	12
4	Übersichtsseite, iWebkit (Eigenentwicklung)	18
5	Wiki-Artikel, iWebkit (Eigenentwicklung)	18
6	Beispiel einer iUi Seite [iUia]	20
7	Weiteres Beispiel einer iUi Seite [iUia]	20
8	Beispiel Geräte mit Webkit (Eigener Test)	21
9	Beispiel Geräte ohne Webkit (Eigener Test)	21
10	Beispiel einer Wiki-Seite (Eigenentwicklung)	23
11	Editieren der Wiki-Seite (Eigenentwicklung)	23
12	Ein Mockup des späteren Designs	27
13	Prototyp: Karte	29
14	Prototyp: Wiki	29
15	Prototyp: Editieren	29
16	Prototyp: Diskussion	29

10 Literatur

- [ACI] Acid browser tests - the acid3 test. URL <http://acidtests.googletoad.com>.
- [AGP] GNU affero general public license - GNU project - free software foundation (FSF). URL <http://www.gnu.org/licenses/agpl.html>.
- [Anda] Android hub and household gadgets coming this year | android phone fans. URL <http://phandroid.com/2009/07/24/android-hub-and-household-gadgets-coming-this-year/>.
- [Andb] Android internal structure: A look into google android's internal structure simply put for anyone to understand. URL <http://www.brighthub.com/mobile/google-android.aspx>.
- [Andc] Android sdk. URL <http://developer.android.com/sdk/index.html>.
- [App07] Apple Inc. iPod touch - Technical Specifications, October 23, 2007. URL <http://web.archive.org/web/20071023075137/http://www.apple.com/de/ipodtouch/specs.html>.
- [App08] Apple Inc. iPhone - Technical Specifications, May 3, 2008. URL <http://web.archive.org/web/20080503120747/http://www.apple.com/iphone/specs.html>.
- [App09a] Apple Inc. Apple Developer Connection - Darwin, September 29, 2009. URL <http://developer.apple.com/Darwin/>.

- [App09b] Apple Inc. Apple iPhone 3Gs Features - Home Screen, September 29, 2009. URL <http://www.apple.com/iphone/iphone-3gs/home-screen.html>.
- [App09c] Apple Inc. iPhone 3g - Technical Specifications, September 21, 2009. URL <http://www.apple.com/iphone/specs-3g.html>.
- [App09d] Apple Inc. iPhone 3Gs, September 29, 2009. URL <http://www.apple.com/iphone/iphone-3gs/>.
- [App09e] Apple Inc. iPhone 3gs - Technical Specifications, September 21, 2009. URL <http://www.apple.com/iphone/specs.html>.
- [App09f] Apple Inc. iPhone Human Interface Guidelines for Web Applications, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09g] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 11, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09h] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 13, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09i] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 17, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09j] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 18, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09k] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 21, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09l] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 22, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.

- [App09m] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 23, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09n] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 24, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09o] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 25, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09p] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 30, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09q] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 31, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09r] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 33, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09s] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 34, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09t] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 35, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09u] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 36, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09v] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 42, Januar 6, 2009. URL <http://developer.apple.com/safari/>

library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/
iPhoneWebAppHIG.pdf.

- [App09w] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 42, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09x] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 46, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09y] Apple Inc. iPhone Human Interface Guidelines for Web Applications - Seite 7, Januar 6, 2009. URL <http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneWebAppHIG.pdf>.
- [App09z] Apple Inc. iPhone OS Technological Overview, May 27, 2009. URL <http://developer.apple.com/iPhone/library/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iPhoneOSTechOverview.pdf>.
- [App09-27] Apple Inc. iPhone OS Technological Overview - iPhone OS Technologies, May 27, 2009. URL <http://developer.apple.com/iPhone/library/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iPhoneOSTechOverview.pdf>.
- [App09-28] Apple Inc. iPhone3gs - features - safari, September 21, 2009. URL <http://www.apple.com/iphone/iphone-3gs/safari.html>.
- [App09-29] Apple Inc. iPod touch - Technical Specifications, September 21, 2009. URL <http://www.apple.com/ipodtouch/specs.html>.
- [App09-30] Apple Inc. Mango Browser for iPhone, September 29, 2009. URL <http://itunes.apple.com/WebObjects/MZStore.woa/wa/viewSoftware?id=312259486&mt=8>.
- [App09-31] Apple Inc. VanillaSurf Browser for iPhone, September 29, 2009. URL <http://itunes.apple.com/WebObjects/MZStore.woa/wa/viewSoftware?id=302678135&mt=8>.
- [BSD] Open source initiative OSI - the BSD License:Licensing | open source initiative. URL <http://www.opensource.org/licenses/bsd-license.php>.
- [Chra] Chrome supports video tag. URL <http://googlechromereleases.blogspot.com/2009/05/dev-channel-release-301822.html>.

- [Chrb] Google chrome faq. URL <http://www.google.com/chrome/intl/en-GB/webmasters-faq.html>.
- [Dev] DeviceAtlas | DeviceAtlas. URL <http://deviceatlas.com/>.
- [dot07] dotMobi mobile web developers guide. 2007.
- [Flo09] Florian Bösch. css position fixed and overflow scroll, Oktober 9, 2009. URL http://groups.google.de/group/iphonewebdev/browse_thread/thread/5b3c015ee9c2cde4.
- [Fou] Free Software Foundation. The gnu general public license - gnu project - free software foundation (fsf). URL <http://www.gnu.org/licenses/gpl.html>.
- [Hic] Ian Hickson. Hmtl audio/video codec streit. URL <http://lists.whatwg.org/htdig.cgi/whatwg-whatwg.org/2009-June/020620.html>.
- [HTCa] HTC - products - T-Mobile g1 - specification. URL <http://www.htc.com/www/product/g1/specification.html>.
- [HTCb] HTC launches hero android smartphone with custom HTC sense UI and flash support. URL <http://www.mobileburn.com/news.jsp?Id=7337>.
- [Ins] Instant mobilizer. URL <http://instantmobilizer.com/>.
- [iUia] Introduction - iui - iUI introduction wiki page. - project hosting on google code. <http://code.google.com/p/iui/wiki/Introduction>. URL <http://code.google.com/p/iui/wiki/Introduction>.
- [iUIb] iui - project hosting on google code. URL <http://code.google.com/p/iui/>.
- [iWea] iWebKit - make a quality iPhone website or webapp. URL <http://iwebkit.net/>.
- [iWeb] iWebKit community • index page. URL <http://community.iwebkit.net/>.
- [Jav] *JavaScript*. Galileo Computing. ISBN 978-3-89842-234-5.
- [Mar09] Marcelo Wolfgang. fixed position element in mobile safari, Oktober 9, 2009. URL http://groups.google.com/group/iphonewebdev/browse_thread/thread/7eed27e1377dfc98.
- [moba] Home | mobiThinking. URL <http://mobithinking.com/>.
- [mobb] mobiReady - dotMobi compliance & mobileOK checker. URL http://ready.mobi/launch.jsp?locale=en_EN.
- [Moz] Mozilla supports audio and video. URL https://developer.mozilla.org/En/Using_audio_and_video_in_Firefox.

- [new09] news.iwebkit.net. Comment on Update Soon, Oktober 9, 2009. URL <http://news.iwebkit.net/?p=209>.
- [Ric08] Richard Herrera. Fixed positioning in mobile safari, August 5, 2008. URL <http://doctyper.com/archives/200808/fixed-positioning-on-mobile-safari/>.
- [Sel] SELFHTML: stylesheets / stylesheets und HTML. URL <http://de.selfhtml.org/css/intro.htm>.
- [UiU] Universal iPhone UI kit. URL <http://www.minid.net/iphone/>.
- [Ven] Source: Apple asked google not to use multi-touch in android, and google complied | VentureBeat. URL <http://digital.venturebeat.com/2009/02/09/apple-asked-google-not-to-use-multi-touch-in-android-and-google-complied/>.
- [W3Ca] W3c - world wide web consortium. URL <http://www.w3.org/>.
- [W3Cb] W3C patent policy. URL <http://www.w3.org/2004/02/05-patentsummary.html>.
- [W3Cc] W3C. Mobile web best practices 1.0, basic guidelines. URL <http://www.w3.org/TR/mobile-bp/>.
- [W3Cd] W3C. W3c - current members. URL <http://www.w3.org/Consortium/Member/List>.
- [W3Ce] W3C. W3c - extensible markup language - iso 8879:1986. URL <http://www.w3.org/TR/2000/WD-xml-2e-20000814.html>.
- [W3Cf] W3C. W3c - validator (w3c mobileok checker). URL <http://validator.w3.org/mobile/>.
- [Weba] The great WebKit comparison table. URL <http://quirksmode.org/webkit.html>.
- [Webb] The WebKit open source project - WebKit project goals. URL <http://webkit.org/projects/goals.html>.
- [Wika] Ajax (Programmierung) – wikipedia. URL [http://de.wikipedia.org/wiki/Ajax_\(Programmierung\)](http://de.wikipedia.org/wiki/Ajax_(Programmierung)).
- [Wikb] Extensible hypertext markup language – wikipedia. URL <http://de.wikipedia.org/wiki/XHTML>.
- [Wikc] Framework – wikipedia. URL <http://de.wikipedia.org/wiki/Framework>.

- [Wikd] WebKit – wikipedia. URL <http://de.wikipedia.org/wiki/WebKit>.
- [Wik09a] Wikipedia. Registerkarte — wikipedia, die freie enzyklopädie, 2009. URL <http://de.wikipedia.org/w/index.php?title=Registerkarte&oldid=63968347>. [Online; Stand 30. September 2009].
- [Wik09b] Wikipedia. Wi-fi — wikipedia, die freie enzyklopädie, 2009. URL <http://de.wikipedia.org/w/index.php?title=Wi-Fi&oldid=64110932>. [Online; Stand 30. September 2009].
- [Wik09c] Wikipedia Community. iPhone OS version history, September 21, 2009. URL http://en.wikipedia.org/wiki/IPhone_OS_version_history.
- [Wik09d] Wikipedia Community. List of iphone and ipod touch models, September 3, 2009. URL http://en.wikipedia.org/wiki/List_of_iPhone_and_iPod_Touch_models#Features.

11 Glossar

Adds Werbung im Internet.

Ajax „Ajax ist ein Apronym für die Wortfolge „Asynchronous JavaScript and XML“. Es bezeichnet ein Konzept der asynchronen Datenübertragung zwischen einem Server und dem Browser, das es ermöglicht, innerhalb einer HTML-Seite eine HTTP-Anfrage durchzuführen, ohne die Seite komplett neu laden zu müssen. [...] womit Ajax eine Schlüsseltechnik zur Realisierung des Web 2.0 darstellt.“ [Wika]

Applikation Synonym für Computerprogramm.

Bottleneck Ein Teil eines Programms, bei dem besonders viele Ressourcen benötigt werden. Häufig nehmen Bottlenecks sehr viel kostbare Zeit in Anspruch und sollten deshalb möglichst vermieden werden.

Clickstream Ein Clickstream ist die Abfolge der Links, auf die ein Benutzer auf einer Website klickt.

clientseitig Auf dem Endgerät des Benutzers.

Cookie Cookies sind clientseitig im Browser persistent gespeicherte Daten.

Cascading Stylesheets (CSS) „Stylesheets sind eine unmittelbare Ergänzung zu HTML. Es handelt sich dabei um eine Sprache zur Definition von Formateigenschaften einzelner HTML-Elemente. Mit Hilfe von Stylesheets können Sie beispielsweise bestimmen, dass Überschriften 1. Ordnung einen großen Schriftgrad aufweisen, in der Schriftart Helvetica, aber nicht fett erscheinen und mit einem Abstand von 1,75 Zentimeter zum darauffolgenden Absatz versehen werden. Angaben dieser Art sind mit reinem HTML nicht möglich.“ [Sel]

Darwin Ein freies Unix-Betriebssystem auf dem Mac OS basiert.

Framework „Ein Framework (engl. für „Rahmenstruktur, Fachwerk“) ist ein Programmiergerüst, das in der Softwaretechnik, insbesondere im Rahmen der objektorientierten Softwareentwicklung sowie bei komponentenbasierten Entwicklungsansätzen, verwendet wird.“ [Wike]

Home Screen Der Desktop eines iPhones, auf dem sich Links zu den Applikationen befinden.

hover-Effekte Effekte die durch das überfahren eines bestimmten abgegrenzten Bereichs mit dem Mouse-Zeiger ausgelöst werden.

iCal Kalendersoftware von Mac OS.

iPhone Ein mobiles Endgerät, das eine Kombination aus MP3-Player, Handy, Email-Client und Web-Browser bereitstellt. Auf ihm können beliebig viele Applikationen nachinstalliert werden.

iPhone OS Das Betriebssystem des iPhones.

iTunes Musikplayer von Mac OS.

JavaScript „Mit JavaScript kann man die eher beschränkten Möglichkeiten von HTML erweitern. Es handelt sich hierbei um eine clientseitige Programmiersprache, das heißt, alles läuft im Browser ab, und man muss keine besonderen Servervoraussetzungen erfüllen.“ [Jav]

Logs Log-Dateien stellen eine Art Protokoll des dazugehörigen Programmes dar. Das Programm protokolliert auftretende Fehlermeldung und Informationen in einer solchen Datei.

Mac OS X Betriebssystem der Apple Computer.

MacBook Laptop von Apple.

Magnetometer Messgerät für das Erdmagnetfeld.

mobiles Endgerät Geräte, die leicht genug sind, um ohne Probleme am Körper getragen werden können.

Multi-Touch Display Berührungssensitives Display des iPhones und des iPod touch.

Open Source Open Source Software zeichnet sich dadurch aus, dass der Quelltext offen verfügbar ist, die Software beliebig kopiert und verbreitet werden darf und auch in veränderter Form weitergegeben werden darf.

Panning Das Verschieben eines Bildschirmabschnittes.

Pop-Up Menus Menus, die in einem eigenen Fenster geöffnet werden.

Safari Safari ist der Web-Browser von Mac OS.

Selection-Lists Eine Liste mit verschiedenen Optionen, aus denen eine ausgewählt werden kann.

serverseitig Auf Seiten des Internetservers.

SDK Ein Software Development Kit ist eine Sammlung von Programmen und Tools, die das Schreiben einer bestimmten Software vereinfachen sollen.

Subdomain Eine Subdomain ist eine Domain, die einer anderen Domain untergeordnet ist. Zum Beispiel ist „google“ eine Subdomain der Top-Level Domain „.com“.

Tab „Eine Registerkarte, nach dem englischen Begriff auch Tab genannt, ist eine Sortier- und Navigationshilfe, die der weiteren Unterteilung von Einzelelementen dient.“ [Wik09a]

Touchscreen Ein Bildschirm, der Benutzereingaben durch Berührungen ermöglicht.

UI Toolkit UI Toolkits sind eine Art Bibliothek, also in unserem Fall eine Sammlung von fertigen Scripten und Vorlagen, die genutzt werden kann um grafische Benutzeroberflächen zu bauen.

Web Clip Ein Feature des iPhones, das es ermöglicht, Web-Links als Icon auf dem Home Screen einzubinden.

WebKit „WebKit ist eine freie HTML-Rendering-Bibliothek, auf deren Grundlage ein Webbrowser gebaut werden kann. WebKit ist eine von der Firma Apple entwickelte Abspaltung der HTML-Engine KHTML, um sie als Grundlage für den Mac-OS-X-Webbrowser Safari einzusetzen. Sie wird mittlerweile von Apple, Nokia, Google und anderen weiter entwickelt.“ [Wikd]

Wi-Fi Wi-Fi ist ein Kunstbegriff der Wi-Fi Alliance, die Wireless-Produkte auf Kompatibilität nach Standards testen und bei bestandenem Test, das Wi-Fi Logo und Zertifikat vergeben. [Wik09b]

XHTML „Der W3C-Standard Extensible HyperText Markup Language (erweiterbare HTML; Abkürzung: XHTML) ist eine textbasierte Auszeichnungssprache zur Strukturierung und semantischen Auszeichnung von Inhalten wie Texten, Bildern und Hyperlinks in Dokumenten.“ [Wikb]

Zooming Das Vergrößern oder Verkleinern eines Bildschirmabschnittes auf einem Computer.

12 Anhang

12.1 Auszug vom Prototypen

12.1.1 index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta content="yes" name="apple-mobile-web-app-capable" />
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
    <meta content="minimum-scale=1.0, width=device-width, maximum-scale=0.6667, user-scalable=no" name="viewport" />
    <link href="css/iwebkit.css" rel="stylesheet" type="text/css" />
    <link href="css/style.css" rel="stylesheet" type="text/css" />
    <script src="javascript/functions.js" type="text/javascript"></script>
    <title>Teamprojekt Prototyp</title>
    <meta content="keyword1, keyword2, keyword3" name="keywords" />
    <meta content="Description_of_your_site" name="description" />

    <script language=javascript>
    <!--
    if (navigator.userAgent.match(/Android/i))
    {
      document.write("<link href=\"css/android.css\" rel=\"stylesheet\" type=\"text/css\" _/>");
    }
    -->
    </script>
    <script language=javascript>
    <!--
    if ((navigator.userAgent.match(/iPhone/i)) || (navigator.userAgent.match(/iPod/i)))
    {
      document.write("<link href=\"css/iphone.css\" rel=\"stylesheet\" type=\"text/css\" _/>");
    }
    -->
    </script>

    <link rel="stylesheet" type="text/css" href="css/map.css" />

    <script type="text/javascript" src="javascript/OpenLayers/OpenLayers.js"></script>
    <script type="text/javascript" src="http://www.openstreetmap.org/openlayers/OpenStreetMap.js"></script>
    <script type="text/javascript" src="javascript/util.js"></script>

    <script type="text/javascript">
    //<![CDATA[

    var map;

    var showPopupOnHover = false;
    text = new Array("Informationen zur Karte anzeigen", "Informationen zur Karte verstecken");

    function drawmap() {
      OpenLayers.Lang.setCode('de');

      map = new OpenLayers.Map('map', {
```

```

        projection: new OpenLayers.Projection("EPSG:900913"),
        displayProjection: new OpenLayers.Projection("EPSG:4326"),
        controls: [
new OpenLayers.Control.MouseDefaults(),
new OpenLayers.Control.Attribution()],
        maxExtent:
            new OpenLayers.Bounds(-20037508.34, -20037508.34,
                20037508.34, 20037508.34),

        numZoomLevels: 18,
        maxResolution: 156543,
        units: 'meters'
    });

// Noch mehr Kontrollelemente hinzufuegen..
map.addControl(new OpenLayers.Control.LayerSwitcher());
map.addControl(new OpenLayers.Control.PanZoomBar());

// Position und Zoomstufe der Karte
lon = 10.527099609314;
lat = 52.274053158164;
zoom = 17;

checkForPermalink();

// Layer hinzufuegen

layer_markers = new OpenLayers.Layer.Markers("Marker", { projection: new
    OpenLayers.Projection("EPSG:4326"), visibility: true, displayInLayerSwitcher:
    false });
layer_vectors = new OpenLayers.Layer.Vector("Zeichnungen", {
    displayInLayerSwitcher: false }); map.addLayer(layer_vectors); map.addLayer(
    layer_markers)
layers = new Array();
layer_layerMapnik = new OpenLayers.Layer.OSM.Mapnik("Mapnik");
map.addLayer(layer_layerMapnik)
layers.push(new Array(layer_layerMapnik, 'layer_layerMapnik'));
setLayer(0);

// An die richtige Stelle springen..
jumpTo(lon, lat, zoom);

// Benutzte Marker Icons hinzufuegen..
icons = new Array();
icons[0] = new Array('http://openlayers.org/api/img/marker.png
    ', '21', '25', '0.5', '1');

// Marker hinzufuegen
addMarker(layer_markers, 10.526209115922, 52.274716199717, "<b>Mensa_1</b><p- /><a_-
    href=\"wiki.html\">Zu_diesem_Wiki-Eintrag_springen</a>", true, 0);

geometries = new Array();

```

```

// Nochmal was..
jumpTo(lon, lat, zoom);

checkUtilVersion(4);
}

//]]>
</script>
</head>

<body onload="drawmap();">

  <div id="topbar">
    <div id='leftnav '>
      <a href="index.html"></a>
    </div>
    <div id='title '>Karte</div>
  </div>

  <div id="content">
    <div id="map"></div>
  </div>

  <ul id='tabbar '>
    <a href="index.html"><span><li class="map">Map</li></span></a>
    <a href="wiki.html"><span><li class="wiki">Wiki</li></span></a>
    <a href="edit.html"><span><li class="edit">Edit</li></span></a>
    <a href="discuss.html"><span><li class="discuss">Discuss</li></span></a>
    <a href="history.html"><span><li class="history">History</li></span></a>
  </ul>

</body>
</html>

```

12.1.2 style.css

```

/* default css */

div#content {
  top: -12px;
  height: 363px;
  overflow: auto;
}

ul#tabbar {
  position: absolute;
  top: 370px;
  height: 60px;
  width: 100%;
  background: black url('../images/teamprojekt_tabbar.png') repeat-x;
  z-index: 2000;
  text-align: center;
  margin: 0 auto;
  padding: 3px 0 0;
  list-style: none;
}

#tabbar span {
}

```

```

#tabbar li {
  padding: 29px 3px 3px;
  float: left;
  position: relative;
  width: 54px;
  color: #bbb;
  font-size: 10px;
  text-decoration: none;
  font-weight: bold;
  margin: 0 auto;
}

#tabbar a {
  color: #bbb;
  text-decoration: none;
  font-weight: none;
}

#tabbar .map {
  background: url("../icons/icon_blog.png") no-repeat center 0
}

#tabbar .wiki {
  background: url("../icons/icon_document.png") no-repeat center 0
}

#tabbar .edit {
  background: url("../icons/icon_pencil.png") no-repeat center 0
}

#tabbar .discuss {
  background: url("../icons/icon_post.png") no-repeat center 0
}

#tabbar .history {
  background: url("../icons/icon_copy.png") no-repeat center 0
}

```

12.1.3 android.css

```

/* Android specific css */

div#content {
  top: -12px;
  height: 100%;
  overflow: auto;
}

ul#tabbar {
  position: absolute;
  top: 370px;
  height: 60px; /* old: 52 */
  width: 100%;
  background: black url('../images/teamprojekt_tabbar.png') repeat-x;
  z-index: 2000;
  text-align: center;
  margin: 0 auto;
  padding: 3px 0 0;
  list-style: none;
}

```