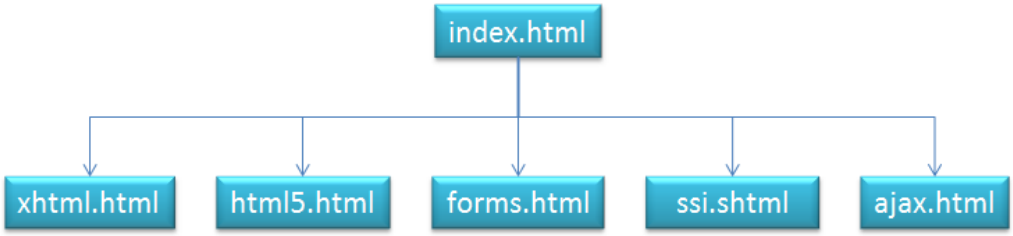
Web-Anwendungen (WA) Laboraufgabe WS 2018/2019

Dipl. Ing. Stefan Koß – FH-Kiel – stefan.koss@fh-kiel.de
Dipl. Inf. Florian Caspar – FH-Kiel – florian.caspar@fh-kiel.de

Zielsetzung

Ziel ist es, eine vollständige Webpräsenz zu einem beliebigen Thema zu erstellen, wobei möglichst viele Aspekte der modernen Webprogrammierung angewendet und geübt werden. Der Fokus liegt auf dem Erstellen der HTML- und CSS-Dokumente für ein Web-Frontend. Es sollen Formulare erstellt und Server Anfragen mit den HTTP Methoden GET und POST geübt werden. Außerdem sollen Webseiten erstellt werden, in denen mit JavaScript dynamisch Objekte im DOM modifiziert oder erstellt werden. JavaScript soll auch dazu verwendet werden, Serveranfragen mittels AJAX zu generieren.
Optional können motivierte Studentinnen und Studenten auch Bibliotheken verwenden (bspw. jQuery) und/oder tiefer in die Internetserver-Programmierung einsteigen (bspw. Apache, PHP, Tomcat, Node.js).
Die Grundlegende Struktur der Webpräsenz *kann* bspw. wie in folgender Grafik angelegt werden. Jede einzelne HTML-Datei *kann* dabei einzelne im Labor ausdrücklich geforderte Merkmale enthalten. Sie können aber auch selbst eine Struktur bestimmen und zusätzliche Elemente einbringen, solange Ihre Präsenz den Minimal-anforderungen der einzelnen Laborelemente genügt. Beachten Sie, dass die Startseite Ihrer Präsenz immer als „index.html“ benannt werden muss, damit Sie beim Aufruf der Webadresse automatisch vom Server abgerufen wird.



Zeitlicher und organisatorischer Ablauf

Das Labor besteht aus 3 verschiedenen Aufgabenteilen, und es gibt insgesamt 5-6 Termine für jede Gruppe. Die Laboraufgabe kann in Teams mit maximal 2 Personen gelöst werden. Beide Teammitglieder müssen bei der Abnahme in der Lage sein, zu beschreiben, was beide Teammitglieder gemacht haben. D.h. Mitglied A hat bspw. JavaScript programmiert und sowohl Mitglied A als auch B müssen dieses erklären können.

Abnahmen sind jeweils zum 2., 4. und 5. bzw. 6. Labortermin fällig, d.h. Aufgabenteil 1 zum 2. Termin, Teil 2 zum 4. Termin, Teil 3 zum 5. bzw. 6. Termin!

Zu den Terminen müssen jeweils die Punkte in der Tabelle zu den jeweiligen Aufgabenabschnitten implementiert, vorgeführt und vom Laborpersonal abgenommen werden.
Aufgrund der Verschachtelung der Vorlesungen mit den Laborgruppen wird es mit Sicherheit zu der Situation kommen, dass Laboraufgaben Stoff verwenden, welcher in der Vorlesungen noch nicht vollständig behandelt wurde. In diesem Regelfall wird von der Studentin / dem Studenten erwartet, sich das fehlende Wissen aus dem Web, Vorlesungsskript oder einschlägigen Büchern selbst anzueignen. Dieser Aufwand fällt unter die Kategorie Selbststudium, wie in der Modulbeschreibung entsprechend vermerkt ist. Des Weiteren können auch Aufgrund der Vielfältigkeit von HTML und CSS nicht alle Tags und deren Attribute vollständig in der Vorlesung abgedeckt werden. In diesem Fall gilt, wie vorher, das Selbststudium. Generell sind die Laboraufgaben so gestellt, dass die reine Laborzeit nicht dazu ausreicht, sie zu lösen. Der Extraaufwand fällt wieder unter die Kategorie des Selbststudiums.

Notwendige Software

Zum Durchführen des Labors sollte folgende Software auf dem eigenen Rechner installiert werden:

- Texteditor bspw.:
  - Webstorm (wir haben eine Lizenz und es gibt ein „live-debugging“ Plugin für Chrome) http://www.jetbrains.com/webstorm/
  - Eclipse http://www.eclipse.org/
  - Notepad++ http://notepad-plus-plus.org/
- Webbrowser (wenigstens 2):
  - Chrome https://www.google.com/intl/en/chrome/browser/
  - Firefox http://www.mozilla.org/en-US/firefox/new/
    - Zusätzlich das Firefox Debugging Plugin http://getfirebug.com/
  - Internet Explorer (bei Windows dabei)
  - Safari (bei OSX dabei)
  - Opera http://www.opera.com/de

Außerdem wird ein Webserversystem benötigt. Sehr gut nutzen lässt sich dazu Apache and Friends „Xampp“:

- Webserver Apache bspw. im Paket mit Xampp (http://www.apachefriends.org/de/xampp.html)

Möchten Sie gerne einen Laborrechner verwenden, sollten Sie sich die entsprechenden Tools auf einem USB-Speicher-Stick vorinstallieren. Auf dem Stick sollten Sie dann auch Ihre selbst erstellten Dateien entsprechend verwalten. Xampp lässt sich auf auch auf einem USB-Stick installieren. Hier gibt es verschiedene „lite“ und „portable“ Versionen: https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/1.8.1/ . Aktuelle Anleitungen lassen sich im Internet per Suche problemlos finden.

Ihre Webpräzenz lässt sich nach dem Starten des Servers (bspw. Xampp Apache) typischerweise unter http://localhost abrufen. Falls Xampp genutzt wird, befinden sich alle HTML Dateien im Verzeichnis …\xampp\htdocs . Erstellen sie ein Unterverzeichnis bspw. WA, in welchem sie alle ihre Dateien ablegen. Diese können dann über http://localhost/WA/Dateiname.html nach dem Starten von Apache über das Xampp Control Panel aufgerufen werden.

Laborberichte

Laborberichte müssen für WA nicht angefertigt werden. Es genügen die Abnahmen zu den entsprechenden Terminen und das Demonstrieren der fertigen Webpräsenz auf den Server. Der HTML, CSS und JavaScript Code muss hinreichend dokumentiert sein!

Anwesenheitspflicht

Zu den Laboren herrscht Anwesenheitspflicht, die zu Beginn des Labors vom Laborpersonal abgefragt wird. Eine valide Entschuldigung bei Fernbleiben ist vorzulegen.

Abkürzungen

AJAX – Asynchronous JavaScript and XML
API – Application Programming Interface
CGI – Common Gateway Interface
CSS – Cascading Style Sheets
DOM – Document Object Model
HTML – Hypertext Markup Language
HTTP – Hypertext Transfer Protocol
JS – JavaScript
SSI – Server Side Include
XHTML – Extensible Hypertext Markup Language

Cheat Sheets

- HTML5: http://www.smashingmagazine.com/2009/07/06/html-5-cheat-sheet-pdf/
- CSS: http://coding.smashingmagazine.com/2009/07/13/css-3-cheat-sheet-pdf/
- JS: http://www.addedbytes.com/cheat-sheets/javascript-cheat-sheet/
- HTML5 + CSS: http://webdesignledger.com/tutorials/15-useful-html5-tutorials-and-cheat-sheets

Updates / Version

Dieses Dokument mag natürlicherweise Fehler enthalten. Sollte ihnen etwas merkwürdig vorkommen, kontaktieren sie bitte das Laborpersonal. Wir bemühen uns um entsprechende Anpassungen.

Die aktuelle Version ist vom 21.09.2018 22:19:21.

Aufgabenteil 1: Grundlagen der Entwicklung von XHTML-Dokumenten

Ziele:
- Einführung in die grundlegende Entwicklung von XHTML-Dokumenten.
- Start mit der Webpräsenz, Erzeugen einer Startseite „index.html“ sowie ersten Unterseiten.
- Erstellen eines „XHTML 1.0 Strict“ konformen Webdokumentes und dessen Validierung mittels Validators.
- Grundlegendes Seitenlayout und Design mittels CSS.

Hinweise:
- Erzeugen Sie eine sinnvolle Verzeichnisstruktur, in der Sie bspw. Bilddateien, CSS Dateien usw. jeweils in einem eigenen Unterverzeichnis ablegen (z.B. Wurzel „/“ mit HTML Dateien, dann „/img“ für Bilddateien, dann „/css“ für Stylesheets, „/cgi-bin“ für Server-Skripte, „/js“ für JS usw.).
- Sie können mit Webstorm in Kombination mit Chrome und dem JetBrains Plugin Ihre Seitenänderungen in Echtzeit im Browser verfolgen. Außerdem bietet Webstorm Ihnen eine automatische Validierung der Syntax (jedoch nicht der XHTML-Konformität).

Aufgabenstellung:
Folgende Eigenschaften sollen in wenigstens einer XHTML-Datei und CSS realisiert werden und müssen zum 2. Labortermin vom Laborpersonal abgenommen werden:

Meilenstein	Seiteneigenschaft	✓
1	Webseite im XHTML 1.0 Strict Doctype und UTF-8 Zeichenencodierung mit CSS Layout und Realisierung grundsätzlicher Seitenstrukturen mit DIV-Tags (Titel, Menü, Überschrift, Paragraphen, Fußnote mit Autorenadresse und Email-Link usw.).	
2	Dabei Einbindung eines CSS aus einer separaten Datei.	
3	Positionierung und Formatierung der Seitenelemente ausschließlich mit CSS und DIV-Tags, bzw. CSS Klassen und/oder IDs.	
4	Fließtexte ausschließlich mit semantischen HTML Tags (bspw. <p>, <h1-6>, , usw.).	
5	Links, Navigation auf weitere interne und externe Seiten.	
6	Verschiedene Farben und Schriftarten (Text, Links, Hintergrund) über entsprechende Definitionen im CSS.	
7	Listen (Sortierte, Unsortierte), Formatierung ausschließlich mit CSS.	
8	Tabellen (einfache und geschachtelte, Tabelle in Tabelle, unterschiedliche Zellenrahmen und Farben mittels CSS).	
9	Grafiken (einfach, mit Navigation und Links, gif, jpeg, png).	
10	Definition und Einbindung eines Druckformates über CSS.	
11	Neudefinition des Styles der Standardtags (bspw. body, h1, p usw.) im CSS.	
12	Bildpositionierung wenigstens zweier Bilder übereinander (z-index) mit CSS.	
13	Metatags im Header (Author, Charset).	
14	Validierung des XHTML-Dokumentes mit dem Validator unter http://validator.w3.org/ .	
15	Aktivierung des eigenen Webservers (bspw. Xampp Apache) mit eigenen Dateien.	

Aufgabenteil 2: Entwicklung von HTML5-Dokumenten, Formularen und Serveranfragen

Ziele:
- Erweiterung der Webpräsenz mit Unterseiten.
- Erstellen von HTML5 Webdokumenten unter Benutzung von HTML5 Semantik.
- Einbinden von weiteren Medienformaten (Vektorbilder, animierte Bilder, Videos, …).
- Einfache Formulare und klassische HTTP Serverkommunikation über CGI.
- Dokumente mit SSI.

Hinweise:
- Sie können ein Echo Skript „echo.pl“ im LMS herunterladen und es unter xampp\cgi-bin speichern. Mit der URL http://localhost/cgi-bin/echo.pl werden dann alle POST Anfragen in Textform ausgegeben. Sie können auch gerne selbst ein Serverskript erstellen (bspw. in Perl, PHP, Python, JS usw.) bzw. das o.g. erweitern.
- Sie können eigene Skripte unter dem „cgi-bin“ Verzeichnis auf dem Server verwenden, oder in Ihrer eigenen Apache Installation (Xampp „htdocs“ Verzeichnis. Gleiches gilt für die SSI HTML Dateien.
- Für SSI müssen Sie Ihre HTML-Dateien mit der Endung „.shtml“ auf dem Server ablegen, damit sie automatisch ausgewertet werden.
- Falls sie Probleme beim Ausführen der CGI-Skripte haben, überprüfen sie, ob der Interpreterpfad in der Skriptdatei richtig gesetzt ist.

Aufgabenstellung:
Folgende Eigenschaften sollen in HTML5-Dateien und CSS realisiert werden und müssen zum 4. Labortermin vom Laborpersonal abgenommen werden:

Meilenstein	Seiteneigenschaft	✓
1	Webseite in HTML5 Doctype und UTF-8 Zeichencodierung und CSS Layout Datei, sowie Meta Tags im Header (Author, Charset).	
2	Realisierung grundsätzlicher Seitenstrukturen (Titel, Menü, Überschrift, Paragraphen, Fußnote mit Autorenadresse usw.) mit HTML5 Semantik (d.h. Verwendung von den Tags „header“, „nav“, „section“, „article“, „aside“ usw. anstelle von DIV-Tags).	
3	Es soll zum Design der Webseite ausschließlich CSS verwendet werden (wie im 1. Versuchsteil).	
4	Einbindung von skalierbaren Vektorgrafiken (.svg Dateien).	
5	Einbindung von animierten GIF-Dateien.	
6	Einbindung von Videodateien (mp4, WebM oder Ogg über das „video“ Tag).	
7	Definition eines Formulars mit den Eingabefeldtypen „text“, „password“, „radio“, „email“, „date“, „submit“, „reset“, wobei entsprechend beschreibende Labels und das Attribut „required“ auftauchen sollen. Submit Button mit Grafikhintergrund.	
8	Senden der Formulardaten mit der GET und/oder POST Methode an ein CGI-Skript, welches diese auswertet und eine entsprechende Antwort sendet.	
9	Eine weiteres HTML Dokument mit beliebigem Doctype mit SSI Tags.	
10	Format Konfiguration im SSI Dokument (bspw. Datum unterschiedlich darstellen).	
11	Echo Befehl im SSI Dokument.	
12	Dateigröße und Datum der letzten Änderung im SSI Dokument.	
13	Weitere Datei vom Server einbinden im SSI Dokument.	
14	Kommentare im HTML zur Beschreibung der Webseite.	

Aufgabenteil 3: Einbinden von JS, dynamische Webseiten, AJAX-Serveranfragen

Ziele:
- Erweiterung der Webpräsenz um dynamische Elemente.
- Einstieg in die Browserprogrammierung mit JavaScript und erweiterte HTML5 Elemente.
- Einstieg in die Browser API mit Zugriff auf DOM-Elemente mit JS und Debuggen von JS.
- Persistierung von Daten im Browser (Cookies).

Hinweise:
- Für die Versuchsdurchführung ist eine sinnvolle Debugging-Umgebung notwendig. Sie können bspw. die Chrome Entwicklertools nutzen (standardmäßig bei Chrome dabei) oder das Firebug Plugin für Firefox.
- Um Cookies im Browser zu setzen, ist es notwendig, die Webseite von einem Server abzurufen. Es genügt nicht, einfach die Datei von der Festplatte im Browser zu öffnen, weil der Browser direkt geöffneten Dateien nicht erlaubt, Cookies abzulegen: Cookie Zugriff ist auf Adressen einer Domäne beschränkt.
- Beim Programmieren der AJAX Anfrage müssen abhängig von dem Browser unterschiedliche API Methoden aufgerufen werden.

Aufgabenstellung:
Folgende Eigenschaften sollen in HTML5-Dateien realisiert werden und müssen zum 5. bzw. 6. Labortermin vom Laborpersonal abgenommen werden:

Meilenstein	Seiteneigenschaft	✓
1	Einbinden einer externen JS Datei in das HTML5 Dokument, die alle Skriptteile der folgenden Aufgaben beinhaltet.	
2	Ausgabe des Browsertyps in einem dynamisch erzeugten DIV-Tag mit JS. Benutzen Sie hierzu das „body“ „onload“ Ereignis.	
3	Speichern und Auslesen eines Cookies mit JS (bspw. das letzte Besuchsdatum). Kann bspw. in derselben Funktion geschehen, wie Meilenstein 2.	
4	Timer-gesteuertes Zeichnen (Animation) in einem HTML5 Canvas Element. Benutzen Sie hierzu die „setInterval()“ Methode und schreiben Sie eine entsprechende Funktion.	
5	Senden einer AJAX-Anfrage an den Server. Hier können Sie das in Aufgabe 2 erstellte Formular wiederverwenden. Bei Mausklick auf den „Submit-Button“ soll ein entsprechendes JS-Ereignis abgefangen werden, welches die AJAX-Anfrage startet. Erstellen sie für die AJAX-Anfrage ein JavaScript Objekt, welches alle Daten der Anfrage als Schlüssel-Wertpaare enthält.	
6	Darstellen der Serverantwort auf die AJAX Anfrage in einem mit JS dynamisch verändertem DIV-Tag mittels der Callback Funktion der AJAX-Anfrage.	
7	Auslesen und Modifizieren von Formularwerten mit JS. Fügen Sie hierzu bspw. einen Button ein, der beim Drücken alle Formularfelder auf Initialwerte setzt.	
8	Ausgeben von Debugging-Nachrichten und Werten auf die JS-Konsole. Geben Sie bspw. Nachrichten jeweils bei Funktionsaufrufen aus.	
9	Setzen von Breakpoints und Beobachten von Variablen in einer JS-Browser-Debugging-Umgebung.	