

# 1

# Übung Web- und Multimedia-Engineering

Organisatorisches und Allgemeines  
Aufgabenstellung A1 und thematische Einführung

# Inhalte / Gliederung

## 1. Organisatorisches und Allgemeines

- Website, Zugang und Kontakt
- Ablauf und Abgabe
- Überblick Aufgabenstellungen
- Terminplan
- Teambildung und Vorbereitung

## 2. Aufgabenstellung A1: HTML5, CSS3 und JS

## 3. Grundlagen Webentwicklung

- HTML
- CSS
- JS

## 4. Hilfreiches, Tipps und Links

Teil 1

# **Organisatorisches und Allgemeines**

# Die Übung zur Vorlesung

## ■ Zentrale Webseite für die Übung

[https://imld.de/study/teaching/ws\\_17-18/wme\\_17-18/ueb-wme\\_17-18/](https://imld.de/study/teaching/ws_17-18/wme_17-18/ueb-wme_17-18/)

- Allgemeine Informationen, Neuigkeiten und Ankündigungen, Übungsaufgaben, Materialien, Übungsergebnisse

### ZUGANGSDATEN

Benutzername: mtstudent  
Passwort: \*\*\*\*

## ■ Ansprechpartner und Betreuung

-  **Dipl.-Ing. Ricardo Langner**  
Büro: APB/2050  
Kontakt: [ricardo.langner@tu-dresden.de](mailto:ricardo.langner@tu-dresden.de)
-  **Philipp Heisig**  
Kontakt: [philipp.heisig@tu-dresden.de](mailto:philipp.heisig@tu-dresden.de)

# Die Übung zur Vorlesung

- Versteht sich als **Ergänzung** zur Vorlesung
  - Keinesfalls Ersatz, nur zur Übung gehen = „keine gute Idee“
  - Kann nur einige Teilen des Vorlesungsinhalts abdecken
  - Nicht synchron zur Vorlesung (Format-bedingt sehr schwer 
- Verständnis durch **praktische** Aufgaben- und Problemstellungen
- Unterstützung und Hilfestellungen während des Selbststudiums
- Austausch durch Gruppen-/Teamarbeit
- Qualifikation bzw. Zulassung zur Klausur
  - Mindestens 50% der Übungspunkte

# Ablauf der Übung

- Drei Übungstermine je Woche, alle im APB/E065:
  - Montags 2.DS + Montags 3.DS + Dienstags 3.DS
  - Wichtig: Einschreibung per jExam  (ggf. noch nachholen)
- Teamarbeit während des gesamten Semesters möglich
  - Zwei Studierende je Team (Studiengang, Übungstermin, o.ä. egal)
  - Teams bleiben über alle Aufgaben hinweg bestehen
  - Gleiche Bewertung für alle Teammitglieder
  - Unbedingt die Teamnummer aufschreiben
- Übungsbewertung
  - Basiert auf
    - Erfüllung konkreter Teilaufgaben/Anforderungen
    - Quellcode, Kommentare, Fehler, Aufbau und Struktur, Grafik und Design
  - Veröffentlichung als Liste mit allen Teams und deren %te  
(im Passwort-geschützen Bereich der Übungswebseite)

# Ablauf der Übung

- Abgabe der Ergebnisse
  - **Immer** während der Übung (Montag/Dienstag)
  - Hochladen als gepacktes ZIP-Archiv
    - Jedes Team erstellt ein ZIP-Archiv (**kein** RAR) mit Lösung
    - Auf Benennung achten: **Aufgabe<NR>\_Team<T\_NR>.zip**
      - <NR> Nummer der Aufgabe (Übung), 1-2-3-4
      - <T\_NR> Teamnummer, 1-2-3-n
  - Ergebnisabgabe umfasst **immer**
    - alle Quell-Dokumente, eine README.md mit Teamnummer und Namen
    - Denken Sie auch an Materialien, wie z.B. Bilder, Datensätze
  - Nachreichungen werden **nicht** akzeptiert!
    - 0 Punkte bei verpasster Abgabe
    - Abgabe verpassen ist „keine gute Idee“, es gibt nur 4 Aufgaben

# Ablauf der Übung

- Präsentation der Musterlösung
  - Optional immer eine Woche nach der Abgabe
  - Je nach Bedarf und Interesse: Vorbereiten (Fragen/Probleme)
  - Als erster Teil der Konsultation
- Diskussion und Austausch: Auditorium als Plattform
  - Jeder sollte Account haben: <https://auditorium.inf.tu-dresden.de/de/groups/110408018>



The screenshot shows the Auditorium platform interface. At the top, there is a navigation bar with icons for help, groups, my groups, help, and a search bar. Below the navigation bar, the main content area displays a group page for "Web & Multimedia Engineering". The page title is "Web & Multimedia Engineering" with a "Ich folge der Gruppe" button. A descriptive text block explains the purpose of the course, mentioning it covers methods and tools for designing and implementing distributed web applications. It also lists two links: "zur Vorlesungswebsite" and "zur Übungswebsite". Below this, there is a section with tags related to the course. At the bottom of the page, there are buttons for "Gruppenmitglieder bearbeiten", "Gruppe bearbeiten", and "Gruppe löschen".

## Ankündigungen

# Ablauf der Übung

- Ein Themenkomplex, vier Aufgabenstellungen

- Jeweils unterschiedlich lange Bearbeitungszeit
  - Gemeinsames Thema über alle Übungen



world\_data

- **A1:** Grundlagen client-seitige Technologien: HTML5, CSS3 & JS
    - Grundgerüst für eine Website als Interface für world\_data
  - **A2:** Grundlagen server-seitige Technologien: XML und PHP
    - CSV sowie XML Transformation, Grundlagen PHP Serverkomponente
  - **A3:** Erweiterung server-seitige Technologien: Node.js & AJAX
    - Erstellung eines REST-Services, Abfragen durch den Client via AJAX
  - **A4:** Anwendungsfall – Visualisierung mit D3.js & Leaflet
    - Visualisierung von Daten mittels interaktiver Bar Charts und Karten

Woche	Datum	Übungsthema /-inhalt	Folien	Materialien
KW 41	9./10.10.	keine Übung (erste Vorlesungswoche)		
KW 42	16./17.10.	Einführung Aufgabenstellung A1: HTML + CSS + JS		<ul style="list-style-type: none"> <li>■ Screenshots</li> <li>■ Logo</li> <li>■ Daten (CSV)</li> </ul>
KW 43	23./24.10.	Konsultation		
KW 44	30.10.	Mo. Konsultation, Di. 31.10. keine Übung (Reformationstag)		
KW 45	6./7.11.	<b>Abgabe A1</b>		
KW 45	6./7.11.	Einführung Aufgabenstellung A2: PHP + XML		Materialien für A2
KW 46	13./14.11.	Lösung A1 und Konsultation		
KW 47	20./21.11.	<b>Abgabe A2</b>		
KW 47	20./21.11.	Einführung Aufgabenstellung A3: Webservices Node.js + AJAX		Materialien für A3
KW 48	27./28.11.	Lösung A2 und Konsultation		
KW 49	4./5.12.	Konsultation		
KW 50	11./12.12.	<b>Abgabe A3</b>		
KW 50	11./12.12.	Einführung Aufgabenstellung A4: Anwendungsbeispiel		Materialien für A4
KW 51	18./19.12.	Lösung A3 und Konsultation		
KW 52	25./26.12.	Jahreswechsel (keine Übung)		
KW 1	1./2.1.	Jahreswechsel (keine Übung)		
KW 2	8./9.1.	Konsultation		
KW 3	15./16.01.	Konsultation		
KW 4	22./23.01.	<b>Abgabe A4</b>		
KW 4	22./23.01.	Abschluss, Feedback und Fragen		
KW 5	29./30.01.	Lösung A4 und offene Fragen (nur nach Anmeldung)		
Woche	Datum	Übungsthema /-inhalt	Folien	Materialien

[https://imld.de/study/teaching/ws\\_17-18/wme\\_17-18/ueb-wme\\_17-18/](https://imld.de/study/teaching/ws_17-18/wme_17-18/ueb-wme_17-18/)

## Teambildung

(je 2 Studierende)

Notieren Sie ihre Teamnummer 

# Ablauf der Übung

- Eintrag in Mailingliste
  - Sie werden automatisch eingetragen!

**Abonnieren von Mt-wme**

Abonnieren Sie Mt-wme, indem Sie das folgende Formular ausfüllen:

In Kürze erhalten Sie eine Bestätigungs-e-Mail, um sicherzustellen, dass es wirklich Sie sind, der abonnieren möchte. Administrator der Liste eingesehen werden kann.

Ihre e-Mailadresse:

Ihr Name (optional):

Sie können weiter unten ein Passwort eingeben. Dieses Passwort bietet nur eine geringe Sicherheit, sollte aber verhindern, dass andere **Verwenden Sie kein wertvolles Passwort**, da es ab und zu im Klartext an Sie geschickt wird!

Wenn Sie kein Passwort eingeben, wird für Sie ein Zufallspasswort generiert und Ihnen zugeschickt, sobald Sie Ihr Abonnement bestätigt haben. Passwort jederzeit per Email zuschicken lassen, wenn Sie weiter unten die Seite zum ändern Ihrer persönlichen Einstellungen aufrufen.

Wählen Sie ein Passwort:

Erneute Eingabe zur Bestätigung:

Welche Sprache bevorzugen Sie zur Benutzerführung?

Teil 2

## Aufgabenstellung A1

### HTML5, CSS3 und JS



world\_data

# Aufgabenstellung A1

- Einfaches Webinterface zur Anzeige der Daten, fünf Komponenten:
  - Navigationsleiste (Menü im Header), einfache Text-Box,  
Tabellenansicht der Daten, Anzeigeoptionen, Fußbereich (Footer)
- Statische Webseite ohne spezielle Funktionen: Grundgerüst
- Dummy-Daten, grundlegend übertragen (oder einlesen)
- Testen: Firefox (aktuelle Version)
- Abgabe: Montag/Dienstag, **6.11.2016 & 7.11.2016** 

# Aufgabenstellung A1



- Erstellen (Rekonstruieren) Sie das folgende Webinterface

## World Data Overview ...

- Materialien: Screenshots, Logo, Daten (CSV-Datei)

# Aufgabenstellung A1: Bewertungskriterien

- Allgemeine Kriterien und Verwendung
  - Nur HTML, CSS und JavaScript
  - Ausgewählte Bibliotheken, Frameworks o.ä. einbinden:
    - Font Awesome (>= 4.4), <http://fontawesome.io/>
    - CSS-Reset, <https://github.com/murtaugh/HTML5-Reset/blob/master/assets/css/reset.css>
    - Webfont Roboto, <https://fonts.google.com/specimen/Roboto>
  - Dateikodierung/Encoding: **UTF-8 (und Zeilenende Unix-LF)**
  - Valides HTML5 und CSS3
  - HTML Header-Angaben: mind. title, description, author, keywords
  - HTML5-Elemente: header, footer, nav
  - Fußzeile (Footer): Namen und Teamnummer

# Aufgabenstellung A1: Bewertungskriterien

- Design
  - Basisschrift: Roboto, 14px, Zeilenabstand: 145%, Style: Normal
  - Iconschrift "Font Awesome" für Nav und Spaltensorierung
  - Hover für Standard-Links: Farbe und Unterstrich
  - Responsive Design: „mobile-first“ Ansatz mit mindestens drei expliziten, sinnvollen Schwellwerten, bspw.:
    - >= 320 Phones; >= 768 Tablets; >= 1200 Desktops:  
Container-Breite (Seiten-Wrapper) 1170px, links-rechts-zentriert
    - Anpassen der Spaltenanzahl für Text-Box über der Tabelle (3/2/1-spaltig)

# Aufgabenstellung A1: Bewertungskriterien

- Design
  - Header mit Logo und sechs Menüelemente/Links
    - Farbgradient im Header (*Hinweis: geht auch ohne Bild*)
    - „Klickbares“ Logo mit Hover (farblos/grau)
    - Liste (Unordered List, ul) für Menüelemente
    - „Font Awesome“ Icon für Menüelemente
    - Hover der Menüelemente: Kurze Animation für Hintergrundfarbe im Header (CSS-Transition), siehe Live-Demo
    - Bleibt beim Scrollen am oberen Bildschirmrand (Sticky Navigation)

# Aufgabenstellung A1: Bewertungskriterien

- Design
  - Tabelle für Datenanzeige
    - Elemente thead und tbody einsetzen
    - Mind. folgende Spalten: ID und Country, sowie fünf Attribute
    - Mind. 25 Länder (ID und Country) mit beliebigen Daten (Zufall, CSV, ...)
    - „Show/Hide“ Funktionalität für fünf Attributspalten
    - Sortierung der Tabelle (auf- und absteigend) nach Country
    - „Font Awesome“ Icons in Spalte Country
    - Responsive Design: Auf Basis der CSS-Schwellwerte (mindestens drei) automatisch die jeweils letzte Spalte ausblenden
- Empfohlene Zusatzaufgaben (ohne Bewertung)
  - Daten dynamisch aus CSV einlesen
  - Responsive Design für Header/Navigation:
    - Menüelemente zusammenfassen/einklappen  
(Hinweise u.a. <http://www.hongkiat.com/blog/responsive-web-nav/>)

Teil 3

## **Grundlagen Webentwicklung:** Eine thematische Einführung

## ■ HTML5 – Grundstruktur

- ```
<!DOCTYPE html>
<html>
  <head>
    <!-- ... -->
  </head>
  <body>
    <!-- ... -->
  </body>
</html>
```



- **head** enthält Seitentitel, CSS- und JS-Referenzen, etc.
- **body** enthält sichtbare Seitenelemente: Überschriften, Texte, Bilder, Tabellen, etc.

## ■ HTML – Basics

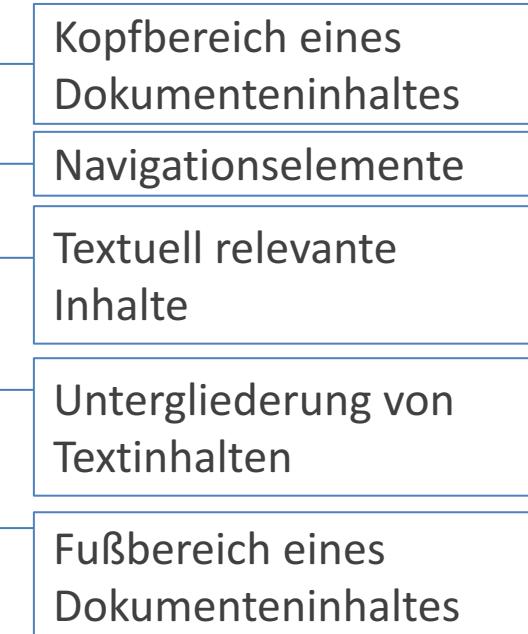
- <h1>Überschrift ersten Grades</h1>
- 
- <p>Ich bin ein Absatz.</p>
- Bilder, Skripte etc. können **absolut** oder **relativ** adressiert werden
  - Start ohne Slash, z.B. „bilder/logo.jpg“ ist relativ zum aktuellen Verzeichnis
  - Start mit Slash, z.B. „/css/layout.css“ ist relativ zum Wurzelordner des Servers
  - Start mit http://, z.B. „http://mt.inf.tu-dresden.de/lehre/wme“ ist absolut
- **Semantische Tags** in HTML5 für SEO
  - Struktur: <section>, <article>, <nav>, ...
  - Textsemantik: <time>, <meter>, <progress>, ...



## ■ HTML5 – Strukturtags

- Statt wie bisher mit <div>-Tags werden Dokumente nun mittels neuer Tags strukturiert → verbessert die Maschinenlesbarkeit und macht den Code übersichtlicher

```
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <header><h1>HTML5 Demo</h1>
      <nav> ... </nav>
    </header>
    <article>
      <h2>Semantische Tags</h2>
      <section> ... </section>
    </article>
    <footer> ... </footer>
  </body>
</html>
```



## ■ HTML – Spezielle Elemente

- Listen können durch `<ul>` (unordered) oder `<ol>` (ordered) definiert werden. Die Kind-Elemente werden durch `<li></li>` definiert.
- Tabellen werden durch `<table>` definiert. Sie bestehen aus `<thead>` und `<tbody>` sowie Reihen:

```
<tr>
  <td></td>
</tr>
```

- Alle Elemente können zusätzliche Informationen über die HTML5 `data-*` Attribute erhalten:

```
<ul>
  <li data-animal-type="bird">Owl</li>
  <li data-animal-type="fish"></li>
</ul>
```



## ■ Cascading Style Sheets (CSS)

- Regeln zur Darstellung eines HTML-Dokuments
- Farben, Rahmen, Abstände, Schriften, ...
- bestehend aus
  - Selektor
  - Eigenschaft-Wert-Paaren
- Syntax
  - Selektor {Eigenschaft:Wert; Eigenschaft:Wert;...}
  - z.B.: p{font-family:"Times New Roman"; font-size:20px;}  
passt Schriftart und -größe aller Absätze an
- *class* und *id* um HTML-Elementen Eigenschaften zuzuordnen
  - class als Selektor für Vielzahl von Elementen zb: <p class='excersise'>
  - id für einzelnes selektieren von Elementen <nav id='main\_menu'>



## ■ Cascading Style Sheets (CSS)

- Einbinden von CSS

- direkt im Quellcode

```
<h1 style=„color:red;“>Webdesign für Anfänger</h1>
```

- am Anfang der HTML-Datei

```
<style>
    p.subtitle { font-size:12pt;color:#0000FF; }
    a:hover     { text-decoration:none; }
</style>
```

- als externe Datei

```
<link rel=„stylesheet“ href=„/css/layout.css“>
```



## ■ Cascading Style Sheets (CSS)

- Selektor gibt zu formatierende Elemente an:

- Global: \* (alle Elemente des Dokuments)
- Elementtyp: h1 (alle Überschriften 1. Grades)
- Klasse: .subtitle (alle Elemente mit class-Attribut „subtitle“)
- ID: #navigation (Element mit der ID „navigation“)
- Pseudo-Klasse: a:hover (beim Überfahren des Elements mit der Maus)
- Liste: h1, img, p (alle 1. Überschriften, Bilder und Absätze)
- Unterelemente: p a (alle Links innerhalb von Absätzen)

- nahezu beliebig kombinierbar
- spezifischere Regeln überschreiben allgemeinere
  - Klasse ist spezifischer als Element
  - ID ist spezifischer als Klasse
  - betrifft jedoch nur **widersprüchliche** Attribut-Wert-Paare!



## ■ CSS – Weitere Selektoren

- CSS-Versionen definieren neue oder erweiterte Selektortypen:  
Attribut-Selektoren, Pseudo-Klassen, Kombinatoren
- **Attribut-Selektoren:** erlauben Selektion von Elementen unter Prämissen der id-Benennung



<div id="nav-primary"/>	div[id^="nav"] { background:#ff0; }	Alle Elemente die mit „nav“ beginnen
<div id=„second-nav“/>	div[id\$="nav"] { background:#ff0; }	Alle Elemente die mit „nav“ enden
<div id=„supernavone“/>	div[id*="nav"] { background:#ff0; }	Alle Elemente die „nav“ enthalten

- **Strukturelle Pseudo-Klassen:** erlaubt die Selektion von Elementen abhängig von ihrer Position im DOM-Baum:
  - root, nth-child, first-child, last-child, only-child,only-of-type,empty, ...

## ■ CSS – Weitere Selektoren

- nth-child() – Funktion kann nach vielfältigen Formeln Kindelemente selektieren

<code>:nth-child(3) { color:#f00; }</code>	Selektiert das dritte Kindelement
<code>:nth-child(odd) { color:#f00; }</code>	Selektiert alle ungeraden Kindelemente (even für gerade)
<code>:nth-child(2n+7) {color:#f00; }</code>	Selektiert alle Elemente die der Formel $a + b$ entsprechen (a - Zyklusgröße, n - Zählvariable (ab 0), b - Offset)

- **Beispiel:** abwechselnde Farben der Tabelle hier?
- `tr:nth-child(2n) { background-color: #DOD8E8; }`  
`tr:nth-child(2n+1) { background-color: #E9EDF4; }`



## ■ CSS – Kombination von Selektoren [8]



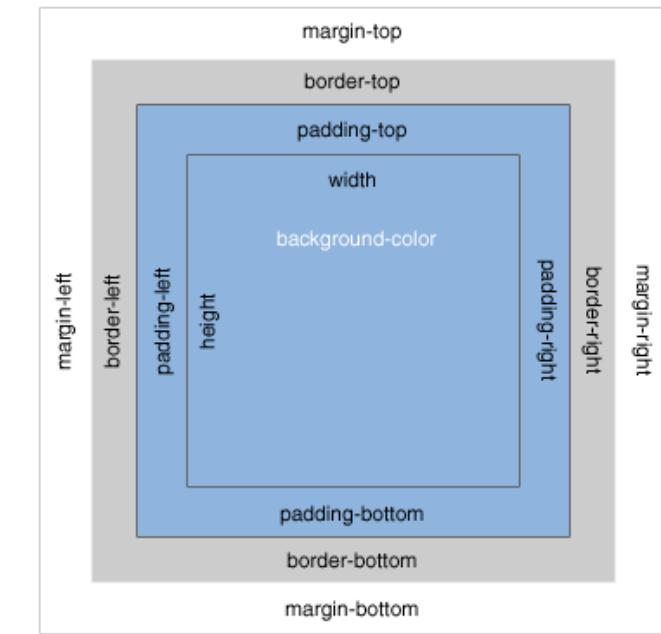
X > Y	Selektiert <u>nur</u> die <u>direkten</u> Kinder Y von X	div#container > ul { border: 1px solid blue; }
X + Y	Selektiert <u>nur</u> das Y-Element nach einem X	ul + p { color: blue; }
X ~ Y	Selektiert <u>alle</u> Y nach einem X (weniger strikt als X + Y)	ul ~ p { color: blue; }

### – Beispiel:

```
<div id="container">
    <ul> ←
        <li> List Item
            <ul>
                <li> Child </li>
            </ul>
        </li>
        <li> List Item </li>
    </ul>
    <p> Paragraph </p> ←
    <p> Paragraph </p> ←
</div>
```

## ■ Cascading Style Sheets (CSS)

- Stil gibt an, wie Elemente formatiert werden sollen
  - Rahmen und Abstände: **CSS Box Model**
  - Positionierung: immer relativ zum Elternelement
    - absolute, relative, inline, fixed
  - Farben
    - als RGB-Hex-String: #**FFFFFF**
    - als RGB-Tupel: **rgb(0,0,0)**
    - Weitere: **rgba**, **hsl**, **hsla**
  - Transparenzen: **opacity: 0.2;**
  - Schrift
    - Größe: **font-size: 12pt;**
    - Stil: **font-style: italic;**
    - Stärke: **font-weight: bold;**
    - Extras: **text-decoration: underline;**

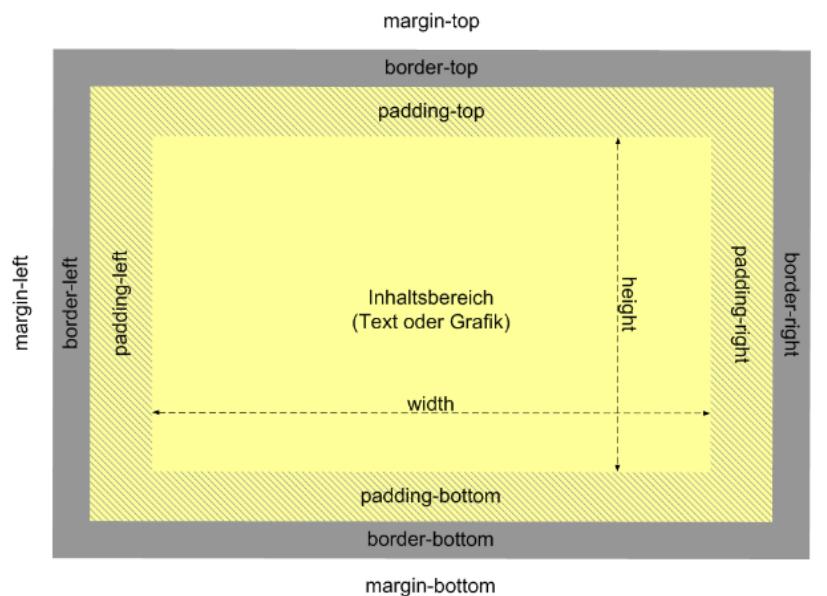


Quelle: [\[http://www.mandalatv.net/itp/drivebys/css/\]](http://www.mandalatv.net/itp/drivebys/css/)

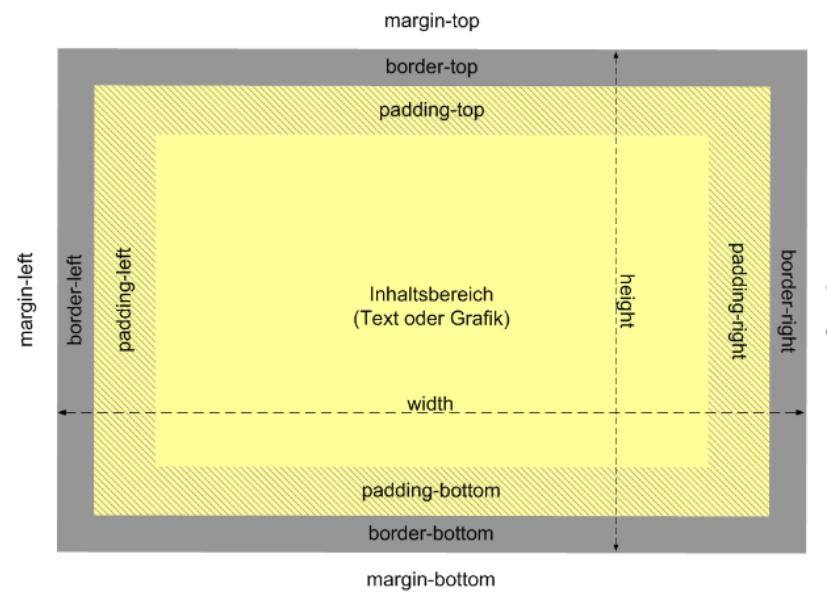


## ■ CSS3 – Neues Box-Modell [9]

- Bisher: {box-sizing: content-box} → Höhe und Breite definieren Inhaltsbereich; Padding, Border, Margin werden hinzugefügt
- Neu: {box-sizing: border-box} → alle Außenelemente sind in Höhe und Breite enthalten



Klassisches Box-Modell für Block-Elemente – {box-sizing: content-box} (Standardeinstellung) – <http://little-boxes.de/>



CSS3-Box-Modell für Block-Elemente – {box-sizing: border-box} – <http://little-boxes.de/>

## ■ Cascading Style Sheets (CSS)

- **Media Types** geben Zielmedien für Regeln an
  - @media print, handheld, projection, tv, speech, braille
- z.B.:

```
@media handheld,print {  
    p.subtitle{color:#000000; }  
}
```
- **CSS 3 Media Queries** als Erweiterung
  - ermöglichen Adaptivität der Darstellung an Bildschirmauflösung etc.
  - UND-verknüpft
  - (max-/min-)width, color, aspect-ratio, ...
- z.B.:

```
@media screen and (max-width: 500px) {  
    p.subtitle { font-size: 14pt; }  
}
```



## ■ Java Script (JS)

- Scriptsprache für Einsatz im Browser und Server
- Typische **Anwendungsbereiche** von JavaScript:
  - Überprüfung von Formularen vor dem Absenden
  - Dynamische **Interaktion**, z. B. Buttons, Darstellungsgröße, ...
  - Dynamische **Inhalte**, z. B.: AJAX, WebSockets, ...
  - **Veränderung** bzw. Animation von Eigenschaften der HTML-Elemente
- **Leistungsmerkmale:**
  - Zugriff auf Bestandteile des geladenen Dokumentes über DOM
  - Dynamische Änderung des geladenen Dokumentes
  - Steuerung externer Komponenten
  - Reaktion auf Benutzerinteraktionen, Fenstermanagement
  - Bei Serverseitiger Anwendung: Erstellen von komplexen Programmen, Schnittstellen etc. ...



## ■ Einbindung

- Innerhalb eines HTML-Elements

```

```



- Innerhalb eines HTML-Dokuments in einem <script>-Teil

```
<script language="javascript">  
  //...Hier kommt der Code ...  
</script>
```

- Innerhalb einer externen Datei

```
<script src="menue.js" type="text/javascript"></script>
```



- **Funktionen**

```
function Fkt.name (Parameterliste) {Funktionsrumpf}
```

- Rückgabewert kann hinter Schlüsselwort return angegeben werden

- Beispiel: `function square(i) {... return i*i; }`

- **Operatoren** entsprechen im Wesentlichen denen von Java

- Zuweisung:

- =

- Vergleichsoperatoren:

- `==, !=, <, <=, >, >=, ===`

- Rechenoperatoren:

- `+, -, *, /`

- logische Operatoren

- `&&, ||`

- **Verzweigungen und Schleifen** ebenso (Beispiele)

- `if (...) {  
...}  
else {  
...}`

- `while (...) {  
...}  
for (...) {  
...}`

## ■ Objekte

- zum Zugriff auf Elemente des HTML-Dokuments
- Zugriff auf Objekt-Attribute & –Methoden über den **Auswahloperator ". "**  
z. B.: `navigator.geolocation.getCurrentPosition(succ,err);`  
zum Zugriff auf den aktuellen Standort des Benutzers

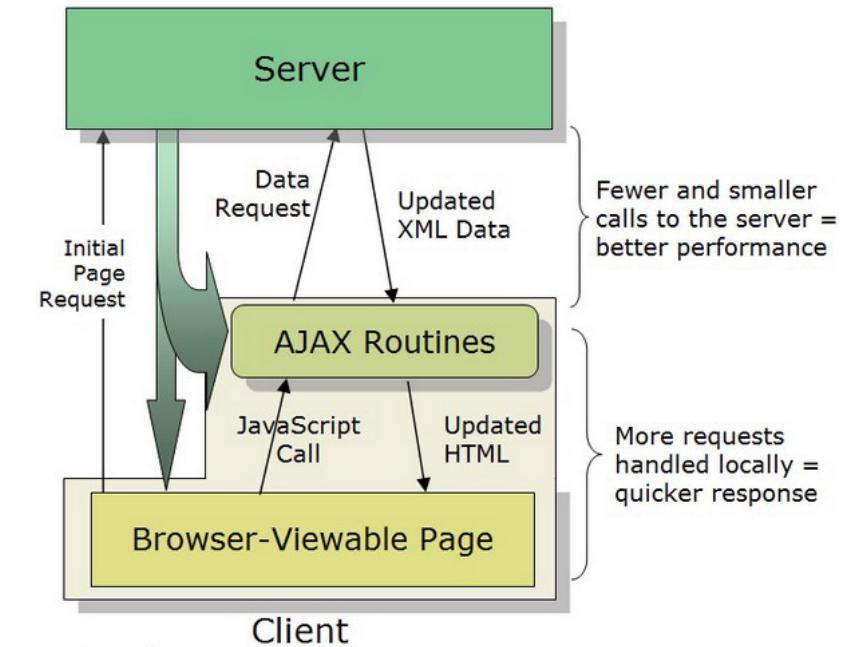
```
var cup = { color: `green` };
Console.log(cup.color) //green
```

- JavaScript enthält **vorgefertigter Objekttypen**: Sprach-Objekte und Client-Objekte
  - **Sprach-Objekte**: Repräsentation häufig genannter Datentypen,  
z. B. **Array**, **Math** (math. Grundfkt.), **Date**, **Function** oder **RegExp** (reguläre Ausdrücke)
  - **Client-Objekte**: Zugriff auf Teile des **HTML-Dokumentes** und Zustand dieser



## ■ AJAX – Asynchronous JavaScript and XML

- Hauptproblem bei komfortablen Web-Anwendungen ist träge Reaktion
  - Je Benutzerinteraktion eine Serveranfrage, auf Ergebnis muss gewartet werden („blockierende Interaktion“)
- Kernidee AJAX
  - Webseite pro Interaktion nicht neu geladen, sondern dynamisch modifiziert
  - Server-Anfragen asynchron im Hintergrund, Warten meist nicht notwendig
- Bausteine: HTML, JavaScript
  - Dynamische Modifikation des Dokumentenbaums über DOM (Document Object Model) des (X)HTML-Dokuments
  - Asynchrone Datenübertragung mittels XMLHttpRequest

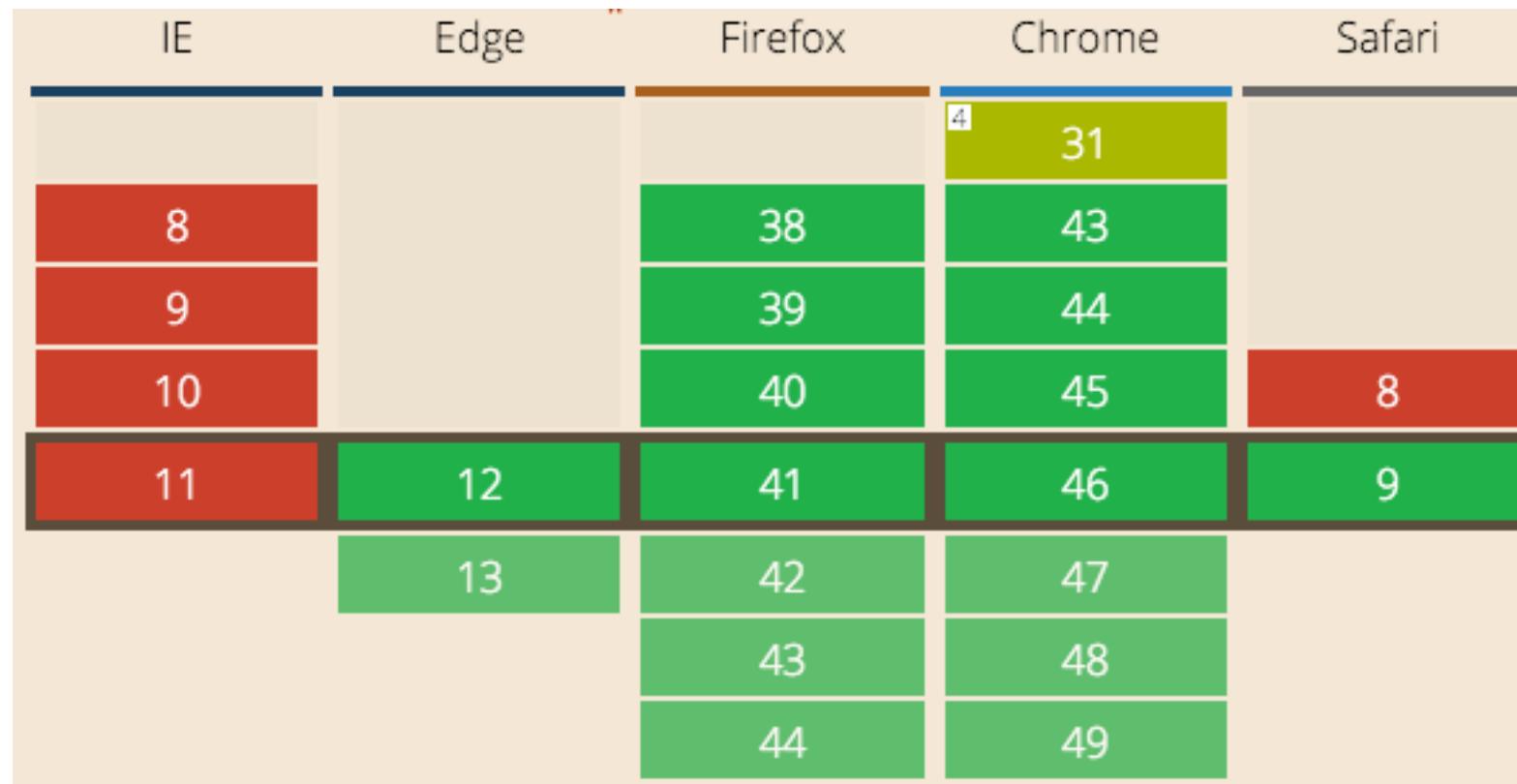


Bildquelle:

[[http://www.codeproject.com/KB/showcase/FarPointAJAX/AJAX\\_process.jpg](http://www.codeproject.com/KB/showcase/FarPointAJAX/AJAX_process.jpg)]

## ■ Browser-Support

- Nicht alle HTML5, CSS3 & JS Elemente werden von allen Browsern unterstützt
  - Empfehlung: [\[http://caniuse.com/\]](http://caniuse.com/)



Teil 4

## **Hilfreiches, Tipps und Links**

## Hilfreiches, Tipps und Links

- Klassisches Thema für Selbststudium:  
„Step-by-Step learning“
- Erstellung eines Farb- und Designschemas mittels Paper-Prototyping [1, 2]
- Hilfreich bei der Gestaltung sind Rastervorlagen [3]
- Umsetzung des Entwurfs in einem Editor der Wahl
  - Liste von HTML-Editoren [10] und Universelle Texteditoren [11]
- Validierung des kompletten Markups
  - HTML5 [4] und CSS3 [5]

# Hilfreiches, Tipps und Links

- Responsive Webdesign [6]
  - Optimale Ansicht einer Website auf allen Arten von Geräten
  - Früher drei Versionen vs. Heute eine flexible Version
  - Einfache Navigation, Informationsfindung mit möglichst wenig Veränderungen an der Seite selbst (Resizing, Panning, Scrolling)
  - Hauptbestandteile: flexible Schrift, flexibles Grid, Media Queries



[7]

Quelle:  
[[http://en.wikipedia.org/wiki/File:Boston\\_Globe\\_responsive\\_website.jpg](http://en.wikipedia.org/wiki/File:Boston_Globe_responsive_website.jpg)]

# Hilfreiches, Tipps und Links

---

- JavaScript
  - Einfacher Einstieg und Überblick [12]
  - Neue Features [13]
  - Vanilla JS ;) [14]

# Quellen, Referenzen

- [1] „Paper Prototyping“, [[http://de.wikipedia.org/wiki/Paper\\_Prototyping](http://de.wikipedia.org/wiki/Paper_Prototyping)]
- [2] „Six Signs That You Should Use Paper Prototyping“,  
[<http://today.java.net/pub/a/today/2003/12/23/sixSigns.html>]
- [3] „Free Printable Sketching, Wireframing and Note-Taking PDF Templates“,  
[<http://www.smashingmagazine.com/2010/03/29/free-printable-sketching-wireframing-and-note-taking-pdf-templates/>]
- [4] „W3C Markup Validation Service“, [<http://validator.w3.org/>]
- [5] „W3C CSS Validation Service“, [<http://jigsaw.w3.org/css-validator/>]
- [6] „Responsive Webdesign mit HTML5 und CSS3 – Grundlagen“,  
[<http://t3n.de/news/responsive-webdesign-html5-css3-grundlagen-335305/>]
- [7] „Media Queries: Very Good Examples“, [<http://mediaqueri.es/>]
- [8] „The 30 CSS Selectors you Must Memorize“,  
[<http://net.tutsplus.com/tutorials/html-css-techniques/the-30-css-selectors-you-must-memorize/>]
- [9] CSS3: Das etwas andere Box-Modell mit {box-sizing:border-box},  
[<http://www.drweb.de/magazin/css3-das-etwas-andere-box-modell-mit-box-sizingborder-box-39078/>]

## Quellen, Referenzen (2)

- [10] Liste von HTML-Editoren, [[http://de.wikipedia.org/wiki/Liste\\_von\\_HTML-Editoren](http://de.wikipedia.org/wiki/Liste_von_HTML-Editoren)]
- [11] Liste von Texteditoren, [[http://de.wikipedia.org/wiki/Liste\\_von\\_Texteditoren](http://de.wikipedia.org/wiki/Liste_von_Texteditoren)]
- [12] Einfacher Einstieg & Überblick JS, [<http://www.w3schools.com/js/>]
- [13] Neue Features JS, [<http://es6-features.org/>]
- [14] Vanilla JS, [<http://vanilla-js.com/>]

# Fragen?



INTERACTIVE MEDIA LAB  
DRESDEN

**Interactive Media Lab Dresden**  
Professur für Multimedia-Technologie

*Kontakt:*

Ricardo Langner ([ricardo.langner@tu-dresden.de](mailto:ricardo.langner@tu-dresden.de))  
Philipp Heisig ([philipp.heisig@tu-dresden.de](mailto:philipp.heisig@tu-dresden.de))

# Changelog

Datum / Zeit	Beschreibung
2017-10-17 9:00	▪ Initiale Downloadversion