

Hinweis:

Diese Druckversion der Lerneinheit stellt aufgrund der Beschaffenheit des Mediums eine im Funktionsumfang stark eingeschränkte Variante des Lernmaterials dar. Um alle Funktionen, insbesondere Animationen und Interaktionen, nutzen zu können, benötigen Sie die On- oder Offlineversion. Die Inhalte sind urheberrechtlich geschützt.
©2018 Beuth Hochschule für Technik Berlin

EZS - Einführung zum Studienmodul




1 Einführung und Überblick

Der Titel Internetanwendungen für mobile Geräte eröffnet einen nicht unbeträchtlichen Gestaltungsspielraum bei der Themenauswahl und Schwerpunktsetzung einer Lehrveranstaltung. Könnten darunter doch auch native Anwendungen verstanden werden, die für eine der gängigen Anwendungsplattformen wie iOS oder Android entwickelt werden, ihre Funktionalität jedoch auf Grundlage von Anwendungskomponenten erbringen, die „über das Internet“ zugegriffen werden.

Dieses Kriterium erfüllen derzeit zahlreiche der als Apps kaum mehr aus unserem Alltag wegzudenkenden nativen mobilen Anwendungen. Auch wenn diese auf einem Endgerät installiert werden und ggf. im Offline Modus lauffähig sind, handelt es sich hierbei doch in vielen Fällen um Bestandteile eines verteilten Softwaresystems, das zusätzlich über Komponenten verfügt, die zentral auf einem oder mehreren öffentlich zugänglichen Servern – in vielen Fällen in der sogenannten „Cloud“ – für die Nutzer der App zur Verfügung gestellt werden.

Cloud Computing

Eine fundierte und differenzierte Definition dieses sehr inflationär gebrauchten Begriffs finden Sie z. B. unter

 <http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>.

Die Behandlung nativer Anwendungen im Hinblick auf Aspekte der Client - Server Interaktion wäre daher durchaus ein Themenbereich, der im Rahmen der vorliegenden Lehrveranstaltung denkbar wäre.

Webanwendungen

Gegenüber einer solchen weit gefassten Auslegung des Veranstaltungstitels schränken wir unsere Betrachtung auf Webanwendungen ein, die auf mobilen Endgeräten lauffähig sind. Darunter verstehen wir Anwendungen, die auf Basis der standardisierten Webtechnologien HTML, CSS und JavaScript entwickelt und durch einen auf einem Endgerät installierten Browser ausgeführt werden. Als solche ist ihre Funktionsfähigkeit zunächst undenkbar, ohne eine bestehende Internetverbindung zu einem Server, der die für die Ausführung der Anwendung erforderlichen Ressourcen bereit stellt.

Wie wir am Ende der Veranstaltung sehen werden, stehen uns jedoch auch für Webanwendungen mittlerweile Ausdrucksmittel zur Verfügung, die deren Ausführung im Offline Modus erlauben. Ggf. ist dann lediglich für das initiale Laden und für Aktualisierungen der Anwendung ein Zugriff über das Internet erforderlich, entsprechend der Installation nativer mobiler Anwendungen über einen der dafür im Netz verfügbaren als „App Store“ bezeichneten Distributionskanäle.

Fat Client

Eine grundsätzliche Abgrenzung, die wir in Bezug auf den Veranstaltungsstoff vornehmen, ist also die Betrachtung von Anwendungen auf Basis von Webtechnologien als Alternative zu nativen mobilen Anwendungen. Nun bestehen aber für die Realisierung von Webanwendungen und der möglichen Verteilung von deren Funktionen auf browserseitig ausgeführte vs. serverseitig ausgeführte Komponenten zahlreiche Alternativen.

Deren entgegengesetzte Extrempositionen lassen sich in den Begriffen einer „Fat Client“ im Ggs. zu einer „Thin Client“ Architektur fassen. Ein wesentliches Kriterium zur Unterscheidung der beiden Alternativen stellt bei Webanwendungen die Frage dar, inwieweit durch den Server dynamische Bestandteile der durch den Client realisierten Nutzeroberfläche in Form von HTML-Markup bereitgestellt werden, oder ob der Server lediglich strukturierte Daten in XML oder JSON sowie Medienressourcen liefert, die der Client für den Aufbau seiner Nutzeroberfläche verwendet. Diesbezüglich wird sich die vorliegende Veranstaltung ausschließlich mit letzterer Alternative beschäftigen und die hierfür erforderlichen Ausdrucksmittel auf Seiten von Client und Server exemplarisch vorstellen.

Damit verfolgen wir jedoch nicht nur im Ansatz die Grundzüge einer „Fat Client“ Architektur für Web Applikationen. Wir nähern uns zugleich auch sehr nahe an Architekturprinzipien an, die für native mobile Anwendungen charakteristisch sind, bei welchen es sich ihrerseits um typische „Fat Client“ Anwendungen handelt, die mittels endgerätespezifischer nativer Softwarekomponenten realisiert werden. Aus dieser Perspektive stellen die von uns verwendeten Webtechnologien lediglich eine Implementierungsalternative gegenüber den Ausdrucksmittel nativer Anwendungen dar, der jedoch identische Architekturprinzipien zugrunde gelegt werden können.

Insbesondere wird die Veranstaltung daher auch auf die Verwendung serverseitiger Anwendungskomponenten zum Zweck der Persistierung und Bereitstellung von Daten eingehen, welche von der auf dem Endgerät ausgeführten Anwendung verwendet werden.

Basics vs. Frameworks

Bestimmt ist Ihnen bewusst, dass die genannten Webtechnologien seit ihrer Entstehung in den 90er Jahren gewisse Generationswechsel durchlaufen haben, dass aber nicht alle derzeit in Gebrauch befindlichen Endgeräte bzw. die darauf installierten Browser den aktuellen Stand dieser Technologien unterstützen. Da im Echteinsatz laufende Webanwendungen jedoch üblicherweise für eine maximale Anzahl an potentiellen Nutzern zugänglich sein sollen, stellt diese Situation *Entwickler* in der Praxis vor die Herausforderung, die Funktionsfähigkeit ihre Anwendungen für eine möglichst große Anzahl an Browsern zu gewährleisten.

Pluralform „Entwickler“

Hier und im folgenden verwenden wir zum Zweck der Lesbarkeit für Pluralformen das generische Maskulinum und weisen darauf hin, dass darin stets alle genderbezogenen Ausprägungen inkludiert sind.

No jQuery

Nicht zuletzt die dafür erforderliche Handhabung verschiedener Varianten von JavaScript APIs hat die Bereitstellung generischer Softwarekomponenten wie [!\[\]\(3e2231b1ad3ca8da8658228c00dd08e0_img.jpg\) jQuery](#) begünstigt, die sich neben den in laufender Weiterentwicklung befindlichen Grundlagentechnologien HTML, CSS und JavaScript als „de facto Standards“ in der Webentwicklung etabliert haben. Die Nutzung dieser Lösungen kann in der Praxis zwar mit erheblichen Effizienzgewinnen gegenüber einer „handgemachten“ Alternative verbunden sein, sie verschleiert aber auch bis zu einem gewissen Grade die darunter liegenden Standards.

Für den Autor dieses Studienmoduls wurde dies anhand eines Vergleichs zweier im Sommer 2013 durchgeführter Implementierungsprojekte offensichtlich, bei denen zum einen die aktuellen Standards, zum anderen das speziell für mobile Nutzerschnittstellen verfügbare Framework [!\[\]\(5361750c22c4e047a52f4eac1ec2d4cc_img.jpg\) jQuery Mobile](#) zum Einsatz kamen, welches auf jQuery basiert. Nicht zuletzt diese Erfahrung gab den Ausschlag dafür, im Rahmen der vorliegenden Lehrveranstaltung jQuery komplett aus der Betrachtung auszublenden und seine Verwendung bei der Umsetzung von Implementierungsbeispielen und Übungsaufgaben auszuschließen.

„Lab“-Ansatz

Im Gegenzug betrachten wir ausschließlich die aktuellsten Versionen der Standardtechnologien und insbesondere die aktuell verfügbaren standardisierten JavaScript APIs und loten diese im Sinne eines für eine Hochschullehrveranstaltung angemessenen „Lab-Ansatzes“ im Hinblick auf ihre Brauchbarkeit für die Umsetzung von Anforderungen an mobile Nutzerschnittstellen aus. Als Browser verwenden wir dafür zunächst ausschließlich Firefox, der für Android Geräte, z. T. in der  Betaversion, konsistent zu den Versionen für stationäre(re) Geräte verfügbar ist. Die hierfür entwickelten Implementierungsbeispiele wurden jedoch hinsichtlich ihrer Lauffähigkeit auf den Browsern Chrome und Safari für MacOS und iOS weitest gehend verifiziert bzw. angepasst.

Im Zuge der Implementierungsmaßnahmen erschien es jedoch auch sinnvoll, zur Erleichterung, aber auch zur Unterstützung der Übungsaufgaben wieder verwendbare Funktionalität in JavaScript-Bibliotheken auszulagern – exemplarisch anhand einer Komfortfunktion zur Ausführung von XMLHttpRequest. Diese Implementierungen sind gegenüber vergleichbaren Lösungen, die durch die genannten Frameworks bereitgestellt werden, zwar in verschiedener Hinsicht defizitär. Sie sollten jedoch in ihrer Funktionalität – und ggf. auch hinsichtlich der genannten Defizite – für Sie transparent sein und damit zum Verständnis der grundlegenden Ausdrucksmittel von Webtechnologien beitragen.

Nachhaltigkeit

Diese Ausdrucksmittel sind aber nicht nur sehr umfangreich, sondern werden im Laufe der kommenden Jahre durch eine fortschreitende Weiterentwicklung geprägt sein. Die Sinnhaftigkeit einer an nachhaltiger Stoffvermittlung interessierten Lehrveranstaltung kann daher nicht darin liegen, eine erschöpfende Einführung in die betreffenden Ausdrucksmittel zu geben. Sie sollte auch nicht versuchen, in Konkurrenz zur Gesamtmenge der zahlreichen Tutorials zu treten, die zur Erläuterung dieser Technologien existieren. Aus diesem Grunde fokussiert sich die Lehrveranstaltung in ihren praktischen Teilen weitgehend auf solche Ausdrucksmittel, denen eine zentrale Rolle für die jeweils betrachteten Funktionsaspekte von Webanwendungen zukommt und bezüglich derer wir eine vergleichsweise Langlebigkeit erwarten – auch wenn sie evtl. im einen oder anderen Fall durch komfortablere Lösungen im Rahmen von Standards oder darauf aufbauende Bibliotheken und Frameworks ergänzt werden.

In den theoretischen Abschnitten der Lerneinheiten beschäftigen wir uns mit verschiedenen Aspekten, vornehmlich aus den Bereichen Softwareentwicklung und HCI im weiteren Sinne, die wir für die jeweils behandelten Themen als relevant erachten. So ist es das Ziel der Veranstaltung, Sie durch einen Blick auf ausgewählte Ausdrucksmittel und Verwendungsaspekte von Webtechnologien dazu zu befähigen, sich ein weiteres Spektrum dieser Technologien und deren zukünftige Weiterentwicklungen selbständig anzueignen, entsprechend den von Ihnen ggf. zu bearbeitenden praktischen Anforderungen.

Nachfolgend geben wir einige Hinweise zum Aufbau der Lehrveranstaltung und zu Hintergründen und Reifegrad des hier vorliegenden Materials und nehmen dann eine kurze Klärung der wichtigsten Grundbegriffe vor, die uns in dieser Veranstaltung immer wieder begegnen werden.

2 Aufbau des Studienmoduls

Theorieaspekte

Im Bewusstsein darum, dass Ihnen im Rahmen der Lehrveranstaltung Web-Programmierung bereits grundlegende Kompetenzen in der Handhabung der eingangs genannten Webtechnologien vermittelt wurden, gibt die Veranstaltung in zwei einführenden Kapiteln zu **HTML** und **CSS** bzw. zu **JavaScript** lediglich einen groben Überblick über die jeweiligen Ausdrucksmittel. Ein Fokus liegt dabei zum einen auf Theorieaspekten, die aus Sicht des Autors von besonderem Interesse sind, sowie auf ausgewählten Anforderungen hinsichtlich der Gestaltung und des Verhaltens mobiler Anwendungen, anhand derer sich die Verwendung der genannten Ausdrucksmittel noch einmal exemplarisch illustrieren lässt.

Serverseitige Anwendungskomponenten

Danach werden mit **NodeJS** und **MongoDB** zwei aktuelle Technologien zur Realisierung serverseitiger Anwendungskomponenten eingeführt und gezeigt, wie Sie unter angemessener Verwendung der Ausdrucksmittel des **HTTP-Protokolls** auf diese Anwendungskomponenten zugreifen können. Die Entscheidung für NodeJS wurde insbesondere auch dadurch motiviert, dass es sich hierbei um eine serverseitige Ausführungsumgebung für JavaScript handelt und es Ihnen damit möglich ist, Ihr bezüglich JavaScript bereits erworbenes Wissen in die Umsetzung serverseitiger Implementierungsmaßnahmen einzubringen.

User Interface

Mit **Formularen** stellen wir den Nutzern unserer Anwendung dann ein User Interface (UI) für den lesenden und schreibenden Zugriff auf einen serverseitigen Datenspeicher zur Verfügung. Die Umsetzung dieses UI wird durch zahlreiche Funktionen in HTML, CSS und JavaScript unterstützt und ermöglicht u. a. auch das Hochladen von Dateiinhalten, die wir auf Seiten unserer NodeJS Implementierung verarbeiten können.

Medien-Wiedergabe

Daran anschließend werden die **Multimedia Elemente**, die uns in HTML aktuell zur Verfügung stehen, zum einen mit Blick auf die Wiedergabe von Inhalten im Browser kennen lernen. Zum anderen werden wir dort zeigen, wie Sie die **Kamera** Ihres Endgeräts als Eingabevorrichtung zur clientseitigen Erstellung neuer Inhalte verwenden können.

Aktuelle Ausdrucksmittel zur **clientseitigen Datenspeicherung** bilden den Inhalt von Lerneinheit „LDS - Lokale Datenspeicherung“. Dort werden wir insbesondere mit **IndexedDB** eine **Datenbank** kennenlernen, die wir als lokale Alternative zur serverseitigen MongoDB Datenbank nutzen werden.

Offline Webanwendungen

Damit ausgestattet gehen wir in Lerneinheit „OFF - Offline Webapps“ einen letzten Schritt in Richtung von **Webanwendungen**, die funktional und im Hinblick auf die Nutzererfahrung analog zu nativen mobilen Anwendungen realisiert werden können. So werden wir ihnen dort zeigen, wie Sie **Webanwendungen offlinefähig** machen und als „**Webapps**“ auch von den Bedienelementen eines Browsers unabhängig machen können.

Vielleicht vermissen Sie in dieser Inhaltsübersicht grundlegende Themen, die für **mobile Anwendungen** unabhängig von den für die Umsetzung verwendeten Ausdrucksmitteln relevant sind. Aus Sicht des Autors gehören dazu insbesondere die physikalischen und technischen Grundlagen drahtloser Kommunikation, eine Betrachtung verschiedener Typen mobiler Endgeräte und ihrer Hardwareausstattung sowie eine Beschäftigung mit den Nutzungsszenarien für mobile Anwendungen. Insbesondere letztere erscheint erforderlich, um ein Verständnis für die rasante Durchdringung unseres Alltags zu gewinnen, welche für mobile Anwendungen insbesondere seit der Markteinführung des iPhone 2007 und der ersten Android Geräte 2008 konstatiert werden kann. Aber auch Potentiale für zukünftige Anwendungen lassen sich aus einer solchen Betrachtung beurteilen.

Zusatzinformationen

Ausführungen zu den genannten Themen finden Sie in zwei Lerneinheiten eines durch den Autor dieses Skripts verfassten Skripts zu einer VFH Lehrveranstaltung, die unter dem Titel *Mobile Application Development* im 2. Semester des Masterstudiengangs Medieninformatik angeboten wird.

3 Stand und Struktur des Moduls

Das hier vorliegende Lehrmaterial ist entstanden auf Grundlage von zehn Lerneinheiten, die im Wintersemester 2013/2014 für eine Lehrveranstaltung unter dem Titel „Multimedia Engineering II“ an der Beuth Hochschule für Technik Berlin vorausschauend auf die Erstellung dieses Veranstaltungsskripts entwickelt wurden. Zielgruppe dieser Lehrveranstaltung waren Studierende im 5. Fachsemester des Bachelor-Präsenzstudiums Medieninformatik, die zuvor noch keine Einführung in die Verwendung von Webtechnologien erhalten hatten.

Aus diesem Grund waren insgesamt vier Lerneinheiten für die Beschäftigung mit HTML, CSS und JavaScript vorgesehen. In der Bezeichnung der Übungsaufgaben finden Sie die Kürzel HTM, CSS, JSL und JSR wobei JSR in der Lerneinheit JSL integriert ist. Abhängig vom tatsächlichen Kenntnisstand, den Sie als Teilnehmer in die Veranstaltung mitbringen, können diese Lerneinheiten im Hinblick auf die Bearbeitungszeit entsprechend der Kapitelnumerierung zu jeweils einer Lerneinheit zusammengefasst werden oder als eigenständige Lerneinheiten beibehalten werden, wobei JSR sich allein auf die Verwendung der XMLHttpRequest API beschränkt.

Dies gilt auch für die Lerneinheiten FRM und MFM zu Formularen, welches ebenfalls inhaltlich und auf Ebene der Übungsaufgaben in zwei Lerneinheiten untergliedert ist. Die anderen Kapitel entsprechen jeweils einer Lerneinheit. Dabei ist aber mit Blick auf Lerneinheit NJM zu beachten, dass hier mit NodeJS und MongoDB zwei serverseitige Technologien eingeführt werden, bezüglich derer eine Vertrautheit Ihrerseits nicht vorausgesetzt werden kann. Auch mit Blick auf den Umfang der textuellen Ausführungen könnte dieser Lerneinheit eine Bearbeitungszeit eingeräumt werden, die der zweier gewöhnlicher Einheiten entspricht.

Die praktischen Teile des Skripts verwenden zahlreiche Codebeispiele, die sich zwar an den funktionsfähigen Implementierungsbeispielen orientieren, diese aber teils vereinfachen, teils auch aus Sicht des Autors optimieren. Insbesondere die in diese Beispiele eingeflossenen Optimierungen wurden in den „echten“ Implementierungen bisher nicht durchgängig getestet und können daher z. T. auch noch syntaktische Defizite aufweisen.

Nur rudimentär vorhanden sind in der vorliegenden Version dieses Skripts Referenzen auf Publikationen und andere Verlautbarungen im Umfeld des für unser Thema relevanten Fachdiskurses, die die hier an verschiedenen Stellen durch den Autor getätigten Aussagen belegen oder differenzieren. In dieser Hinsicht ist das Skript nicht unbedingt als Anschauungsmaterial für eine Bachelorarbeit geeignet. Weitgehend verfügbar sind hingegen Referenzen auf API-Dokumentationen und anderes Material, das Sie für die individuelle Vertiefung des hier präsentierten Stoffes nutzen können.

4 Anwendungsfall

Die Veranstaltung verwendet einen durchgängigen Anwendungsfall. Dieser basiert auf einem Designkonzept, welches durch die Designerin [www Stefanie Klekamp](#) für ein App-Projekt entwickelt wurde, das in den zurückliegenden Semestern gemeinsam von Lehrenden und Studierenden der Humboldt-Universität zu Berlin und der Beuth Hochschule durchgeführt wurde.

Ausgangspunkt dafür war, dass das an der Humboldt-Universität angesiedelte [www Heiner-Müller-Archiv](#) vorhandenes Multimedia-Material – Bilder, Audio, Video – und Texte zu Leben und Werk des Schriftstellers [www Heiner Müller](#) (1929-1995) über eine mobile App einer breiteren Öffentlichkeit zugänglich machen möchte.

Folgende Abbildung zeigt Ihnen einige Ansichten, die im besagten Designkonzept für die unter dem Namen Müllers Welt entwickelte App vorgesehen sind.

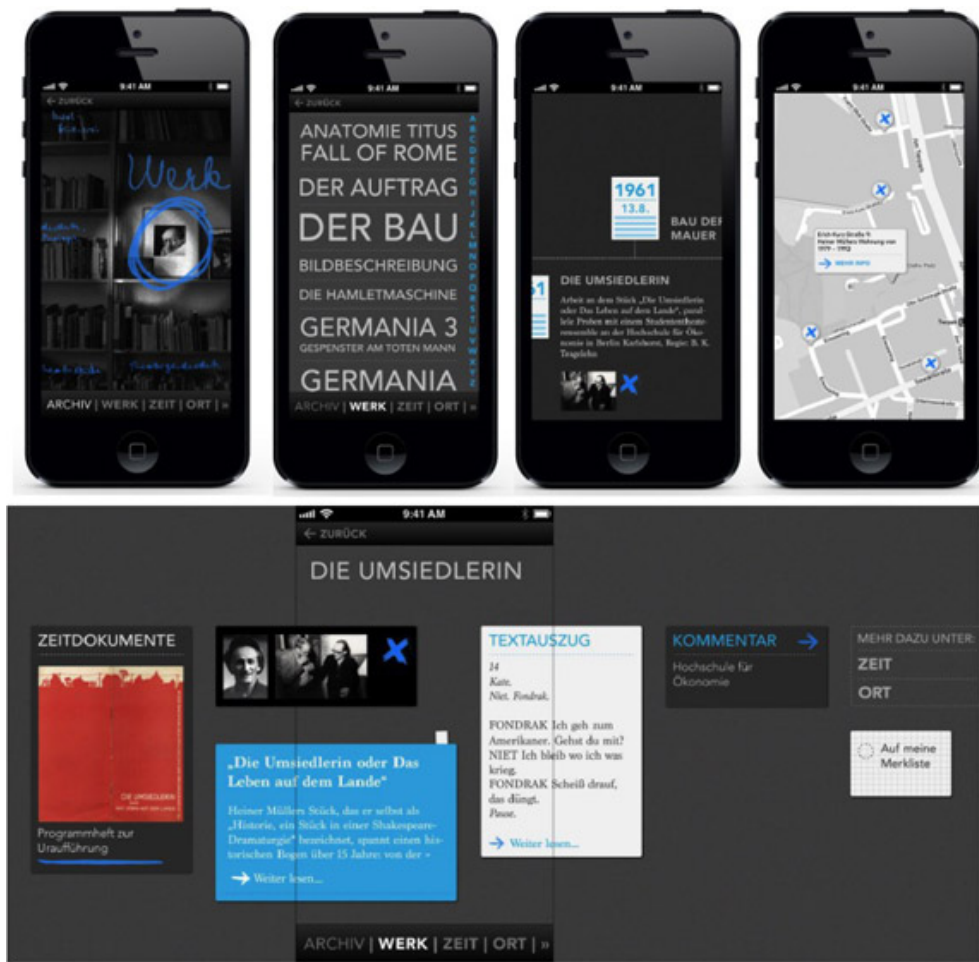


Abb.: Ansichten der mobilen Anwendung Müllers Welt

Ansichten der mobilen Anwendung Müllers Welt, die die verfügbaren Inhalte anhand der Dimensionen „Werk“, „Zeit“ und „Ort“ erschließen (oben). Oben links dargestellt ist die Startansicht der App. Unten sehen Sie die Gestaltung der Übersichtsseiten, die bei Auswahl eines Themas aus der Werksliste angezeigt wird. Diese Ansicht steht im Zentrum des Übungsprogramms unserer Veranstaltung

Schwerpunkt des Übungsprogramms bildet die Beschäftigung mit dem als „Übersichtsseite Werk“ bezeichneten Ansichtstyp und einer Auswahl der dafür vorgesehenen Gestaltungselemente. Diese werden Sie zunächst mit HTML und CSS modellieren und darstellen und dann mit JavaScript „dynamisieren“. Letzteres beinhaltet insbesondere auch den kompletten clientseitigen Aufbau dieser Ansichten auf Grundlage einer vom Server geladenen Konfiguration. Ab der Lerneinheit COH wird der Fokus dann auf einem ausgewählten Gestaltungselement namens „Objekt“ liegen, mit dessen Erstellung, Modifikation und Persistierung Sie sich in den darauf folgenden Lerneinheiten beschäftigen werden:

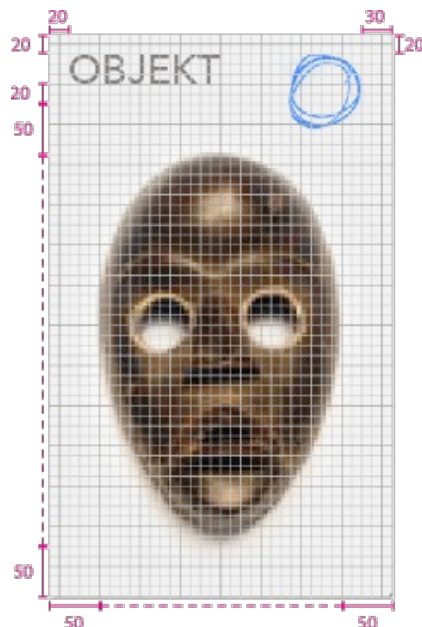


Abb.: Objekt aus Anwendungsbeispiel

Die Lerneinheiten zur lokalen Datenspeicherung in Lerneinheit LDS und zu Offline Anwendungen in Lerneinheit OFF werden in ihren Implementierungsbeispielen zusätzlich eine einfache Anwendung bereitstellen, die mit Listenansichten einen standardisierten Ansichtstyp verwendet, der in nativen mobilen Anwendungen sehr weit verbreitet ist. Beispielsichten hierfür finden Sie in folgender Abbildung.

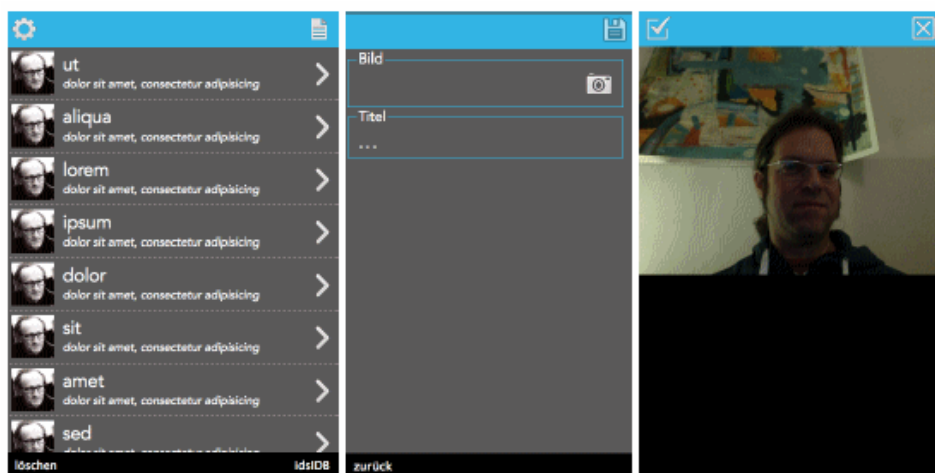


Abb.: Ansichten einer mobilen Webanwendung

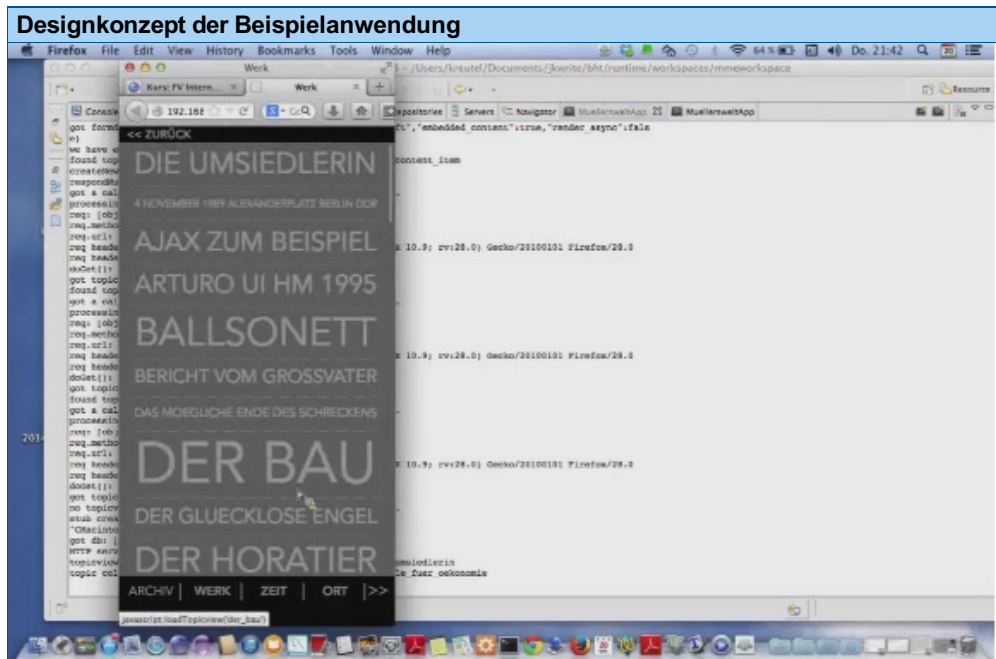
Ansichten einer mobilen Webanwendung, die sich an der Realisierung von Standard-Bedienelementen wie Listenansichten und Formularen in nativen Anwendungen orientiert.

Hier könne insbesondere die Umsetzung der HTML-Struktur und die darauf aufbauenden CSS-Regeln evtl. bereits zu einem früheren Zeitpunkt für Sie interessant sein. Fühlen Sie sich daher dazu eingeladen, das verfügbare Beispielmaterail nach Belieben zu sichten.

Im folgenden Film zeigt der Autor des Studienmoduls eine Beispielanwendung in der Teile des Designkonzepts umgesetzt wurden.



Film



© Beuth Hochschule Berlin - Dauer: 08:30 Min. - Streaming Media 19 MB

5 Begriffsklärung

Hier wollen wir eine kurze Klärung der drei wichtigsten Begriffe vornehmen, die wir in den vorstehenden Abschnitten bereits verwendet haben und auf die wir im Verlauf der Lehrveranstaltung sehr häufig zu sprechen kommen werden.

- Browser
- Webserver
- Webanwendung

Browser

Browser

Ein Browser oder auch Web-Browser ist eine Softwareanwendung, die als Client unter Verwendung des HTTP-Protokolls auf Inhalte, die über das Internet durch einen Webserver bereitgestellt werden, zugreifen und diese Inhalte entsprechend ihrem Inhaltstyp für einen Nutzer darstellen kann. Insbesondere stellt ein Browser eine Ausführungsumgebung für die Sprachen HTML, CSS und JavaScript entsprechend den dafür existierenden Spezifikationen bereit und ist dazu in der Lage, ein mittels dieser Sprachen beschriebenes Graphisches User Interface darzustellen und dessen Bedienung durch den Nutzer zu steuern.

In folgender Abbildung sehen Sie exemplarisch die Architektur des Firefox-Browsers dargestellt, deren Kern die Layout Engine Gecko ist.

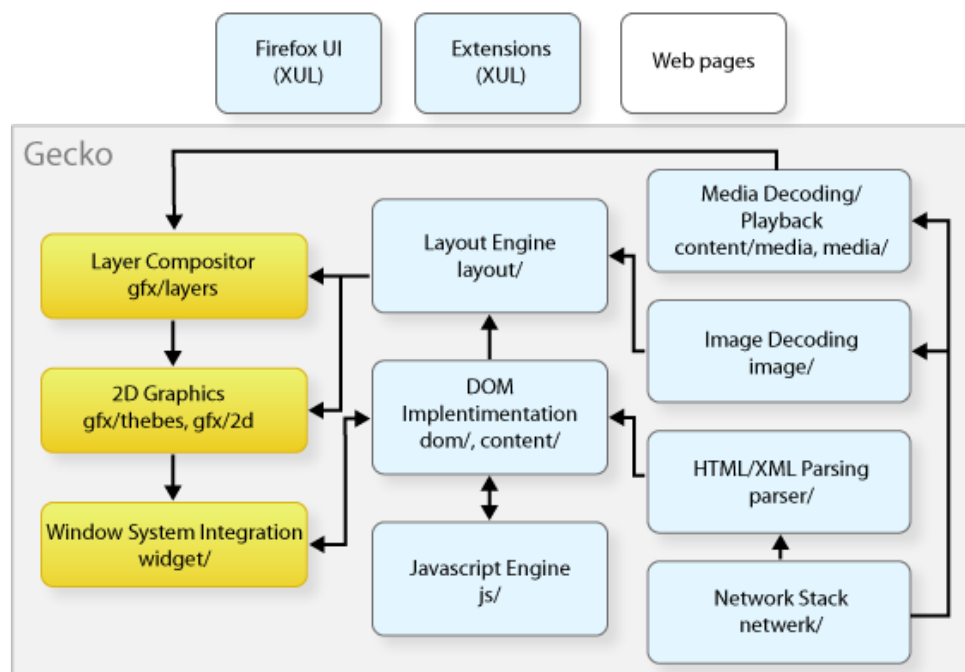


Abb.: Architektur des Firefox Browsers

Diese übernimmt sowohl den Aufbau und die Steuerung der Nutzeroberfläche des Browsers selbst und seiner Bedienelemente als auch die Darstellung der durch Firefox geladenen und durch CSS mit Stileigenschaften versehenen HTML-Dokumente. Dafür verwendet Gecko Komponenten die einzelne Teilfunktionen bereitstellen, z. B. für das Parsen von XML und HTML, die Repräsentation eines geladenen HTML-Dokuments als DOM-Objekt und den Zugriff auf dessen konstituierende Elemente, die Anwendung von CSS und die Interpretation von JavaScript sowie für das Abspielen von Multimedia. Nicht zuletzt wird auch die Funktion des Netzwerkzugriffs via HTTP durch eine Teilkomponente von Firefox erbracht.



Hauptkonkurrent von Gecko ist die www.w3.org Webkit Layout Engine, die u. a. in den Browsern Safari, Chrome und Opera, aber auch im Standardbrowser von Android verwendet wird.

Siehe für weiterführende Informationen zu Gecko sowie die www.mozilla.org Mozilla Developer Website.

Webserver

Ein Webserver ist eine Softwareanwendung, die dazu in der Lage ist, gemäß der Spezifikation des HTTP-Protokolls HTTP-Requests die von einem oder mehreren Clients übermittelt werden, entgegenzunehmen, gleichzeitig und unabhängig voneinander zu bearbeiten und jeweils mit einem protokollgemäßen HTTP-Response zu erwidern. Alternativ kann mit diesem Begriff auch die Hardware bezeichnet werden, auf der Webserver als Softwareanwendungen effizient ausgeführt werden können. Wenn im Rahmen dieses Skripts von einem „Server“ die Rede ist, ist damit immer ein Webserver im erstgenannten Sinne gemeint, es sei denn es wird ausdrücklich auf eine davon abweichende Bedeutung hingewiesen.

Das Funktionsspektrum von Webservern, d. h. die Art und Weise, wie sie HTTP-Requests verarbeiten und erwidern, ist sehr breit gefächert. Im einfachsten Fall besteht die Erwidern in der Übermittlung einer statischen Ressource, die durch die im Request angegebene URL identifiziert wurde. Dies entspricht einem „Fernzugriff“ auf statische Inhalte, die in einem Dateisystem vorliegen, durch den Client, der den Request initiiert.

Beispielsweise gibt der Webserver, den die von uns verwendete Entwicklungsumgebung  Aptana unmittelbar nach dem Starten bereitstellt, Zugriff auf die Projekte im Aptana Workspace. Der  Apache HTTP-Server als die am weitesten verbreitete Lösung dieser Art verfügt darüber hinaus über die Möglichkeit, via des Common Gateway Interface (CGI) native Anwendungen aufzurufen und z. B. die Ausführung von Perl oder PHP durch einen entsprechenden Interpreter zu initiieren.

Am anderen Ende des Funktionsspektrums kann ein Webserver die Bearbeitung eines Requests durch Verwendung einer mehrschichtigen Enterprise-Softwarearchitektur initiieren. In diesem Fall wird üblicherweise ein Application Server eingesetzt, der unter Verwendung eines Webserver im vorgenannten Sinn ein oder mehrere Anwendungsprogramme („applications“) bereitstellt, d. h. deren Funktionalität für einen Client via HTTP verfügbar macht.

Beachten Sie, dass die Begriffe „Webserver“ und „Web Application Server“ nicht als Synonyme behandelt werden sollten, da die hier beschriebene Funktionalität eines Application Servers deutlich über die zuvor genannte Basisfunktionalität eines Webserver wie Apache hinausgeht. Einem Application Server kann aber z. B. ein Apache Webserver vorgeschaltet sein, der die über ein öffentliches Netzwerk eintreffenden Requests intern an den Application Server weiterleitet.

Webanwendung

In einem weiten Sinne ist eine Webanwendung bzw. Web-Applikation eine Softwareanwendung, auf die ein Client via HTTP zugreifen kann. Diese Begriffsfassung verallgemeinert verschiedene Ausprägungen von Webanwendungen. Zum einen trifft sie keine Aussage bezüglich des Orts, wo die Anwendung ausgeführt wird, d. h. ob die Ausführung clientseitig oder serverseitig erfolgt. Zum anderen ist sie agnostisch gegenüber der Frage, ob die Funktionalität einer Webanwendung eine Nutzerschnittstelle beinhaltet oder ob die Anwendung eine reine Datenzugriffsschnittstelle implementiert und dem Client die Weiterverarbeitung der Daten inklusive derer Verwendung in den Ansichten eines UI überlässt. Ist letzteres der Fall, wird eine Webanwendung üblicherweise als Web Service bezeichnet.

Im Rahmen unserer Veranstaltung verstehen wir unter einer Webanwendung eine Softwareanwendung mit Nutzerschnittstelle, die auf Basis von HTML, CSS und JavaScript implementiert wird, durch einen Webserver bereitgestellt wird und in einem Browser als Client zur Ausführung gebracht wird.

Zusammenfassung

- Unter Webanwendungen - die auf mobilen Endgeräten lauffähig sind - verstehen wir Anwendungen, die auf Basis der standardisierten Webtechnologien HTML, CSS und JavaScript entwickelt und durch einen auf einem Endgerät installierten Browser ausgeführt werden.
- Ein wesentliches Kriterium zur Unterscheidung von Fat Client im Ggs. zu einer Thin Client Architektur stellt bei Webanwendungen die Frage dar, inwieweit durch den Server dynamische Bestandteile der durch den Client realisierten Nutzeroberfläche in Form von HTML-Markup bereitgestellt werden, oder ob der Server lediglich strukturierte Daten in XML oder JSON sowie Medienressourcen liefert.
- Ein Browser oder auch Web-Browser ist eine Softwareanwendung, die als Client unter Verwendung des HTTP-Protokolls auf Inhalte, die über das Internet durch einen Webserver bereitgestellt werden, zugreifen und diese Inhalte entsprechend ihrem Inhaltstyp für einen Nutzer darstellen kann.
- Ein Webserver ist eine Softwareanwendung, die dazu in der Lage ist, gemäß der Spezifikation des HTTP-Protokolls HTTP-Requests die von einem oder mehreren Clients übermittelt werden, entgegenzunehmen, gleichzeitig und unabhängig voneinander zu bearbeiten und jeweils mit einem protokollgemäßen HTTP-Response zu erwidern.
- „Webserver“ und „Web Application Server“ sollten nicht als Synonyme behandelt werden da die Funktionalität eines Application Servers deutlich über die Basisfunktionalität eines Webservers hinausgeht.

Sie sind am Ende dieser Lerneinheit angelangt.