EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Mathematisch-Naturwissenschaftliche Fakultät
Fachbereich Informatik
Informationsdienste – Prof. Dr. Thomas Walter

Einführung Internet-Technologien

Sommersemester 2020

Übungsblatt 1

Theorie

Die Lösungen zu den Theorieaufgaben werden in den Tutorien besprochen und im Moodle veröffentlicht. Sie brauchen *keine* Lösungen zu den Theorieaufgaben abgeben!

1. Erklären Sie die Idee von Auszeichnungssprachen sowie nötige Konzepte.

Lösung: Die grundlegende Idee hinter Auszeichnungssprachen ist bereits seit dem 15. Jahrhundert bekannt. Zu damaliger Zeit mussten Drucksetzer noch das Druckbild in mühseliger Kleinarbeit aus Teilkomponenten zusammensetzen. Damit der Drucksetzer bestimmte Inhalte mit gewünschten Druckschablonen setzte, wurden ihm entsprechende handschriftliche Markierungen angegeben; beispielsweise sollte ein Teil als kaligrafische Überschrift gesetzt werden, während der Text Druckbuchstaben-Bausteine verwenden sollte. Diese wurden jedoch deutlich vom Inhalt abgegrentzt; beispielsweise über eine Farbe oder symbolische Untermalung.

Diese Idee wurde im Rahmen der Digitalisierung immer relevanter: Einerseits mussten Informationen zwischen digital arbeitetenden Systemen ausgetauscht werden, andererseits war eine Verbindung von diese Inhalte beschreibenden Zusatzinformationen (Metainformationen) gewünscht. Auf diese Weise konnten Systeme die Inhalte eigenverantwortlich (weiter-)verarbeiten und bei Bedarf auf die Metainformationen zurückgreifen. Ist beispielsweise die Metainformation „Überschrift Typ 1“ vom Inhalt „Übungsserie 01“ getrennt, so kann man einen einfachen Texteditor nutzen, der keine Überschriften und Inhaltsunterscheidung kennt. Dieser ignoriert die Metainformation einfach und hätte dennoch den Inhalt „Übungsserie 01“, den er darstellen kann. Nutzt man einen komplexeren Texteditor, der Überschriften kennt, so kann dieser auf Basis der Metainformation „Überschrift Typ 1“ den Inhalt „Übungsserie 01“ entsprechend stylen.

Dementsprechend ist die Idee hinter Auszeichnungssprachen die *Trennung von Inhalt und Metainformationen*. Hierfür muss eine eindeutig erkennbare Trennungsvorschrift existieren. Im Umfeld der digitalen Datenverarbeitung ist zudem sicherzustellen, dass Inhalte sowie Metainformationen zudem menschenlesbar als auch maschinenlesbar sind. Sprich: Ein Mensch muss erkennen was sind die Inhalte und was die Beschreibung und gleichzeitig muss automatisiert auf die Inhalte /

Metainformationen zugegriffen werden können, damit diese automatisiert weiterverarbeitet werden können.

2. Was versteht man unter einer Dokumenttypdefinition (DTD)?

Lösung: Eine Dokumenttypdefinition (DTD) legt die Sytnax einer Auszeichnungssprache fest; d.h. welche Elemente existieren, wie sind diese aufgebaut und wie können diese miteinander in Bezug gesetzt werden. Somit können Dokumente anhand der DTD automatisiert auf syntaktische Korrektheit überprüft werden. Jedoch ist eine semantische Überprüfung über die DTD nicht möglich. Hierfür sind – wenn überhaupt möglich – zusätzliche Beschreibungssprachen nötig.

3. Erklären Sie den Anwendungszweck der Auszeichnungssprache HTML.

Lösung: HTML ist als Auszeichnungssprache für Inhalte im World Wide Web konzipiert. Sie trennt die Struktur der Inhalte von den eigentlichen Inhalten. In den ersten Versionen wurde teilweise noch Layout-Auszeichnung über HTML geregelt (z.B. das <b>-Tag in HTML1.0). In den neueren Versionen von HTML wurde jedoch weitestgehend auf Layout-Auszeichnung zugunsten von CSS verzichtet.

Über HTML existiert eine standardisierte Beschreibungssprache, auf deren Basis Render-Engines Inhalte von unbekannten Servern dennoch korrekt strukturiert aufbereiten können. Wichtigste Komponente von HTML ist dabei der Hyperlink (a-Tag) über welchen in HTML-Dokumenten auf andere (HTML-)Dokumente, die auch auf komplett anderen Servern liegen können, verwiesen werden kann. Somit lässt sich automatisiert durch die Dokumente nach Bedarf navigieren.

4. Was ist SGML? Wie stehen SGML und HTML miteinander in Bezug?

Lösung: Die Standard Generalized Markup Language (SGML) wurde 1986 im ISO8879-Standard veröffentlicht. Die SGML ist als standardisierte Sprache zum erstellen von Auszeichnungssprachen konzipiert.

HTML und XML sind die wohl bekanntesten Auszeichnungssprachen, die auf Basis der SGML erstellt wurden.

5. Erstellen Sie ein Minimalbeispiel für eine „HTML5“-validen Textauszeichnung.

Lösung:

```
<!DOCTYPE html>
<title>Minimalistic</title>
<h1>Heyho - Let's go!</h1>
```

Man beachte: Die Tags html, head sowie body können in HTML5 weg gelassen werden. Diese waren im ersten HTML-Standard ebenfalls nicht definiert, weshalb die meisten Render-Engines aus Abwärtskompatibilitätsgründen bereits mit solchem Markup umgehen können. Zum besseren Verständnis - insbesondere wenn man noch keine Erfahrung mit HTML hat - bietet sich jedoch zunächst die genaue

Untergliederung der Seiten-Beschreibenden Informationen sowie der Seiteninhalte an:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Minimalistic</title>
 </head>
 <body>
  <h1>Heyho - Let's go!</h1>
 </body>
</html>
```

6. Geben Sie Beispiele für HTML-Metainformationen, inklusive deren Verwendung an.

Lösung:

- ZeichenCodierung des Dokuments auf UTF-8 setzen:

  `<meta charset="UTF-8">`

- Schlüsselwörter für Suchmaschinen-Agenten setzen:

  `<meta name="keywords" content="HTML, CSS, JavaScript, Python, PHP">`

- Beschreibung der Webseiteninhalte:

  `<meta name="description" content="Intec Tutorials are great!">`

- Autor der Webseite angeben:

  `<meta name="author" content="Hans Mole">`

7. Erklären Sie, weshalb dieselbe HTML-Auszeichnung in verschiedenen Browsern teils unterschiedlich angezeigt wird.

Lösung: Das Problem an der unterschiedlichen Darstellung ist die unterschiedliche Umsetzung des HTML-Standards in den verschiedenen Render-Engines der Browser. Diese setzen zwar die Regeln korrekt um – beispielsweise ein h1-Tag entspricht der größten Überschrift – jedoch ist nirgendwo definiert, wie die Regeln zu visualisieren sind. Dementsprechend können Browserhersteller selbst definieren, wie die Darstellung der Tags erfolgt; beispielsweise wie breit ein Tabellenrahmen ist oder die Stärke von margin und padding eines Elements. Der Cascading-Stylesheets-Ansatz (siehe nächste Übungsserie) kann verwendet werden um für Browser die Darstellung vorzudefinieren. Hier sei jedoch schon darauf verwiesen, dass dies teilweise aufgrund fehlender Umsetzung in den Render-Engines dennoch zu Problemen in der Praxis führen kann.

8. Weshalb werden Umlaute in der HTML-Auszeichnung beim Rendering kryptisch dargestellt?

Lösung: Dies liegt meistens daran, dass der Browser die HTML-Auszeichnung in einer anderen Codierung interpretiert, als diese gespeichert wurde.

Beispiel: Man hat ein HTML4-Dokument mit UTF-8-Codierung erstellt. Der Browser versucht jedoch, die Zeichen in ISO-8859-1-Codierung zu interpretieren:

	UTF-8	**ISO-8859-1**
Bytes pro Zeichen:	1 – 4 Bytes	1 Byte
Bytes:	`c3 a4`	`c3 a4`
Darstellung:	ä	Ã ¤

In diesem Fall würde also statt eines „ä“ die Buchstabenkombination Ã ¤ angezeigt.

Bei HTML4 war die Codierung ISO-8859-1 Standard. Bei HTML5 ist es UTF-8, wodurch es zumindest kaum noch zu Problemen kommt.

Ein weiterer Grund für die kryptische Darstellung ist das Fehlen eines Symbols in der Zeichencodierung. Beispielsweise ist das Eurozeichen „€“ nicht in ISO-8859-1 enthalten. Hierfür gibt es die Möglichkeit die Zeichen direkt über HTML-Unicode-Auszeichnung einzufügen. Dies wird über die Spezielle Zeichenkombination &Unicode-Codierung; ermöglicht. Die Unicode-Codierung kann hierbei entweder als HTML-Name, Hexadezimal oder Dezimal angegeben werden. Für das Eurozeichen beispielsweise `&euro;` (HTML-Name), `&#x20AC;` (Hexadezimal) oder `&#8364;` (dezimal). In UTF-8 ist das Eurozeichen jedoch auch kein Problem mehr.

9. Welche Probleme müssen Sie hinsichtlich asiatischen und arabischen Schriften berücksichtigen?

Lösung: Asiatische und arabische Schriften haben zusätzlich zur Codierungsproblematik das Problem, dass sie teilweise nicht von links nach rechts sonder von rechts nach links bzw. von oben nach unten geschrieben werden. Dies muss zunächst von den Render-Engines berücksichtigt und im HTML korrekt angegeben werden. Weiterhin basieren diese Schriften häufig auf sehr detailreichen (asiatische) und miteinander stark verwobenen (arabische) Schriftzeichen. Daher sind die benötigten Fonts in der Regel sehr groß und in den meisten (westlichen) Browsern nicht per Default installiert. Dementsprechend gestaltet sich die - doch stark nach westlichem Vorbild geprägte - Webseiten-Erstellung für asiatische und arabische Webseiten deutlich komplexer.

10. Ist der HTML5-Standard als final anzusehen?

Lösung: Das kommt auf die Definition von final an. Der HTML5-Standard ist als offener Standard konzipiert. Dies bedeutet, dass er modular aufgebaut ist und bei Bedarf um neue Module erweitert werden bzw. um irrelevant gewordene Module

verkleinert werden kann. Dies bedeutet einerseits, dass er inhaltlich flexibel und verändert werden kann, die Render-Engines jedoch abwärtskompatibel veraltete Seiten, die auf dem HTML5-Standard erstellt wurden noch darstellen können.

11. Welche Neuerungen bringt der HTML5-Standard gegenüber Vorgängerversionen?

Lösung: Die wohl offensichtlichste Änderung am HTML5-Standard zu seinen Vorgang ist die einfache und deutlich kürzere Dokumenttypdefinition <!Doctype html>. Dahinter versteckt sich dann, wie bereits mehrfach beschrieben, technisch ein rießiges Feld von Neuerungen durch den modularisierte Aufbau des Standards und die daraus resultierenden Anforderungen an Render-Maschinen und Browser.

Im Bereich der Seitenstruktur bietet HTML5 semantische Divs, welche es ermöglicht seine Inhalte Suchmaschinen-Optimiert auszuzeichnen; beispielsweise Header, Footer, Article und Aside. Die bisher existierenden semantischen Tags wie beispielsweise die Überschriften wurden ohne Änderungen übernommen.

HTML5 bietet auch neue Formular Eingabe-Elemente, welche deutlich spezifischer zugeschnitten sind; beispielsweise telephone, url und email, search, number oder range. Dadurch können direkt im Browser - ohne JavaScript zu nutzen - Eingaben auf Korrektheit überprüft werden und die Bandbreite im Internet wird geschohnt.

Über das neue HTML5-Canvas Element bietet der Standard eine performante und simple Möglichkeit in Kombination mit JavaScript animierte Grafiken zu verwenden. Dies ist insbesondere dann relevant, wenn Nutzer auf externe Plugins verzichten möchten; beispielsweise Adobe Flash oder Microsoft Silverlight. Typische Anwendungsfelder sind Browser-basierte Spiele sowie (logischerweise) Animationen.

Zudem erleichter das HTML5-Video / Audio-Tag es, Videos und Audio-Dateien auf seinen Webseiten einzubinden. Problematisch hierbei ist aktuell noch, dass sich die Standard-Entwickler/Browser-Hersteller noch auf keinen Dateistandard geeinigt haben. Somit kann es passieren, dass ein Video in einem Brower korrekt abgespielt wird, in einem anderen jedoch nicht.

Für weiterführende Informationen sei an dieser Stelle auf `https://www.w3schools.com/html/html5_new_elements.asp` sowie `http://www.selfhtml5.org/html5-features/`.

12. Was ist ein HTML-Formular? Welche Eingabe-Felder bietet der HTML5-Standard?

Lösung: Ein HTML-Formular wird über das semantische form-Tag angegeben. Dieses zeigt dem Browser an, dass die enthaltenen Kindelemente eine Eingabeelemente für den Nutzer enthalten. Dies kann zum Einen zu semantischen Zwecken in der client-Seitigen Programmierung Verwendung finden; beispielsweise wenn ein JavaScript-Programm über den forms-Selektor auf Eingabeelemente zugreifen soll. Zum Anderen sind HTML-Formulare nötig um Nutzereingaben über HTTP vom Client an den Server zu schicken (siehe spätere Übungsserie). Wichtig ist hierbei, dass auf einer Seite beliebig viele Formulare eingebettet sein können.

HTML-Formulare haben typischer Weise folgenden Aufbau:

```
<form action="skript" onsubmit="javascript skript" method="methode">
 <label for="inputname">Bezeichner</label><input type="typ" id="
    inputname" />
 <label for="inputname">Bezeichner</label><input type="typ" id="
    inputname" />
 ...
 <input type="submit" Value="Abschicken" />
</form>
```

Für weitere Informationen sei auf `https://www.w3schools.com/tags/att_input_type.asp` verwiesen.

13. Erklären Sie den Anwendungszweck der Cascading Style Sheets (CSS) Technologie.

Lösung: Über Cascading Style Sheets (CSS) kann das Layout einer Webseite unabhängig vom Inhalt dieser definiert werden. Um möglichst viel Layout an zentraler Stelle definieren zu können, ist es möglich eine separate CSS-Datei zu erstellen und in die HTML-Beschreibungen einzubinden. Neuere CSS-Versionen (CSS3) gehen dabei deutlich weiter als reines Layout bereitzustellen: Sie ermöglichen es teilweise dynamisches Verhalten ohne Programmiersprache auf seine Webseiten zu integrieren; beispielsweise Animationen oder Inhaltsanpassung in Abhängigkeit von Geräte-Eigenschaften.

14. Nennen Sie die Vorteile der CSS-Nutzung.

Lösung: Der Hauptvorteil ist die Zentralisierung der Layout-Beschreibung. Anstatt bei Anpassungen mehrere Webseiten-Beschreibungen ändern zu müssen, kann dies zentral erfolgen. Dies resultiert in einer deutlichen Zeitersparnis und reduziert die Wahrscheinlichkeit typischer Fehlerquellen wie beispielsweise Schreibfehler und Vergessen einzelner Seiten. Weiterhin bringt der Ansatz einen Vorteil für die Performance des Webseiten-Renderings: Anstatt für jede Seite einer Domain deren Layout bei jedem Aufruf mit zu laden, kann das Layout einmalig beim ersten Zugriff geladen werden. Dies reduziert die benötigte Bandbreite und erlaubt es der Render-Engine des Browsers das zugrundeliegende Layout template-Artig zu cachen. Über letzteres kann die Anzeige weiterer Seiten derselben Domain (mit demselben CSS-Aufbau) deutlich schneller erfolgen. Ebenfalls in diesem Kontext ist der Vorteil der HTML-Beschreibungsreduktion verankert. Anstatt das der Server die Layout-Beschreibung jedes mal über das Netzwerk senden muss, kann diese einmalig ausgeliefert und dann auf dem Client verwendet werden. Im HTML-Code muss maximal zusätzlich die CSS-Klasse oder ID angegeben sein.

15. Erstellen Sie einen Zeitstrahl mit den wichtigsten Meilensteinen der HTML- und CSS-Entwicklung.

Lösung:

16. Erklären Sie anhand eines geeigneten Beispiels den grundlegenden Aufbau von CSS-Definitionen.

Lösung: CSS-Definitionen setzen sich immer aus einem Selektor aus, welcher die HTML-Elemente definiert, auf welche die Styling-Regeln beim Rendering angewandt werden sollen. Welche Selektor-Typen es gibt, wird in Aufgabe 08 besprochen. In einem durch geschweifte Klammern begrentzten Style-Block erfolgt das Styling. In diesem Block werden die Style-Regeln als key-value-Paare angegeben. Verfügbare Schlüssel (Key) sowie die Werte (Values), die Sie annehmen können, sind im CSS3 Standard festgelegt. Beispielsweise kann die Hintergrundfarbe über den Schlüssel *background-color* angegeben werden, welcher RGB-Werte in Hexadezimal-Schreibweise (*#FF0000*) oder textuelle Farbnamen (*red*) als Werte annehmen kann. Ein Beispiel ist in Abbildung 1 angegeben.

17. An welchen Stellen im HTML können Sie CSS-Angaben definieren?

Lösung: CSS kann an drei Stellen definiert werden: In einer externen Datei, im Header einer Webseite sowie direkt am Tag. Siehe hierzu `https://www.w3schools.com/css/css_howto.asp`.

18. Erklären Sie die verbindliche Reihenfolge, anhand der Browser die zu verwendenden CSS-Regeln auswählt.

Lösung: Die Rendering-Reihenfolge ist (logischer Weise) folgende: Der Browser nimmt zuerst alle extern eingebundenen CSS-Dateien und wendet deren Layout auf die Inhalte an. Diese Regeln haben also die niedrigste Priorität für die finale Darstellung. Daran anschließend wendet die Render-Engine die Befehle an,

```
p {
 background-color: #FF0000;
 margin: 0;
 padding: 0;
 border: 1px solid;
}
\end{verbatim}
```

Abbildung 1: Beispiel-Definition für den Selektor p. Alle p-Tags werden mit roter Hintergrundfarbe (im Beispiel durch RGB-Codierung) gerendert. Weiterhin haben sie weder einen inneren noch einen äußeren Abstand zu umschließenden bzw. umschlossenen Elementen und besitzen einen 1Pixel breiten durchgehenden Rahmen.

welche sie im head-Tag der HTML-Beschreibung findet. In diesem Fall werden die Stylings der externen CSS-Dateien für die finale Darstellung überschrieben. Dementsprechend haben die Stylings im Head die zweitniedrigste Priorität. Sollte das scoped-Attribut implementiert sein, so prüft die Render-Engine, ob in einem HTML-Tag ein style-Tag mit gesetztem scoped-Attribut existiert und wendet die darin beschriebenen Designspezifikationen auf die Kind-Elemente des HTML-Tags an. Dieses hat die zweithöchste Priorität. Das finale Layout wird dann über das style-Attribut festgelegt. Sollte dieses an einem Tag angegeben sein, so überschreibt die Render-Engine jegliches bisher festgelegtes Design mit den dort angegebenen Definitionen. Es hat also die höchste Priorität. Wichtig für diese Abarbeitungsreihenfolge: Es werden nur identische Attribute überschrieben. Es ist beispielsweise durchaus möglich in der externen CSS-Datei die Hintergrundfarbe, im HTML-head die Font-Familie, in einem style-tag mit gesetztem scope-Attribut die Textfarbe und im style-Attribut des p-tags den margin dieses Tags zu setzen um ein Layout zu erhalten, dass alle definierten Eigenschaften umsetzt.

19. Nennen und erklären Sie die gängigsten CSS-Selektoren.

Lösung: Im Kontext von CSS ist Selektor der Überbegriff für die Definition des zu stylenden Elements. Dieser erfolgt über eine eindeutige Identikationsbeschreibung. Schematisch gesprochen ist der Selektor der Bezeichner, der vor dem, mit geschweiften Klammern umschlossenen, Styling-Block angegeben ist. Ein Selektor kann sich dabei auf eine komplette Tag-Familie beziehen, auf eine CSS-Klasse, auf eine einer Tag-Familie zugeordneten CSS-Klasse sowie auf eine CSS-ID.

- Typselektor `Elementname`: CSS-Typselektoren matchen Elemente über den Knotennamen
- Klassenselektor `.klassenname`: In einem HTML-Dokument matchen CSS Klassenselektoren ein Element basierend auf den Inhalten des class Attributs des Elements.

- ID-Selektor `#idname`: In einem HTML-Dokument matchen CSS ID-Selektoren ein Element basierend auf den Inhalten des id Attributs des Elements

Selektoren lassen sich über Kombinatoren kombinieren. Gängige Kombinatoren sind der Nachfahrkombinator `A B`, der Kindkombinator `A > B`, der direkte Nachbarkombinator `A + B` sowie der indirekte Nachbarkombinator `A ~ B`.

`https://www.w3schools.com/cssref/css_selectors.asp` liefert einen Überblick und Beispiele über Selektoren und Kombinatoren.

20. Erklären Sie das Box-Prinzip von CSS. Welche Bedeutung haben die Begriffe *margin*, *border*, *padding* und *content* hierbei?

Lösung:

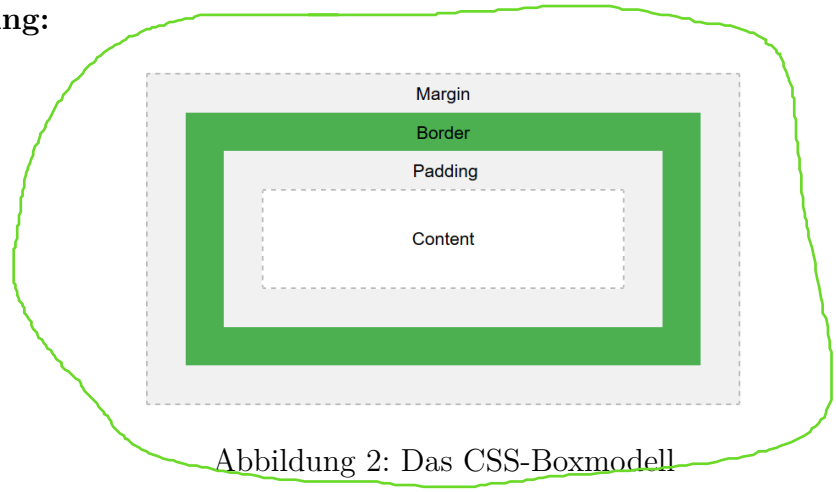

Abbildung 2: Das CSS-Boxmodell

Das Box-Prinzip von CSS bestimmt wie HTML-Elemente für das Rendering grundlegend aufgebaut sind. Diese werden immer als Box aufgefasst (siehe Abbildung 2). Jedes Element in HTML kann weitere Elemente verschachtelt enthalten. Deren finale Positionierung beeinflusst sich durch die Angaben des inneren Abstands (Padding), des äußeren Abstands (Margin) und dem Rahmen (Border). Der Margin gibt an, wie viel Abstand ein Element zu seinem Elternelement halten muss. Die Angabe kann in Pixeln, Prozentual zur Seitengröße (in %) und in Abhängigkeit der Schriftgröße (in *em*) erfolgen. Die Positionierung erfolgt am linken oberen Eck des Rahmens. Der Rahmen kann dabei wiederum eine individuelle Breite (in px, % oder em) besitzen und dargestellt werden (beispielsweise als durchgezogene Linie, als gestrichelte Linie oder als 3D-Darstellung) oder unsichtbar sein. Verfügbare Borders sind auf `https://www.w3schools.com/css/css_border.asp` gelistet. Das Padding gibt an, wie viel Abstand ein Element zu seinen Kindelementen halten muss. Die Angabe kann in Pixeln, Prozentual zur Seitengröße (in %) und in Abhängigkeit der Schriftgröße (in *em*) erfolgen. Bei verschachtelten Elementen addieren sich Margin und Padding zum Gesamtabstand zwischen den Bordern der Elemente. Der Content eines Elements entspricht seinen Kind-Elementen. Diese können entweder weitere HTML-Strukturen sein oder atomare Elemente wie Text oder Bilder.