universität wien

AUSZUG DER DIPLOMARBEIT / EXCERPT OF THE DIPLOMA THESIS

Titel der Diplomarbeit / Title of the Diploma Thesis

„Kekse ohne Salz schmecken nicht“
Ein innovatives Unterrichtskonzept zur Vermittlung von Security im Webdatenbereich

verfasst von / submitted by

Simon Marik

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Magister der Naturwissenschaften (Mag.rer.nat.)

Wien, 2017 / Vienna, 2017

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 190 884 313
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Lehramtsstudium UniStG UF Informatik und Informatikmanagement UniStG - 10/2003 UF Geschichte, Sozialkunde und Polit. Bildg. UniStG - 10/2008
Betreut von / Supervisor:	ao. Univ.-Prof. Dipl.-Ing. Dr. Renate Motschnig
Mitbetreut von / Co-Supervisor:	Ass.-Prof. Mag. Dr. Christian Cenker

Standards in der Webentwicklung

In den vergangenen 30 Jahren hat sich das Internet und vor allem dessen Inhalt technisch sowie optisch grundlegend verändert. Etwas, das sich allerdings nicht geändert hat, ist die altertümliche – womöglich durch Papier geprägte – Erwartungshaltung mancher NutzerInnen und WebentwicklerInnen in Bezug auf die Handhabung mit dem Internet bzw. dessen Websites, welche sich durch zwei unterschiedliche Prinzipien erklären lässt. Zum einen wird erhofft, dass AutorInnen die absolute Kontrolle über die Website haben und zum anderen erwarten sie, dass die Website auf allen Endgeräten, darunter auch allen mobilen, gleich aussieht („responsives Design“).[1]

John Allsopp versuchte in diesem Sinne bereits im Jahr 2000 seine Erfahrungen sowie Prinzipien des Webdesigns in seinem Aufsatz „A Dao of Web Design“ zusammenzufassen, indem er auf das uralte chinesische Weisheitsbuch „Dao De Jing“ Bezug nimmt, welches der Sage nach auf den Gelehrten Lao Tse zurückgeht. Aus diesem Buch, das mit seinen insgesamt 81 harmonischen Versen eine dennoch relativ kryptische Sammlung darstellt, kann man allerdings der folgenden Philosophie aufgrund der oben genannten Problematik sehr viel abgewinnen.[2]

„Akzeptiere, was vor dir liegt, ohne die Situation ändern zu wollen. Wenn wir genau hinschauen, werden wir sehen, dass die Arbeit schneller und leichter vorangeht, wenn wir aufhören, etwas zu wollen.“[3]

Demnach ist der Lehre des Lao Tse zu entnehmen, dass das Web flexibel ist und diese Flexibilität angenommen sowie akzeptiert werden soll, um eindrucksvolle und vor allem mediengerechte Webseiten erstellen zu können. Man muss allerdings dazu sagen, dass der Aufsatz von John Allsopp schon ein bisschen in die Jahre gekommen ist, wenn man bedenkt, dass die Wunderwaffen der Webentwicklung damals noch `<font>` sowie `<table>` hießen. Die Gestaltung von Webseiten mittels CSS stand zu diesem Zeitpunkt gerade erst am Anfang.[4] Erst durch eine eigene Bewegung im Jahr 1998, welche sich für die Vereinheitlichung und Verbreitung expliziter Webstandards einsetzte, konnte es gelingen, HTML-Tabellen aus den Entwicklerstudios und somit aus Webseitendesigns zu verbannen.[5]

Grundsätzlich unterscheidet man heutzutage in der modernen Webentwicklung nur mehr zwischen „Mobile First“ und „Desktop First“ Webseiten, wobei beide Herangehensweisen Vor- und Nachteile in sich bergen. Im Internet findet man sehr gut als auch sehr schlecht programmierte Musterbeispiele beider Varianten. Wirklich wichtig ist allerdings, dass der Inhalt, auch „Content“ genannt, auf allen Geräten zugänglich ist. Genau aus diesem Grund möchte Müller hier auch den Terminus „Content First“ begründen, da er der Ansicht ist, dass Webseiten nur dazu da sind, um Inhalte an Besucher auszuliefern und somit von Anfang an im Zentrum aller Überlegungen stehen sollten. Ob

[1] vgl. *Müller* (2013), S.24.
[2] s. *Allsopp*: A Dao of Web Design, (Stand: 24.09.2016).
[3] *Müller* (2013), S.24.
[4] vgl. *ebd.*, S.25.
[5] s. *The Web Standards Project*: Working together for Standards, (Stand: 24.09.2016).

sich dieser Trend in naher Zukunft noch durchsetzen wird, bleibt allerdings abzuwarten.[6]

„Bis dorthin bleibt der Weg in die Zukunft sehr steinig, da es zur Verbreitung von HTML5 und CSS3 noch einige bekannte, wie auch unbekannte Hürden zu bewältigen gibt. Auf der einen Seite ist der Internet Explorer aus dem Hause Microsoft nach wie vor einer der am meist verbreitetsten Browser weltweit, welcher mit seiner äußerst marginalen Unterstützung von HTML5 und CSS3 bis einschließlich Version 11, der Webstandardverbreitung ziemlich im Weg stand und mit dem neuen Edge-Browser unter Windows 10 nach wie vor steht. Auf der anderen Seite dürfen wir nicht ganz so einfach mit den neuen Technologien, gerade im Medieneinsatz (z.B. von Audio und Video) davongaloppieren und die Barrierefreiheit für Sehbehinderte sowie insbesondere ältere Menschen aus den Augen verlieren.“[7]

Der Internet Explorer ist natürlich nicht der einzige Browser, der in gewisser Art und Weise Probleme macht. Selbst die Varianten von Google, Apple oder Mozilla streiten sich z.B. um die Unterstützung von Audio- und Videocodecs und unterstützen schlichtweg die Formate des anderen nicht.[8] Hoffentlich werden diese Unterschiede – sofern die finanziellen Interessen dieser Unternehmen jemals in den Hintergrund geraten sollten – irgendwann einmal allesamt wegfallen. Doch bis dorthin werden sich die Spezifikationen von HTML5 und CSS3 noch mehrmals weitgehend ändern und mit ihnen die Browserspezifischen HTML5-, sowie CSS3-Implementierungen von Chrome, Safari und Firefox.[9] Der Fokus dieses Kapitels liegt jedoch auf den aktuellen Spezifikationen der Arbeitswerkzeuge HTML5, CSS3, PHP und MySQL, welche im Nachfolgendem enger zusammengefasst und dabei näher erläutert werden.

0.1 HTML5

Wie bereits festgestellt wurde, handelt es sich bei HTML5[10] um eine reine Spezifikation, welche neue Tags, Markups sowie neue JavaScript APIs mit sich bringt. Dabei geht es keineswegs um einen neuerlichen Internethype oder gar einer Plattform, wie manche WebentwicklerInnen fälschlicherweise glauben, obwohl es das W3C selbst als solches propagiert, wie man in der folgenden Abbildung *(Abb. 2.1.1)* deutlich erkennen kann. Leider muss man diesen Sachverhalt – unter anderem deswegen – des öfteren klarstellen, da rund um diese Definition mit der Zeit so viel Verwirrung entstanden ist, dass mittlerweile KundInnen aber auch manche EntwicklerInnen CSS3-Spezifikationen wie Schatten, Verläufe und Transformationen plötzlich als HTML bezeichnen.[11]

Semantik spielt in diesem Kontext eine wirklich große Rolle und begründet im Wesentlichen auch die fundamentale Basis von HTML5. Denn genau dieses Verlangen nach einer besseren Auszeichnung von semantischen Inhalten, wie z.B. durch Suchmaschinen, barrierefreien oder auch normalen, mobilen Webseiten, hat die Entwicklung von HTML5 seit 2007 bis heute enorm vorangetrieben. Ein weiteres großes Ziel von HTML5 besteht darin, Audio-, Video- und sonstige animierte Inhalte ohne jegliche Plug-Ins darzustellen, um auf lange Distanz Third-Party Applikationen wie z.B. den Adobe Flash Player auszumerzen. Dieser wird zwar, aufgrund erheblicher Sicherheitslücken, kaum noch auf Mobilgeräten verwendet. Allerdings würden mobile Endgeräte durch die Nutzung mehrerer Plug-Ins auch zu viele Ressourcen in Form von Akkulaufzeit oder gar Prozessorleistung einbüßen, wodurch sich die Ersparnis solcher Plug-Ins selbstverständlich bezahlt macht.[12]

[6] vgl. *Müller* (2013), S.42.
[7] *Hogan* (2011), S.15.
[8] vgl. *ebd.*, S.18.
[9] vgl. *ebd.*, S.19.
[10] s. *W3C*: HTML5 Recommendation 2014, (Stand: 24.09.2016).
[11] vgl. *Hogan* (2011), S.4.
[12] vgl. *Potschien* (2013), S.11.

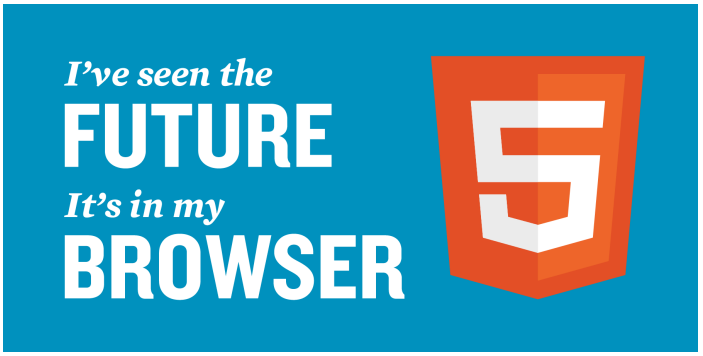

Abbildung 0.1.1: Offizieller HTML5-Sticker inkl. Logo des W3C

Falls man noch alte HTML4 Dokumente oder dergleichen finden sollte, kann man ebenfalls völlig beruhigt bleiben, da die meisten bisherigen Spezifikationen aus vorherigen HTML Versionen selbstverständlich beibehalten und sogar stellenweise optimiert wurden, sodass der Umstiegsaufwand auf HTML5 relativ gering ist. So können beispielsweise alte `<b>` (d.h. fetter Text) oder `<i>` (d.h. kursiver Text) Elemente genauso zu einem validen HTML5 Dokument führen, wie das bereits unter HTML4 üblich war. Das gleiche gilt natürlich auch für die Auszeichnung von Texten, Tabellen, Überschriften, Bilder und Listen. Bei der Entwicklung selbst sind viele alte Elemente, welche keine semantische Bedeutung hatten sondern ausschließlich Gestaltungszwecken dienten, übernommen worden. Eigentlich ziemlich kontraproduktiv, wenn man bedenkt, dass seit der Einführung von CSS 1994 der Usus geprägt wurde, Syntax von Design zu trennen. Dass diese alten Elemente allerdings dennoch überlebt und sogar eine passende Semantik bekommen haben, liegt auch einzig und alleine am Anpassungsaufwand der Auszeichnungen.[13]

HTML5 bietet außerdem einen praktischen und vor allem sicheren Workaround gegen die Browserseitigen „Cross-Site Scripting" Einschränkungen, bei denen verhindert wird, dass man Skripte einer bestimmten Domäne dazu verwenden kann, Skripten einer anderen Domäne zu ändern bzw. auszuführen. Auch eine Unterstützung von sogenannten Websockets wird mittlerweile angeboten, bei denen eine dauerhafte Verbindung zum Server gehalten werden kann, ohne ständig vom Backend die Fortschrittsaktualisierungen abzufragen. Webstorage und WebSQL Datenbanken, welche Daten dauerhaft am clientseitigen Rechner abspeichern, sind dank der neuen Spezifikation absolut kein Tabuthema mehr. Die wohl wichtigste Neuerung stellt aber die verbesserte Barrierefreiheit z.B. bei Screenreadern dar, welche etwa für Personen mit Sehbehinderung Webseiteninhalte verarbeiten und anschließend vorlesen bzw. auf ertastbare Medien umleiten.[14] Da solche Screenreader allerdings nicht mit den schnelllebigen Webbrowser mitkommen, existieren zusätzlich auch noch sogenannte Webreader, welche für Webbrowser optimiert sind.[15] Dies wird durch die bessere Auszeichnung sowie durch neue Elementattribute ermöglicht, in denen sich Funktionen nun bestimmten Elementen zuordnen lassen, sodass Bildschirmlesegeräte diese leichter verarbeiten können.[16] Auch das W3C hat sich dieser Thematik mittels der WCAG-Spezifikation angenähert, welche bis heute gültige Richtlinien für barrierefreie Webinhalte zusammenfasst.[17]

Blickt man zurück, liegen die eigentlichen Wurzeln von HTML5 in einem WHATWG-Entwurf aus dem Jahr 2005 mit dem Titel „Web Applications 1.0", welcher die Richtung vorgab, das Web für zukünftige Webanwendungen fit zu machen. So sollte Software wie Textverarbeitung, Tabellen-

[13] vgl. *Potschien* (2013), S.13.
[14] vgl. *Hagelkruys* (2012), S.48.
[15] vgl. *ebd.*, S.49 f.
[16] vgl. *Hogan* (2011), S.10 ff.
[17] s. *W3C*: Web Content Accessibility Guidelines (WCAG) 2.0 2009, (Stand: 24.09.2016).

kalkulation, Präsentationstechnik, Bild- und Videobearbeitung und vor allem Computerspiele vom lokalen Speicherort in den Browser abwandern und das alles ganz ohne Flash bzw. Java. Aus diesem Grund wurden in dieser Diplomarbeit auch alle selbst erstellten Grafiken mittels einer Webapplikation („draw.io“)[18] gefertigt, um demonstrieren zu können, was zum aktuellen Stand bereits alles möglich ist. Als Musterbeispiel der Vorreiterrolle wäre hier dennoch Google mit seinen Office Anwendungen wie z.B. „Google Drive“ bzw. „Google Docs“ zu erwähnen. Bis allerdings weitere Softwareangebote im Browser funktionieren werden, müssen diese beispielsweise noch lernen, binäre Daten wie Bilder und Videos zu laden und zu bearbeiten, Programmstände offline abzuspeichern sowie sich an lokale Datenschutzgesetze zu halten, weswegen JavaScript in Zukunft eine noch wichtigere Rolle spielen wird als wie bisher.[19]

Der selbsternannte „HTML5-Erklärbär“ Peter Kröner, hat in seinem privaten Blog versucht, eine Landkarte des sogenannten „HTML5-Universums“ *(Abb. 2.1.2)* zu zeichnen, um ein für alle Mal klarzustellen, wie sich HTML5 nun definiert und was genau darüber hinausgeht.[20]

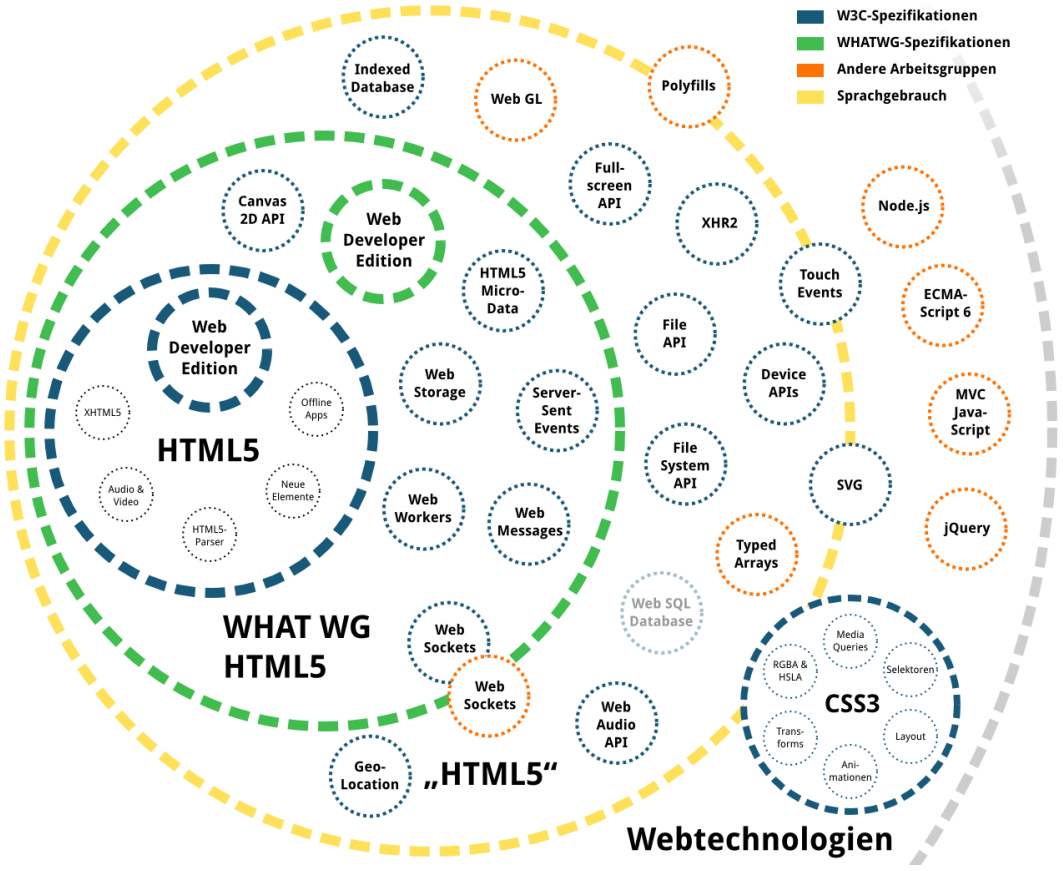

Abbildung 0.1.2: Die Karte des „HTML5-Universums“ nach Kröner

Der zentrale blaue Kreis links in der Mitte heißt „HTML5“ und stellt die HTML5-Spezifikation des W3C[21] dar. Die Spezifikation ist sehr umfangreich, da sie alle Details für die BrowserherstellerInnen enthält, die die Features implementieren müssen. Deshalb gibt es noch eine kompaktere Edition für WebentwicklerInnen. Der etwas größere grüne Kreis ist „HTML – The Living Standard“ von der WHATWG, der ebenfalls in einer Webdeveloper Edition existiert. Er umschließt die W3C-

[18] s. *Draw.io*: Flowchart Maker & Online Diagram Software, (Stand: 24.09.2016).
[19] vgl. *Müller* (2013), S.62 f.
[20] vgl. *ebd.*, S.63.
[21] s. *W3C*: HTML5 Recommendation 2014, (Stand: 25.09.2016).

Spezifikation, enthält aber noch eine Menge mehr. Alles außerhalb dieser beiden Kreise ist nur noch unter Anführungsstrichen als HTML5 zu bezeichnen und sozusagen Teil des erweiterten Universums. Der kleine Kreis namens „Geolocation“ z.B. gehört weder zur W3C noch zur WHATWG, wird aber interessanterweise fast überall als Feature von HTML5 bezeichnet.[22] Selbiges gilt auch für Node.js oder gar jQuery, aber man sieht sehr deutlich anhand der vielen kleinen orangen Kreise, dass das Web mittlerweile sehr viel Neues zu bieten hat.[23]

Zusammenfassend kann man also sagen, dass der Fokus bei HTML5 ganz eindeutig auf Webanwendungen liegt, der sich unter anderem folgenden zentralen Themen widmet:

- Neue Elemente für Struktur und Semantik von Webseiten wie <header>, <nav>, <footer>, <main>, <aside>, <section> und <article>
- Verbesserte Barrierefreiheit für Screenreader usw.
- Neue Formularelemente wie Suche, E-Mail usw.
- Multimedia Elemente und APIs für Video und Audio ohne Fremdtechnologien
- Canvas (per JavaScript programmierbare Bitmap-Grafiken für Animationen, Diagramme usw.)
- Offline-Webanwendungen, die auch ohne Internetverbindung funktionieren
- Visuelle Effekte sowie fortschrittliche Selektoren, dank der CSS3 Erweiterung[24]

0.1.1 Metadaten

Wer glaubt, dass es von HTML5 mehrere Versionen gibt, der irrt gewaltig. Von HTML5 gibt es nämlich nur eine einzige Version, bei der man praktischerweise keinerlei Referenzen mehr auf eine bestimmte DTD angeben muss. So war es bisher nämlich vorgesehen, über den DOCTYPE die verwendete HTML Version einschließlich der verwendeten Variante (z.B. `"transitional"`, `"framset"`, oder `"strict"`) anzugeben. In HTML5 wird nur mehr sehr platzsparend angegeben, dass es sich dabei um ein HTML Dokument handelt, wodurch jede HTML Vorgängerversion automatisch aufgrund der neuen und kurzen Schreibweise umgangen wird.[25] Die vereinfachte DOCTYPE-Deklaration *(Abb. 2.1.3)*, welche original nur mehr aus <!DOCTYPE html> besteht, ist nämlich ebenfalls problemlos dazu in der Lage, Dokumente im Browser standardkonform darzustellen. Selbst in XML geschriebene HTML5 Dokumente benötigen keine DOCTYPE-Angabe mehr. Stattdessen können sie eine XML-Deklaration erhalten, bei dem der MIME-Typ `"application/xhtml+xml"` bzw. `"application/xml"` verwendet werden sollte.[26]

```
1  <!DOCTYPE html>
2     <!-- optionale Kommentare -->
3     <html lang="de">
4        <head>
5           <meta name="author" content="Simon Marik">
6           <meta name="description" content="Thesis Projekt">
7           <meta name="viewport" content="width=device-width,
                initial-scale=1.0, user-scalable=no">
8           <title>Thesis Projekt</title>
9           <link rel="stylesheet" type="text/css"
                href="css/thesis.css">
```

[22] vgl. *Müller* (2013), S.63 f.
[23] s. *Kröner*: Die Karte des HTML5-Universums, (Stand: 25.09.2016).
[24] vgl. *Müller* (2013), S.65.
[25] vgl. *Potschien* (2013), S.14 f.
[26] vgl. *Niederst Robbins* (2014), S.7.

```
10        </head>
11        <body>Webseiten Inhalt...</body>
12    </html>
```

Abbildung 0.1.3: Minimalbeispiel der neuen HTML5 DOCTYPE-Deklaration

Wenn man bedenkt, dass die HTML4 DOCTYPE-Deklaration damals aus einer schier unendlich langen Zeichenkette bestand, die man nur in den seltensten Fällen auswendig wusste und sich meistens von irgendwo anders kopierte, war es schon äußerst sinnvoll, an dieser bei der Spezifikationsaktualisierung einzusparen. Dabei bleibt es auch vollkommen egal, ob man `<!DOCTYPE html>` oder `<!doctype html>` schreibt, was für viele, vor allem für englischsprachige NutzerInnen, extrem komfortabel sein sollte. Interessant daran ist, dass früher selbst die Browser diese endlos lange Zeichenkette nicht einmal gelesen haben. Es war und ist nach wie vor nur wichtig, dass der DOCTYPE in der ersten Zeile steht, um den Browser in den Standardmodus zu versetzen. In der zweiten Zeile geht es dann direkt mit dem Stammelement `<html>` weiter, welches dem Browser den Dokumententyp mitteilt. Sollte die Standardsprache der zu erstellenden Website nicht Englisch sein, so schreibt man z.B. für eine deutsche Seite `<html lang="de">`.[27]

Die Angabe des verwendeten Zeichensatzes erfolgt bei HTML5 nicht mehr über die Angabe des `"http-equiv"` Attributs, sondern wird über das neue `"charset"` Attribut innerhalb eines `<meta>` Elements angegeben. Bei der Angabe von `"http-equiv"`, musste man früher nämlich auch noch den MIME-Typ angeben, wodurch der Eintrag erheblich länger wurde.[28] Als Zeichensatz ist es empfehlenswert UTF-8 zu verwenden, da es eigentlich – außer bei alten Datenbanken mit ISO Zeichensatz – keine guten Gründe gibt, diesen nicht zu verwenden (Default = UTF-8). Der Seitentitel `<title>` sowie die Seitenbeschreibung `<meta name="description" content="...">` haben sich seit damals nicht verändert. Diese beiden Elemente beschreiben unsichtbar den Inhalt einer Seite und sind vor allem für Suchmaschinen wichtig. Man kann das etwa mit einem Eintrag bei Google vergleichen. Der Seitentitel repräsentiert dabei die Überschrift, welche kurz und bündig sein sollte und die Seitenbeschreibung in max. 150 Zeichen die zweizeilige Beschreibung unterhalb der grünen URL abdeckt. Auch die Angabe der AutorIn `<meta name="author" content="...">` ist bei der Umstellung weitgehend gleich geblieben. Einzig und allein das Element für den Viewport namens `"meta-viewport"` kam neu in der `<meta>`-Welt dazu, obwohl es eigentlich nur sehr wenig mit HTML5 zu tun hat. Dieses Element ist nämlich eine Erfindung von Apple und hat sich mit der Zeit zu einer Art „Semistandard“ weiterentwickelt. Als im Jahr 2007 das iPhone auf den Markt kam, gab es noch keine für Smartphones optimierten Webseiten, wodurch die iOS-Entwickler gezwungen waren, Webseiten in mobilen Browsern soweit zu verkleinern, dass sie vollständig auf die kleinen Displays passten. Dabei wurden Seiten auf die gedachte Breite von 980 Pixel (beim iPhone) bzw. 800 Pixel (bei Android) gerendert und anschließend auf die tatsächliche Displaybreite von 320 Pixel heruntergezoomt. Diese Technik findet heutzutage nach wie vor bei so ziemlich allen mobilen Webbrowsern auf fast allen mobilen Betriebssystemen Anwendung, außer das CSS der Website wurde durch Media Queries optimiert. Der optionale Content des Viewport Elements `"width=device-width, initial-scale=1.0"` definiert die Gerätebreite als eigentliches Breitenmaß und verweigert letzten Endes den BenutzerInnen das Verkleinern der Website. Abschließend lässt sich noch sagen, dass man bei „Mobile First“ Webseiten immer den Meta-Viewport verbauen sollte, ihn aber bei „Desktop First“ Varianten getrost weglassen kann.[29]

[27] vgl. *Müller* (2013), S.72.
[28] vgl. *Potschien* (2013), S.15.
[29] vgl. *Müller* (2013), S.73 ff.

0.1.2 Tags

Im direkten Zusammenhang mit HTML5 steht, wie vorhin bereits erwähnt, die sogenannte Semantik, welche semantische Elemente für die Auszeichnung des Seiteninhalts verantwortlich sind. Als sich im Jahr 2004 die WHATWG zusammenfand, kamen CSS-basierte Layouts gerade erst richtig in Schwung und mit ihnen verbreitete sich auch das sogenannte -Element. Erweitert um diverse IDs wie oder Klassen wie , war es eine unglaubliche Bereicherung für alle WebseitenautorInnen, ihre Seiten zu strukturieren und per CSS zu stylen. In HTML5 ist das -Element ebenfalls kein Tabu, allerdings gibt es eine Reihe neuer semantischer Elemente *(Tab. 2.1.1)*, welche in vielen Situationen oft die bessere Wahl sind.[30]

Neue HTML5-Elemente:		
<article>	<figcaption>	<output>
<aside>	<figure>	<progress>
<audio>	<footer>	<rp>
<bdi>	<header>	<rt>
<canvas>	<hgroup>*	<ruby>
<command>*	<keygen>	<section>
<data>**	<main>**	<source>
<datalist>	<mark>	<time>
<details>	<menuitem>**	<track>
<dialog>**	<meter>	<video>
<embed>	<nav>	<wbr>

* Nicht mehr Teil von HTML5

** Nur WHATWG und HTML5

Tabelle 0.1.1: Ein Auszug der neu unterstützten HTML5-Elemente [31]

Selbstverständlich kann man nach wie vor anstatt einer <h1>-Überschrift ein -Element mit selbigen Layout basteln. Die normalen BenutzerInnen würde das im Quelltext sicherlich kaum stören, aber Suchmaschinen würden das -Element keinesfalls als <h1>-Überschrift erkennen. Die neue Semantik hilft also, wie man sieht, vor allem Suchmaschinen, Screenreadern, Browsern aber auch so manche EntwicklerIn. Vor allem bei Layoutbereichen wie der Navigationsleiste oder der Fußzeile muss man nicht mehr auf zahlreiche -Elemente mit unzähligen Klassen oder IDs zurückgreifen, sondern kann sich einfach aus der neuen HTML5 Tagsammlung bedienen *(Abb. 2.1.4)*. Ähnlich wie beim Überschriftenbeispiel haben die neuen Elemente innerhalb des Quelltexts auch ihre eigene Bedeutung, wodurch Suchmaschinen beispielsweise in der Lage sind, Suchbegriffe im Inhaltsbereich <main> höher zu bewerten als in Bereichen wie z.B. <nav> oder <footer>.[32]

Mit der neuen Spezifikation wird auch eine Reihe neuer Eingabetypen für Inputelemente in Formularen definiert, welche innerhalb des <input>-Elements als Wert für das "type" Attribut angegeben werden. Dabei sind Werte wie "color", "date", "datetime", "time", "email", "week", "month", "number", "range", "search" oder "tel" möglich. Um auch der wachsenden Nachfrage nach interaktiven Webinhalten nachzukommen, wurden in HTML5 viele neue APIs eingeführt, welche Tasks proprietärer Plug-Ins standardisieren. Für einige dieser APIs existieren sogar eigene Markup-Bausteine wie beispielsweise für audio, video und canvas. Andere verwenden schlichtweg JavaScript oder serverseitige Komponenten. Folgende APIs sind beispielsweise unter anderem Teil der HTML5-Spezifikation des W3C:

- Media API zum Abspielen von Video- und Audiodateien inkl. Multimedia-Synchronisierung.

[30] vgl. *Müller* (2013), S.76 f.
[31] *Niederst Robbins* (2014), S.4.
[32] vgl. *Müller* (2013), S.79 ff.

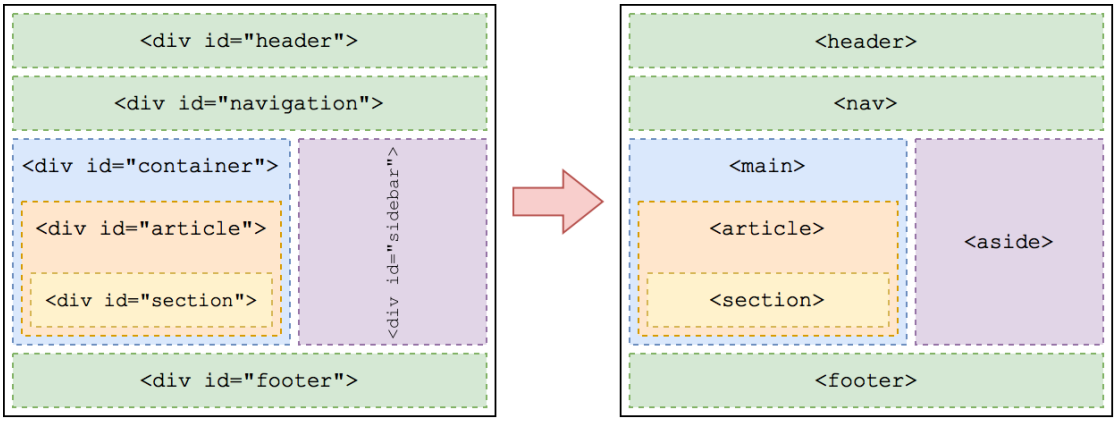

Abbildung 0.1.4: Auszeichnung der Layoutbereiche von HTML4 nach HTML5

- TextTrack zum Einbinden von zeitgesteuerten Untertiteln bei Videoelementen
- Session History API für die Arbeit mit dem Browserverlauf
- Editing API (enthält das neue globale Attribut `"contenteditable"`)
- Drag and Drop API (enthält das neue Attribut `"draggable"`)
- Web Workers API (ermöglicht das Ausführen von Skripts im Hintergrund)
- File API (ermöglicht den Zugriff auf per Formular hochgeladene Dateien)[33]

Es wurden aber auch im Vergleich zu HTML4 einige gebräuchliche Elemente wieder abgeschafft. Einige der Tags wie z.B. `<basefont>`, `<big>`, `<center>`, `<font>`, `<s>`, `<strike>`, `<tt>` oder `<u>` sind bereits extrem veraltet und werden unter HTML5 nicht mehr unterstützt. Es finden sich im WWW aber nach wie vor viele Seiten, welche mit visuellen Editoren wie Adobe Dreamweaver editiert werden und somit noch immer eine Menge an `<font>` und `<center>` Tags enthalten. Selbst die Unterstützung von Frames, welche bei sogenannten Enterprise-Webanwendungen (z.B. Microsoft Outlook Web Access) immer sehr beliebt waren, wurden restlos entfernt. Dies geschah aber aus gutem Grund, da Frames jeher extreme Schwierigkeiten bei der Benutzerfreundlichkeit und auch bei der Barrierefreiheit verursachten und somit einfach nicht tragbar waren. Mit den Frames verschwanden selbstverständlich auch die Elemente `<frame>`, `<frameset>` sowie `<noframes>` und gestalterische Frametools wie `<acronym>` wurden z.B. durch bessere Optionen wie `<abbr>` ersetzt. So viele Attribute wie auch in der neuen Spezifikation dazugekommen sind, von so vielen hat man sich auch beim Umstieg zur neuen Version wiederum verabschiedet. So gehören Attribute wie beispielsweise `"align"`, `"link"`, `"bgcolor"`, `"height"`, `"width"`, `"scrolling"`, `"hspace"`, `"vspace"`, `"valign"`, `"cellpadding"`, `"cellspacing"` oder `"border"` mittlerweile der Vergangenheit an. Aber auch ein paar wichtige Tags wie `"target"`, `"profile"` oder `"longdesc"` mussten leider den Weg für HTML5 ebnen und räumen. Viele Entwickler rebellierten bei `"longdesc"`, da dieses Attribut die einzige Möglichkeit darstellte, BenutzerInnen mit Bildschirmlesegeräten zusätzliche beschreibende Informationen anzubieten.[34]

Josh Duck hat in seinem privaten Blog – um wieder etwas Licht in das Dickicht der Tags zu bringen – ein komplettes Periodensystem nach der aktuellen HTML5-Spezifikation des W3C angefertigt *(Abb. 2.1.5)*, aus dem alle funktionierenden Elemente nun endlich eindeutig hervorgehen sollten.[35]

[33] vgl. *Niederst Robbins* (2014), S.5 f.
[34] vgl. *Hogan* (2011), S.16 ff.
[35] s. *Mozilla Developer Network*: Liste der HTML5-Elemente, (Stand: 25.09.2016).

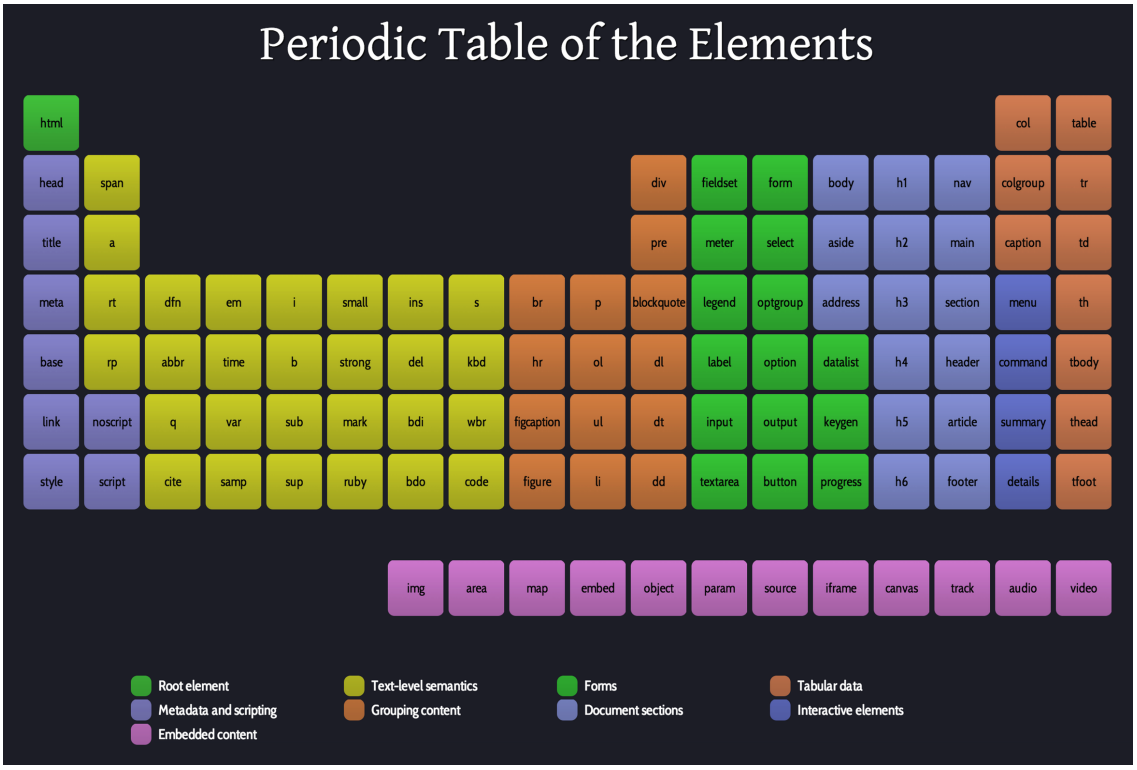

Abbildung 0.1.5: Das Periodensystem der HTML5-Elemente nach Duck

0.2 CSS3

CSS stellt eine Ergänzung zu HTML dar, welche vom W3C als eigener Standard innerhalb des HTML-Universums definiert und somit bereits auch propagiert wurde *(Abb. 2.2.1)*. Der CSS-Standard dient einerseits dazu, BrowserherstellerInnen genauestens anzugeben, wie eine CSS-Eigenschaft dargestellt werden soll, dient aber auch andererseits als zuverlässiges Referenzmaterial für WebentwicklerInnen. In einer solchen Referenz kann z.B. nachgelesen werden, mit welchen CSS-Eigenschaften was gesteuert werden kann, welche möglichen Parameter dabei eingesetzt werden können und welcher Effekt damit schlussendlich erzielt werden kann. Es ist interessant, dass CSS2.1 erst seit Juni 2011 vom W3C offiziell empfohlen wurde, obwohl dieser Standard bereits über Jahre hinweg erfolgreich in der Praxis Anwendung fand und CSS3 seit dem Jahr 2000 in Entwicklung war. Angesichts dessen wird es sicherlich noch einige Zeit bis zum Release von CSS3 dauern, obwohl eigentlich nur auf bereits Bestehendem aufgebaut wird.[36] CSS2.1 stellt nämlich die fundamentale Basis von CSS3 dar, was bedeuten soll, dass CSS3 – wenn es offiziell erscheint – rund alle 115 Eigenschaften des 2.1 Standards umfassen und dabei sogar auf ca. 300 Eigenschaften im eigenen Repertoire aufstocken wird. Für die Grundlagen bedeutet das, dass CSS3 nicht anders aufgebaut ist als die Version 2.1 des Standards und daher – ganz im Gegensatz zu unterschiedlichen HTML Versionen – die Schreibweise komplett ident geblieben ist.[37]

Wie man sieht ist CSS3 ähnlich wie HTML5 ebenfalls noch nicht abgeschlossen. Es gibt selbst noch einige Features welche zwar in CSS3 vorgesehen sind, die allerdings noch von keinem einzigen Browser unterstützt werden. Die meisten Eigenschaften sind bereits in allen gängigen

[36] s. *W3C*: CSS Current Status, (Stand: 26.09.2016).
[37] vgl. *Heller* (2012), S.234 f.

Abbildung 0.2.1: Offizielles CSS3-Klassenbild inkl. Logo des W3C

Browsern implementiert – teilweise sogar bereits in mehreren Versionen. Da sich solche Eigenschaften bereits etabliert haben, rät das W3C auch zur Verwendung dieser, da sie ältere Browser einfach ignorieren würden. Allerdings geht jeder Browser gerade bei der Einführung neuer Eigenschaften seinen eigenen Weg. Dadurch kann es beispielsweise passieren, dass sich Webseiten in unterschiedlichen Browsern komplett anders verhalten bzw. anders aussehen. Aus diesem Grund hat man die sogenannten „Vendor-Präfixe“ eingeführt, wodurch jede Browser-Engine sein eigenes Präfix inkl. eigener Eigenschaft hat, durch die es möglich ist, Eigenschaften für jeden Browser individuell einzusetzen. Selbstverständlich wird man bei den bereits etablierten CSS-Eigenschaften keine inkorrekte Darstellung bzw. Fehlverhalten mehr vorfinden, aber aufgrund der zahlreichen Nutzung älterer Browser zahlt es sich nach wie vor aus – genauso wie bei JavaScript Eigenschaften, die Vendor-Präfixe der eigentlichen Eigenschaft voran zu stellen.[38] Um dies abschließend besser veranschaulichen zu können, listet die folgende Abbildung *(Abb. 2.2.2)* alle gängigen Vendor-Präfixe der Browser Safari & Chrome (`-webkit`), Firefox & Linux-Browser (`-moz`), Internet Explorer & Edge (`-ms`) und Opera (`-o`) anhand der Eigenschaft `border-radius` auf.

```
.testclass {
   -webkit-border-radius: 0.3em;
   -moz-border-radius: 0.3em;
   -ms-morder-radius: 0.3em;
   -o-border-radius: 0.3em;
   border-radius: 0.3em;
}
```

Abbildung 0.2.2: Alle gängigen Vendor-Präfixe für die CSS-Eigenschaft `border-radius`

Zu den Neuerungen von CSS3 zählen unter anderem die vielen neuen, vorhin bereits erwähnten Eigenschaften, sowie die extremen funktionellen Erweiterungen. Während in CSS2.1 noch Farben,

[38] vgl. *Potschien* (2013), S.125 f.

Abstände, Schrifteigenschaften und weitgehend statisches, zweidimensionales Aussehen dominierte, liefert CSS3 inzwischen Eigenschaften, mit denen HTML-Elemente dynamisch bewegt bzw. animiert werden können. So könnte man meinen, dass all das, was prinzipiell einst JavaScript vorbehalten war, sich nun auf den Funktionsumfang von CSS3 verlagerte.[39] Der neue Standard führte aber auch komplett neue Features wie Sprachkonstrukte (z.B. Media Queries), verbesserte Selektoren, sogenannte `@font-face` Regeln, sowie ein komplett überarbeitetes Box-Modell *(Abb. 2.2.3)* ein, welches von innen nach außen die `content-box`, `padding-box`, `border-box` und die `margin-box` abdeckt.[40] Natürlich sind das nicht alle Neuerungen des Standards. Es gibt

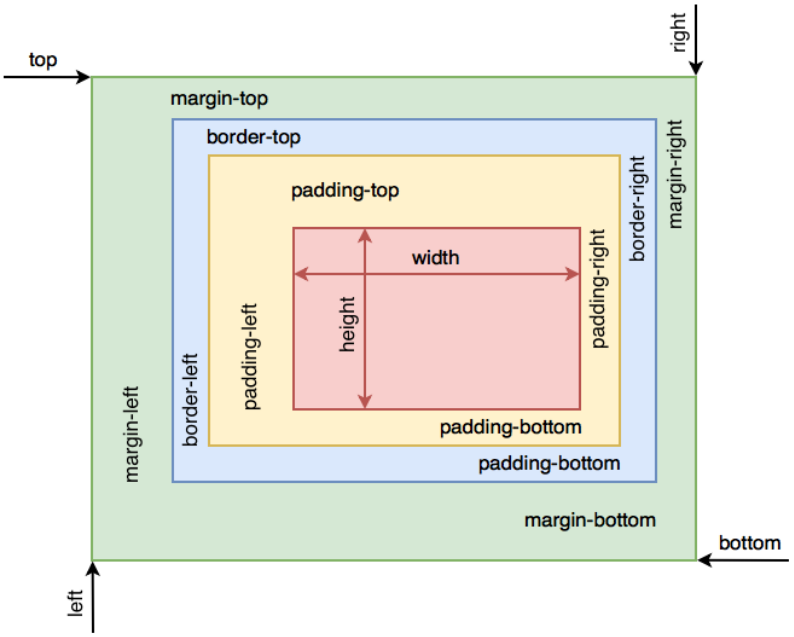

Abbildung 0.2.3: Neu konzipiertes Box-Modell des CSS3-Standards

noch unzählige weitere Eigenschaften, welche CSS3 auszeichnen und einer eigenen Erklärung bedürfen würden. Um das Ganze aber nicht unnötig ausschweifen zu lassen, stellt die folgende Aufzählungsliste die wohl wichtigsten neuen Möglichkeiten in der Eigenschaftswelt von CSS3 dar:

- Abgerundete Ecken
- Schatten
- Transparenz mit RGBA (Rot, Grün, Blau und Alpha - ein Format um Farbinformationen zu definieren, mischt die drei Grundfarben entsprechend von Dezimalzahlwerten zwischen 0 und 256.)
- Transforms (zwei- oder dreidimensionale Transformationen von HTML-Elementen)
- Transitions (Animation von HTML-Elementen)
- Animations (Transitions kennen nur einen Start- und einen Stop-Zustand des Objektes. Für komplexere Animationen wird eine sogenannte „Keyframe-Animation“ benötigt, die mehrere Zustände des Elements im zeitlichen Verlauf abbilden kann.)[41]

[39] vgl. *Heller* (2012), S.235.
[40] s. *SelfHTML Wiki*: CSS/Box-Modell, (Stand: 26.09.2016).
[41] vgl. *Ebner* (2013), S.5 ff.

0.2.1 Responsive Web Design

Ursprünglich angedachte „960-Grid-Frameworks“ verloren in der Webentwicklung zunehmend an Bedeutung, als im Jahre 2007 das iPhone von Apple präsentiert wurde. Durch das iPhone, sowie drei Jahre später durch das iPad, aber auch durch die große Vielfalt anderer verschiedener Gerätegrößen am Markt, entstanden völlig neue Wege der mobilen Internetnutzung, womit auch ein Paradigmenwechsel einherging. Die Annahme, dass Besucher von Webseiten vor einem Schreibtisch bei einem Computer sitzen und die Seite mit einer Maus und Tastatur bedienen, schwindet nach und nach immer mehr. Webentwick- lerInnen können aus diesem Grund heute nicht mehr sicher sein, auf welchen Endgeräten ihr Quelltext angezeigt wird. Dabei können die Bildschirmdiagonalen zwischen 3 Zoll und 50 Zoll oder auch mehr variieren, wenn man bedenkt, dass ein modernes Gerätespektrum vom kleinen Smartphone bis zum großen Fernseher alles abdeckt. Die französische Grafik- und Webdesignerin Stéphanie Walter griff das einstige Zitat des Kampfkünstlers und Schauspielers Bruce Lee – *“Empty your mind, be formless, shapeless – like water. [...] Be water, my friend.”*[42] – auf, um die erforderliche harmonische Anpassungsfähigkeit einer jeden WebentwicklerIn in der heutigen Zeit zu demonstrieren *(Abb. 2.2.4)*. Aber ganz abgesehen von den Geräten hat sich auch das Umfeld bzw. die Umgebung geändert, in welcher Menschen heutzutage Webseiten aufrufen. So surft man beispielsweise bereits in der Küche, im Bad, im Schlafzimmer oder einfach ganz bequem auf dem Sofa. So gesehen besteht die einzige Sicherheit darin, dass der Quelltext von einem Browser gerendert wird. Alles andere sind bloß Annahmen.[43]

Abbildung 0.2.4: Inhalt ist wie Wasser (i.e. “Content is like water”) nach Walter

Der Autor Ethan Marcotte beschrieb bereits im März 2009 die Vorteile sogenannter „fluid grids“, welche allerdings erst im Mai 2010[44] unter dem verbesserten Titel „Responsive Webdesign“, sowie unter der Ergänzung von Media Queries und flexiblen Elementen zu einer absolut bahnbrechenden Vision innerhalb der Webentwicklung wurden. Dabei beschrieb er drei verschiedene Techniken, mit denen seiner Meinung nach mediengerechte Webseiten gebaut werden sollten und responsives Webdesign ermöglichen:

[42] s. *Wikiquote*: Bruce Lee - Quotes, (Stand: 27.09.2016).
[43] vgl. *Müller* (2013), S.28 f.
[44] vgl. *Kadlec* (2013), S.13.

- Prozentbasierte Gridlayouts („fluid grids“)
- Flexible Bilder (z.B. SVG Grafiken)
- Media Queries[45]

„Das ist unser Weg nach vorne. Statt voneinander unabhängige Designs auf eine ständig wachsende Zahl von Geräten zuzuschneiden, können wir sie als unterschiedliche Facetten derselben User-Experience behandeln. Wir können für eine optimale Darstellung gestalten, aber gleichzeitig standardbasierte Technologien in unsere Designs einbetten, damit sie nicht nur flexibler werden, sondern sich auch besser an das jeweilige Medium anpassen, auf dem sie dargestellt werden.“[46]

Aufgrund solcher Artikel wie jener von Marcotte, aber auch durch Bücher und Publikationen anderer AutorInnen wurden die traditionellen Grundfeste des Webdesigns des Öfteren erschüttert, wodurch sich auch in weiterer Folge viel Verwirrung darum bildete. Die Grundidee des Responsive Webdesigns ist es, den HTML-Quellcode mittels Media Queries unterschiedlich innerhalb des CSS zu steuern. Marcotte verwendete nämlich, ganz im Gegensatz zu vielen anderen nach ihm, ein prozentbasiertes Raster für sein Layout, wodurch sich mit der Zeit eine Unterscheidung zwischen „responsive layouts“ und „adaptive layouts“ herauskristallisierte. Während das responsive Layout alle drei von Marcotte genannten Komponenten enthält, verzichtet das adaptive Layout auf schwer umzusetzende fluide Raster und arbeitet anstatt dessen mit festen Breiten innerhalb der Media Queries. Man kann aber getrost sagen, dass beide Methoden gleichberechtigte Formen des responsiven Webdesigns darstellen und somit original zwei unterschiedliche Wege zum selben Ziel sind.[47]

Abschließend sollen nun die wichtigsten Layoutoptionen angeführt werden, welche für die moderne Webentwicklung absolut essentiell, und in beinahe jedem professionell entwickelten Webprojekt vorzufinden sind:

- Feste Breite (Breite der Website wird durch ein Pixelmaß bestimmt - meistens 960 Pixel wegen der Primfaktorzerlegung: $960=2^6*3*5$)
- Fluid Layouts (Breite der Website wird durch eine Prozentangabe bestimmt - bei weitem flexibler z.B. 60% des Containers)
- Elastische Layouts (sind Fluid Layouts sehr ähnlich, jedoch wird die Begrenzung über die Schriftgröße „em“ vorgenommen - 1em entspricht dabei der Höhe der aktuellen Schriftgröße)
- Hybride Layouts (eine Kombination aus fluiden und elastischen Layouts)
- Schriftgrößen:
    - Pixel (zwar die bevorzugte Wahl bei jeder WebentwicklerIn, haben aber den Nachteil, dass sie nicht kaskadierungsfähig sind, keine definierte Größe haben und Probleme bei der Barrierefreiheit bereiten)
    - em (passen sich dynamisch an und können Schriftgrößen kaskadieren, können allerdings den Kontext ungewollt verändern)
    - Prozent (relativ ähnlich wie em, deswegen ist em auch die bevorzugtere Methode)[48]

0.2.1.1 Media Queries

Die wohl wichtigste Neuerung bei CSS3 sind die sogenannten Media Queries, wie vorhin bereits erwähnt wurde. Damit können CSS-Angaben in Relation zu den Eigenschaften des Ausgabegerätes vorgenommen werden. Prinzipiell werden Medienabfragen am häufigsten eingesetzt, um

[45] vgl. *Müller* (2013), S.30 f.
[46] s. *Marcotte*: Responsive Web Design, (Stand: 27.09.2016).
[47] vgl. *Müller* (2013), S.32 ff.
[48] vgl. *Kadlec* (2013), S.25 ff.

CSS-Statements für kleinere Displaygrößen zu setzen. Man kann sie aber auch in Kombinationen mit anderen Techniken benutzen. So ist es beispielsweise möglich, über Media Queries die feinen Unterschiede zwischen den vielen verschiedenen mobilen Ausgabegeräten zu berücksichtigen, um Inhalte bestmöglich darstellen zu können *(Abb. 2.2.5)*. Aber auch Webapplikationen können mittels dieser Abfragen eine angepasste Ausgabe erzeugen und zwar je nachdem, ob das Gerät im sogenannten „Portrait-View"- oder „Landscape-View"-Modus ist.[49]

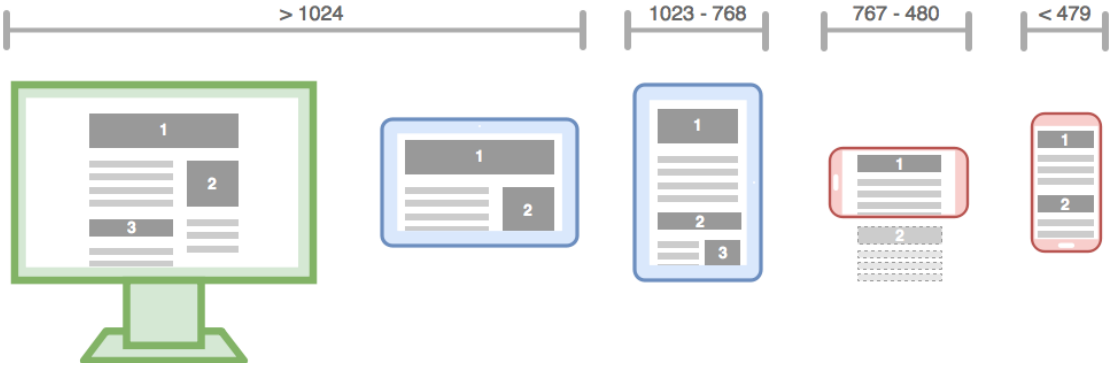

Abbildung 0.2.5: Anordnungsmöglichkeiten durch verschiedene Media Queries nach Pixelbreite

Die Angabe eines Medientyps ist die einfachste Form einer Medienabfrage und gleichzeitig Grundlage für komplexere Abfragen. Ein Medientyp ist ein Schlüsselwort, dem verschiedene gleichartige Ausgabegeräte zugeordnet werden können. Allerdings gibt es keine feste Definition, welche Geräte einem Medientyp zuzuordnen sind *(Tab. 2.2.1)*. Die Beschreibung der einzelnen Medientypen wird dabei ausdrücklich als Richtbeschreibung bezeichnet, da ein Browser nämlich nur einen einzigen Medientyp berücksichtigen darf. Ein Dokument kann also nicht von zwei Stylesheets, welchen unterschiedliche Medientypen zugeordnet wurden, gleichzeitig beeinflusst werden.[50]

Medientyp:	Richtbeschreibung:	Unterstützung:
`all`	alle Ausgabemedien (Standardwert)	- - -
`print`	für Drucker, die Inhalte sichtbar auf Papier drucken	Android, Chrome, Firefox, Internet Explorer, Opera, Safari
`screen`	(Computer-) Bildschirme	Android, Chrome, Firefox, Internet Explorer, Opera, Safari
`speech`	für Sprachsynthesizer, reserviert für ein zukünftiges CSS-Modul (ersetzt den Medientyp `aural`)	CSS2.1, Opera

Tabelle 0.2.1: Übersicht der Medientypen in CSS

„Die Definition der Ausgabemedien und die der Media Queries erfolgt mittels der `@media` Anweisung innerhalb der einzelnen Cascading Stylesheets, welche wiederum andere Styles umfasst. Die öffnende Zeile steht dabei vor dem Style und die schließende geschwungene Klammer danach (Abb. 2.2.6). Die Anweisung `@media` kann in einem Stylesheet mehrmals auftreten, darf aber keinesfalls verschachtelt werden. Bevor das zweite `@media` beginnt, muss das vorherige demnach geschlossen werden."[51]

[49] vgl. *Maurice* (2012), S.115.
[50] s. *SelfHTML Wiki*: CSS/Media Queries, (Stand: 27.09.2016).
[51] *Müller* (2013), S.170 f.

```
@media print {

   /* Ausgabedefinition per Media Query */

}

/* oder via meta-viewport Angabe im head: */

<link rel="stylesheet" href="thesis.css"
    media="screen and (min-width: 960px)">
```

Abbildung 0.2.6: Beschränkung der Styles auf die Druckausgabe per Media Query bzw. `meta-viewport`

0.2.2 Scalable Vector Graphics

Grafiken transportieren Informationen oft viel schneller und exakter als ein geschriebener Text. Sie rufen einen enormen Aufmerksamkeitsgrad, begleitet durch eine erhöhte kognitive Wahrnehmung, außer bei Blinden, hervor und verdoppeln somit – in Kombination mit dem Text – den allgemeinen Informationsfluss. Dass dabei die Speicherfähigkeit im menschlichen Gehirn signifikant angeregt wird und viele Emotionen und Assoziationen dabei geweckt werden, wissen kommerzielle Webseitenbetreiber mittlerweile gut zu nützen.[52] Damit dies im weiteren Verlauf aber auch erfolgen kann, existieren bestimmte Anforderungen an Bilddateiformate innerhalb des Internets, die wie folgt lauten:

- flexible und erweiterbare Dateiformate
- schnellere Verbreitung der Akzeptanz durch Standardisierung
- schnellstmögliche Übertragbarkeit im Internet
- Indizierungsmöglichkeiten für spätere Recherchezwecke[53]

Obwohl Webseiten eigentlich seit Anbeginn weitgehend pixelbasierten Grafiken ausgeliefert waren, wurde mit HTML5 endlich das neue SVG-Format unterstützt, welches man nativ in ein Dokument einbinden kann.[54] Im Grunde genommen ist SVG eine weitere Spezifikation des W3C *(Abb. 2.2.7)*, welche für zweidimensionale Vektorgrafiken, sowie grafische Anwendungen in XML entwickelt wurde. Dadurch, dass SVG ebenfalls eine XML-basierte Sprache darstellt, profitiert sie extrem vom offenen Standard, der Plattformunabhängigkeit und dem enormen Funktionsumfang.[55]

Im Allgemeinen bestehen Vektorgrafiken aus vielen geometrischen Formen, welche mathematisch durch die Koordinaten der Stützpunkte und Art der Linienverbindung (z.B. Kreis, Kurve oder Gerade) definiert sind. Ordinäre Rastergrafiken bestehen beispielsweise nur aus Pixelinformationen, die einerseits einen höheren Speicherplatz aufweisen und vor allem beim Vergrößern Qualitätsverluste mit sich bringen. SVG hingegen nutzt die volle Druckerauflösung und ist sogar für den Bildschirm stets richtig skaliert.[56] Einige weitere Unterschiede dazu sind in der folgenden Tabelle *(Tab. 2.2.2)* angeführt.

[52] vgl. *Hammer* (2008), S.35 ff.
[53] vgl. *Mahle* (2004), S.2 f.
[54] vgl. *Potschien* (2013), S.101.
[55] vgl. *Pomaska* (2005), S.184.
[56] vgl. *ebd.*

Abbildung 0.2.7: Offizielles Scalable Vector Graphics Logo des W3C

Merkmal:	Vektorgrafik:	Pixelgrafik:
Verwendung	exakte Zeichnung	Scans, digitale Fotos
kleinste Informationseinheit	zwei- oder dreidimensionale Objekte in variabler Form	zweidimensionale Pixel in fester Form
Skalierung	stufenlos (ohne Qualitätsverlust)	stufenlos (mit Qualitätsverlust)
Dateigröße	klein bis mittel und variabel (je nach Objektmenge)	groß und fest
Komprimierung	Flash nicht, SVG schon	JPEG, GIF, PNG
Ausgabegröße	beliebig (wird immer neu gerastert)	vorgegeben (bei Erhöhung der Größe erfolgt Pixelwirkung)
Internet / Formate	Flash, SVG, WebCGM, VML	JPEG, GIF, PNG
benötigte Rechenleistung	hoch 3	sehr gering
Eignung für realistische Darstellung	gering (jede Datei wird als Vektor beschrieben)	hoch (jedes Pixel einzeln beschreibbar)

Tabelle 0.2.2: Unterscheidungsmerkmale von Vektor- & Pixelgrafiken [57]

Moderne Grafikprogramme sind mitunter aus diesem Grund bereits in der Lage, Dateien im SVG-Format abzuspeichern. Wenn der Webbrowser dann noch HTML5 unterstützt, können die Vektorgrafiken, wie alle anderen Bildformate auch, mittels `<img>`-Element im Quellcode eingebunden werden. Es ist aber auch möglich, die SVG-Dateien per `<object>`- oder `<embed>`-Tag im Quelltext einzubetten, da sich Vektorgrafiken schlussendlich wie normale Dokumente behandeln lassen. Über die Attribute `"width"` und `"height"` kann außerdem mittels CSS eine beliebige Größe aufgrund der besseren Skalierbarkeit eingestellt werden. Der wohl wichtigste Vorteil einer jeden SVG-Grafik steckt allerdings in der definierten Auszeichnungssprache, ähnlich wie bei HTML-Dateien. Somit kann dieses Dateiformat in einem einfachen Editor bearbeitet werden, wodurch separate Grafikprogramme wie für andere Vektorgrafikformate überflüssig werden.[58]

[57] *Mahle* (2004), S.7.
[58] vgl. *Potschien* (2013), S.101.

Literatur- & Quellenverzeichnis

Ebner, Alexander; Bold, Max (Hrsg.) **(2013):** Webdesign mit CSS3 - Workshops und Praxistipps für das Webdesign mit CSS3; 1. Auflage, Epubli Verlag, Berlin.

Hagelkruys, Dominik (2012): Barrierefreiheit als Thematik im österreichischen Informatikunterricht; Universität Wien, Wien.

Hammer, Norbert (2008): Mediendesign für Studium und Beruf - Grundlagenwissen und Entwurfssystematik in Layout, Typografie und Farbgestaltung; 1. Auflage, Springer Verlag, Berlin.

Heller, Stephan (2012): Workshop HTML5 & CSS3 - Weblayouts professionell umsetzen - ein Einstieg in die Frontendentwicklung; 1. Auflage, Dpunkt Verlag, Heidelberg.

Hogan, Brian P. (2011): HTML5 & CSS3 - Webentwicklung mit den Standards von morgen; übersetzt von: Fröhlich, Stefan; 1. Auflage, O'Reilly Verlag, Köln.

Kadlec, Tim (2013): Praxiswissen Responsive Webdesign; übersetzt von: Fröhlich, Stefan; 1. Auflage, O'Reilly Verlag, Köln.

Mahle, Niko (2004): Das technische und strategische Potential von SVG als standardisierte Grafik-Technologie für grafikorientierte Browser-Anwendungen bei DEKRA; Berufsakademie Stuttgart, Stuttgart.

Maurice, Florence (2012): Mobile Webseiten - Strategien, Techniken, Dos und Don'ts für Webentwickler; 1. Auflage, Carl Hanser Verlag, München.

Müller, Peter (2013): Flexible Boxes - Eine Einführung in moderne Websites; 1. Auflage, Rheinwerk Verlag, Bonn.

Pomaska, Günter (2005): Grundkurs Web-Programmierung - Interaktion, Grafik und Dynamik: Mit XHTML und CSS, XML, JavaScript, Applets, SVG, PHP; 1. Auflage, Vieweg & Sohn Verlag, Wiesbaden.

Potschien, Denis (2013): Pure HTML5 und CSS3; 1. Auflage, Franzis Verlag, München.

Niederst Robbins, Jennifer (2014): HTML5 - kurz & gut; übersetzt von: Lang, Jørgen W.; 5. Auflage, O'Reilly Verlag, Köln.

Allsopp, John: A Dao of Web Design; siehe online unter: http://alistapart.com/article/dao (Stand: 24.09.2016).

Draw.io: Flowchart Maker & Online Diagram Software; siehe online unter: https://www.draw.io (Stand: 24.09.2016).

Kröner, Peter: Die Karte des HTML5-Universums; siehe online unter: http://www.peterkroener.de/die-karte-des-html5-universums (Stand: 25.08.2016).

Marcotte, Ethan: Responsive Web Design; siehe online unter: http://alistapart.com/article/responsive-web-design (Stand: 27.09.2016).

Mozilla Developer Network: Liste der HTML5-Elemente; siehe online unter: https://developer.mozilla.org/de/docs/Web/HTML/HTML5/HTML5_element_list (Stand: 25.09.2016).

SelfHTML Wiki: CSS/Box-Modell; siehe online unter: https://wiki.selfhtml.org/wiki/CSS/Box-Modell (Stand: 26.09.2016).

SelfHTML Wiki: CSS/Media Queries; siehe online unter: https://wiki.selfhtml.org/wiki/CSS/Media_Queries (Stand: 27.09.2016).

The Web Standards Project: Working together for Standards; siehe online unter: http://www.webstandards.org/about/ (Stand: 24.09.2016).

W3C: CSS Current Status; siehe online unter: https://www.w3.org/standards/techs/css#w3c_all (Stand: 26.09.2016).

W3C: HTML5 Recommendation 2014; siehe online unter: https://www.w3.org/TR/html5/ (Stand: 24.09.2016).

W3C: Web Content Accessibility Guidelines (WCAG) 2.0 2009; siehe online unter: https://www.w3.org/Translations/WCAG20-de/ (Stand: 24.09.2016).

Wikiquote: Bruce Lee - Quotes; siehe online unter: https://en.wikiquote.org/wiki/Bruce_Lee (Stand: 27.09.2016).

Abbildungsverzeichnis

Tabellenverzeichnis