WEB-TECHNOLOGIEN

Florian Thürkow
florian.thuerkow@gmail.com

BERUFSAKADEMIE SACHSEN
STAATLICHE STUDIENAKADEMIE
LEIPZIG
UNIVERSITY OF COOPERATIVE EDUCATION

Aufbau der Vorlesung

- Intro
- HTML, CSS und JavaScript – eine Einführung
- Serverkommunikation, Domains, Standards
- Cascading Style Sheets (CSS) / CSS3
- JavaScript und diverse Frameworks
- DOM-Manipulation
- Objekte und JSON
- Cookies und WebStorage
- Ajax
- Responsive Webdesign und mobile Endgeräte

Interessantes

- Voraussetzungen
  - *sicherer Umgang Editor u. Browser*
  - *Texteditor*
  - *Web-Browser*
- Klausur
  - *60 Min. Zeit*
  - *1/3 der Gesamtnote CPWT*
  - *Praktische Prüfung am Computer*
  - *Webseiten-Elemente erstellen*
  - *Code ergänzen / ändern*

Literatur

- Stefan Münz. HTML Handbuch. Franzis, 2005
- css4you.de
- Eric Meyer. CSS: The Definitive Guide. O'Reilly, 2006
- David Flanagan. JavaScript Definitive Guide. O'Reilly, 1998
- Dave Crane. Ajax in Action. Das Entwicklerbuch für das Web 2.0. Addison-Wesley, 2006

HTML, CSS und JavaScript

- HTML
  - *Struktur und Inhalt*
  - *Überschriften, Absätze*
  - *Texte schreiben wir im LibreOffice/Word*
- CSS
  - *Cascading Style Sheets*
  - *Sorgt dafür, dass alles *gut* aussieht*
- JavaScript
  - *Wenn es funktional werden soll*
  - *Rechnen, bewegen, abfragen*

Die Webbrowser

Das erste Dokument

- Aufgabe 1
  - *Starten Sie den Editor*
  - *Erstellen Sie eine neue Datei mit der Endung .htm oder .html*

Achten Sie auf <, > und /

- Aufgabe 2
  - *Tippen Sie diesen HTML-Code ein*

```
<!DOCTYPE html>
<html>
<!-- Meine erste Webseite -->
<head>
    <title>Meine erste
           Webseite</title>
</head>
<body>
    <p>Hallo Web!</p>
</body>
</html>
```

Markup und Tags

- Alles, was zwischen < und > steht, ist ein **Tag** (<html>, , ...)
- Öffnende Tags (<html>) und schließende Tags (, </html>) werden mit einem / gekennzeichnet.
- Ein Tag markiert Text
- HTML: Hypertext Markup Language

<title> Hier steht der Seitentitel</title>

Öffnendes Tag Tag-Body Schließendes Tag

Tags verschachteln

- Tags müssen ineinander verschachtelt werden
- Tags dürfen beliebig tief verschachtelt werden
- Nicht jedes Tag darf an jeder Stelle stehen (<head> innerhalb von <body>: NO!)
- Falsch verschachtelte Tags verursachen keine Fehlermeldungen!

```
<p>
Text und mehr
<span> noch mehr Text </p>
</span>
```

?

Korrekt verschachteltes HTML nennt man wohlgeformt (well-formed)

Struktur einer HTML-Seite

```
<!DOCTYPE html>
<html>
<!-- Meine erste Webseite -->
<head>
    <title>Meine erste
          Webseite</title>
</head>
<body>
    <p>Hallo Web!</p>
</body>
</html>
```

Doctype-Deklaration: Hinweis an den Browser
„Ab jetzt kommt HTML!“

Root-Tag (<html> ist ein MUSS und davor darf nur der Doctype stehen

Kommentare (<!- - Bla bla --> werden vom Browser ignoriert. Ein guter Ort für (Formular-) Passwörter?

Head-Bereich: alles, was mit der Seite zu tun hat, aber nicht zum Inhalt der Seite gehört.

Im Body steht der Inhalt der Seite.

<p> steht für Paragraph, also Textabsatz

Nach </html> darf nichts mehr kommen.

Überschriften

Aufgabe:
Fügen Sie über dem Textabsatz Überschriften der Ebene 1 und 3 ein.

- HTML unterstützt sechs Ebenen von Überschriften
- <h1> bis <h6>

Leseprogramme für Blinde verlassen sich auf die „richtige“ Abfolge der Überschriften.
Die Größe wird später per CSS festgelegt!

Lorem Ipsum

Aufgabe:
Fügen Sie mehrere Absätze mit „Lorem ipsum“ in Ihr HTML-Dokument ein.

- Schnell und ohne nachzudenken: https://loremipsum.de/
- Wird häufig von Webdesignern genutzt
- Das Gehirn versucht nicht den Text zu verstehen und hat die volle Konzentration auf das Layout

Links

- Werden mit <a> gesetzt.
- Neu: Attribute
- Attribute werden in ein öffnendes Tag geschrieben
- Attribute haben Namen und (meistens) einen Wert

Link:
Link nach unten

Ziel:
<a id="unten" />

Aufgabe:
Fügen Sie einen Link in den ersten Absatz Ihrers HTML-Dokumentes ein.

Fügen Sie das Sprungziel an das Ende Ihres HTML-Dokumentes ein.

Aufgabe2:
Fügen einen weiteren Link um von "unten" wieder nach „oben" zu springen.

Das href beginnt mit einem Hash (#), die id nicht.

Links 2

- Bei einem Link auf eine andere Seite wird kein Hash genutzt
- Relativer Link: relativ zur Datei

- Link in einem neuen Fenster öffnen:
  - *Target="_blank" als Attribut für das <a>-Tag*

Aufgabe:
Erstellen Sie eine zweite HTML-Datei (im selben Ordner!) und verlinken Sie die beiden Seiten.

Aufgabe 2:
Verlinken Sie zu einer Webseite Ihrer Wahl.

Aufgabe 3 (schwer):
Füge in die zweite HTML-Datei einen Link ein, der in die erste HTML-Datei direkt ans untere Ende führt.

Bilder

Aufgabe:
Fügen Sie ein Bild in Ihren bestehenden HTML-Code ein.

- <img>-Tag
- Bilder werden über das Attribut src (relativ oder absolut) angegeben
- Ein void-Element (hat keinen Tag-Body oder das / kann weggelassen werden)
- Das Attribut alt enthält eine kurze und aussagekräftige Beschreibung des Bildes
- Die Attribute width und height gibt die Breite und Höhe an (Platz wird „vorreserviert“)
- Beispiel:

-

Immer komprimierte Bildformate nutzen!

Metadaten

- Werden im <head>-Bereich definiert
- Geben Informationen über das Dokument an
- Werden von Suchmaschinen ausgewertet (!!)
- Character Encoding (<meta charset="utf-8">
- Dokument muss in einem Unicode Encoding gespeichert sein
- UTF-8 ist das verbreitste Unicode-Encoding

<meta name="author" content="" />

Fehlersuche

```
<html>
    <head>
        <meta charset=“utf-23“ />
        <title>Fehlersuche<title>
    </head>
    <body>
    <h1>Wer findet 6 Fehler?
    <p>
        <a href=“anderswo.html“>Andere Seite</a target=“_blank“>
    </body>
    </p>
</html>
```

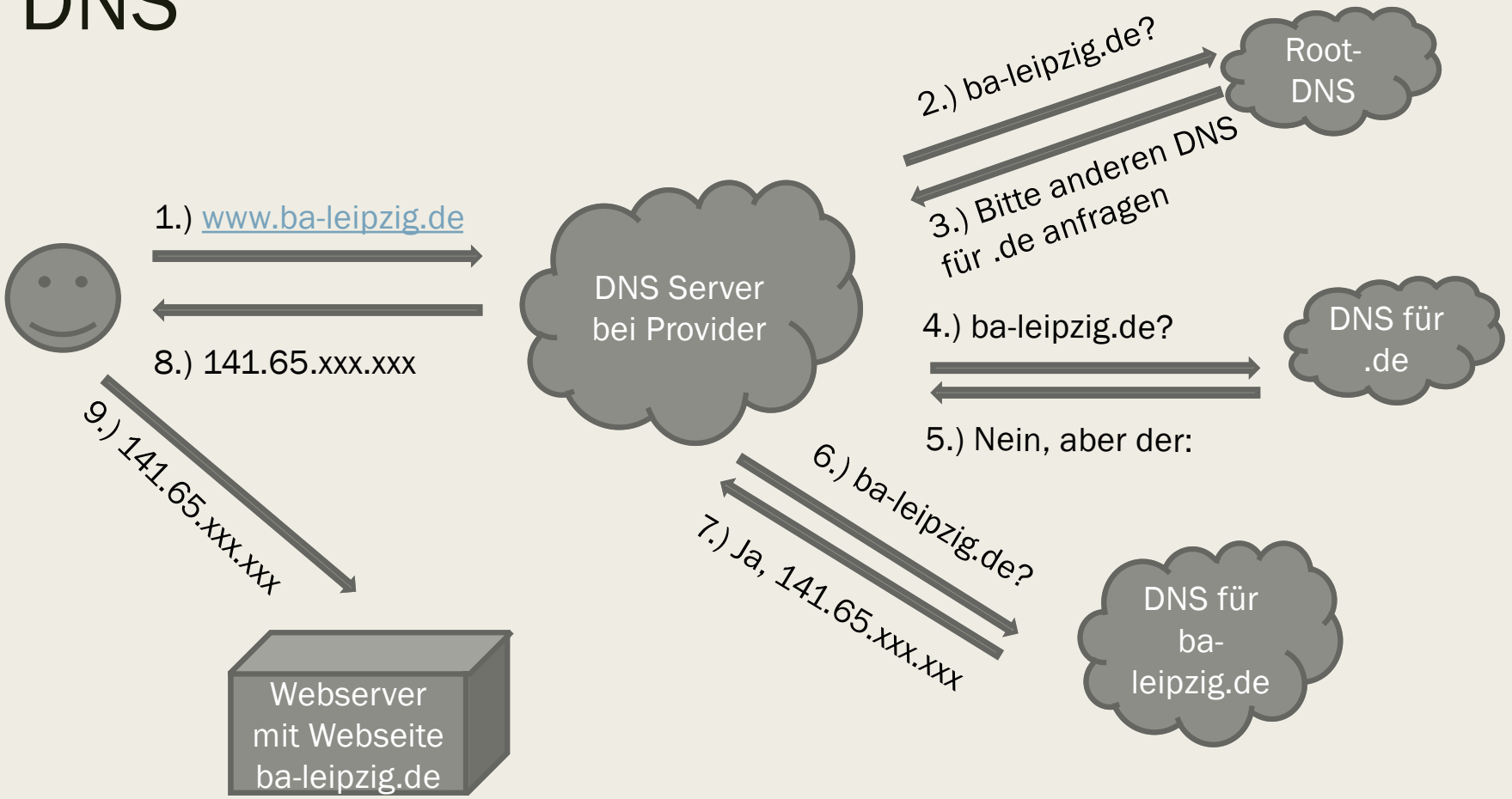
Eigener Webserver

- Lokal installierbar:
- XAMPP (https://www.apachefriends.org/de/download.html)
- WebSpace:
- https://www.heroku.com

Webserver und DNS

- Webseiten werden auf Webservern abgelegt und an die Welt “geliefert“
- Webhoster (All-Inkl, IONOS, Strato, Hosteurope, Hetzner) stellen Webspace und (meistens) eine Domäne (ba-leipzig.de)
- Abfrage von HTML-Dokumenten z.B.: www.ba-leipzig.de/ersteSeite.html
- DNS (Domain Name System) übersetzt Domänennamen in IP-Adressen…

DNS


1.) www.ba-leipzig.de
8.) 141.65.xxx.xxx
9.) 141.65.xxx.xxx
DNS Server bei Provider
2.) ba-leipzig.de?
3.) Bitte anderen DNS für .de anfragen
Root-DNS
4.) ba-leipzig.de?
5.) Nein, aber der:
DNS für .de
6.) ba-leipzig.de?
7.) Ja, 141.65.xxx.xxx
DNS für ba-leipzig.de
Webserver mit Webseite ba-leipzig.de

URLs

- URL - Uniform Resource Locator
- Gibt an, wo eine Ressource gefunden werden kann

Query String zur Übergabe von Parametern

http://www.ba-leipzig.de/seiten/unsereSeite.html?suche=students

HTTP – Hyptext Transfer Protocol
Zur Übertragung von Hypertext Markup Language
+ Bilder, Videos, Skripte

HTTPS – secure, übertragene Daten werden Verschlüsselt

Es muss auf dem Server nicht zwingend eine Datei mit diesem Namen geben!

HTTP-Request

GET /index.html HTTP/ 1.1
User-Agent: Mozilla/5.0 (Windows NT 5.2; WOW64; rv:15.0) Gecko/20100101 Firefox/15.0.1

1. HTTP **get**

Welche Ressource soll geholt werden

2. Der relative Anteil der zu ladenden URL
3. Protokollversion (1.0, 1.1, 2.0)
4. HTTP-Header enthalten Informationen über den Client

HTTP/1.1 200 OK
Content-Length: 12577
Content-Type: text/html; charset=UTF-8

1. Protokollversion
2. Statuscode (wichtig!)
3. Redundant zum Code
4. HTTP-Header

Statuscodes

Statuscode	Meldung	Bedeutung
200	OK	Inhalt wird geliefert
302	Found	Inhalt vorhanden, aber nicht an dieser Stelle. Nutzer wird weitgeleitet
304	Not Modified	Aktuellste Version des Inhalts liegt im Cache (Strg + F5 ;-))
400	Bad Request	Header/Parameter falsch
401	Unauthorized	Nur angemeldete Nutzer
403	Forbidden	Inhalt nicht freigegeben (auch für angemeldete Nutzer)
404	Not Found	Inhalt gibt es nicht
500	Internal Server Error	Server-Fehler (z.B.: PHP)

Geschichte des Webs

- ARPANET (Advanced Research Projects Agency), US Verteidigungsministerium
- Revolution: paketbasierte Computernetzwerke (Pakete mit Empfängeradresse)
- 1982 „Internet Protocol Suite“ – bis heute gültig!
- Hypertext erstmals 1963 von Ted Nelson „Xanadu“ verwendet
- Erster Projektvorschlag für das WWW (Forschungszentrum: CERN), Tim Berners-Lee

Erste Nachricht im ARPANET:
29.10.1969 um 10:30 Uhr „lo“ .

Tim Berners-Lee ist noch heute Direktor des World Wide Web Consortiums (W3C)

Styling mit CSS

- Neues HTML-Tag:
- Hat keine semantische Bedeutung
- Funktioniert in jedem Browser
- Markiert den enthaltenen Text (Text Text)
- Das style-Attribut enthält 1:n Style-Eigenschaften

Aufgabe:
Markieren Sie einen beliebigen Text mit und betrachten Sie das Ergebnis im Browser.

Span style

Aufgabe:
Fügen Sie diese Zeile in Ihr -Element ein.

Schöner Text

Style: enthält die Formatangaben für das HTML-Element

Style-Eigenschaften bestehen aus einem Namen, gefolgt von einem Doppelpunkt und einem Wert.

Jede Style-Eigenschaften wird mit einem Semikolon abgeschlossen

Inline-Styling lieber nicht verwenden!

Stylesheets

Aufgabe:
Übertragen Sie die Style-Angaben in eine .css-Datei und binden Sie diese ein.

- Können im <head>-Bereich oder in einer separaten Datei eingebunden werden
- Stylesheet-Dateien enden typischerweise auf .css
- Im <head>-Bereich mit <style>
- Um eine CSS-Datei einzubinden: <link>

```
<link href="styles.css" rel="stylesheet" />
```

Styles werden von außen nach innen wichtiger. Eigenschaften können somit Überschrieben werden.

Selektoren

Aufgabe (schwer):
Stylen Sie ein beliebiges Element über eine id.

- Tag-Selektor funktioniert in allen Browsern
- Die einfachsten Selektoren wählen **alle Tags einer Art** aus
- Selektor + 1:n Eigenschaften heißt Style- oder CSS-Regel
- Alle Eigenschaften, die zu einem Selektor gehören, werden in geschweiften Klammern gefasst
- Style-Eigenschaften werden wie in den Inline-Styles definiert
- Selektieren nach id funktioniert wie genauso wie Links zu einer #Sprungmarke

```
span {
    color: red;
    background-color: black;
}
```

Klassen-Selektor

- Genauer als Tag-Selektor, gröber als id-Selektor
- Klassen können beliebig oft in einem Dokument vorkommen
- Der Klassen-Selektor besteht aus einem Punkt
- Klassennamen (und ids) müssen mit einem Buchstaben anfangen (danach Ziffern, _ und – sind erlaubt)
- Mehrere Klassen sind erlaubt und werden durch ein Leerzeichen getrennt

```
.studium {
        color: blue;
}
```

```
<p class="studium">Rund ums Studium</p>
```

RGB

- Red, Green, Blue – Format
- Jede Komponente ist zweistellig als hexadezimale Zahl angegeben
- Werte von 0-9 sowie A-F
- 3B = 3*16 + 11 = 59; AF = 10*16+15 = 175 (FF = 255)
- Je höher der Wert, desto intensiver ist die Komponente in der Farbe vertreten

A = 10, B = 11, C = 12,
D = 13, E = 14, F = 15

Background

- background-image: url(bilder/bild1.jpg);
- background-position, background-repeat

Aufgabe
Definieren Sie ein Hintergrundbild für den gesamten Body Ihres HTML-Dokuments

Pseudoklassen

Aufgabe:
Ändern Sie die Link-Farben per CSS

- Links (<a>) verändern ihren Zustand (vor dem ersten Besuch in blau, danach lila)
- Pseudoklasse :visited, :hover
- Werden im Stylesheet mit einem Doppelpunkt angegeben

```
a, a:visited { color: blue }
```

Verschachtelungen

Legen Sie eine neue HTML-Seite an, mit einem Absatz und mehreren -Tags in diesem Absatz.
Legen Sie für den Absatz eine Schriftfarbe fest, zum Beispiel Grün.
Anschließend legen Sie für die einzelnen <span—Tags unterschiedliche Hintergrundfarben fest.

Diskutieren Sie das Ergebnis!

Lösung

```
<style>
        p {color: #00ff00;}
        #span1 {background-color: #000;}
        #span2 {background-color: #eee;}
</style>
```

```
<p>Irgendwas <span id="span1"> mit </span> meinen
<span id="span2"> Spans </span></p>
```

Font-size

- Absolute Größen: Pixel (px) oder Points (pt) – pt = 0,35mm
- xx-small, x-small, small, medium, large, x-large, xx-large (keine genauen Größen festgelegt!)
- Die Größe eines Pixels ist nicht konstant!
- Points passen nicht mit anderen Größenangaben in CSS zusammen

- Relative Größen: 120% bzw. 1.2em
- Macht die Schrift gegenüber dem umgebenden Element um 20% größer

Es gibt keine genaue Umrechnung zw. px und pt. Wie viele Pixel einem Point entsprechen, hängt von Größe und Auflösung des Bildschirms ab.

Font-Size 2

- Fügen Sie Ihrer HTML oder CSS-Datei eine neue Regel für Überschriften hinzu.
- -Tags in Überschriften sollen 10% größer dargestellt werden.

Nachfahren-Selektor

- Der Selektor sucht nur in der entsprechenden Umgebung
- Funktioniert in ALLEN Browsern
- Elemente, die direkt in einem anderen Element enthalten sind, nennt man Kinder (children)
- Elemente, die in einem anderen Element enthalten sind, egal, wie tief verschaltet, nennt man Nachfahren (descendants)

```
h1 .studium {
        color: red;
        font-style: italic;
}
```

Kind-Selektoren

- Ähnlich zu Nachfahren-Selektoren
- Funktioniert in ALLEN Browsern
- Beispiel: h1 > span
- Selektiert alle „span“, die Kinder eines h1 sind

Übung

- Definieren Sie in einem Stylesheet, dass alle gegenüber ihrer Umgebung (z.B. <p>) doppelte Schriftgröße haben sollen
- Definieren Sie danach eine Klasse, die diese Regel wieder aufhebt und der Text wieder eine „normale“ Größe hat

Erklären Sie

- div span (1)
- div span.wichtig (2)
- div p span (3)
- #seitenkopf > span (4)
- .wichtig > p > span (5)
- .wichtig > .wichtig > .wichtig (6)

Lösung

- (1) alle spans innerhalb eines div. Egal, wie tief verschaltet ist
- (2) alle span mit der Klasse wichtig innerhalb eines divs
- (3) spans in einem p in einem div
- (4) spans innerhalb des Elements mit der id seitenkopf
- (5) spans, die direkte Nachfahren eines p sind, das wiederum direkter Nachbar eines Elementes mit der Klasse wichtig ist
- (6) Elemente mit der Klasse wichtig, die von Elementen mit der Klasse wichtig sind, die auch wieder Kinder von Elementen mit der Klasse wichtig sind ;-)

Listen

```
…
<body>
        <p>
                Theoretische Informatik <br/>
                Softwareentwicklung <br />
                Mathematik <br />
                Datenbanken <br />
        </p>
</body>
…
```

Eine gute Idee?

Listenelemente

- (unordered List)
- (ordered List) – aber NICHT sortiert!
- markiert jedes Element mit dem gleichen Zeichen
- zählt die Elemente fortlaufend
- Mit <li> (list item) werden die Elemente einer Liste verpackt

Übung: Erstellen Sie eine nummerierte Liste mit drei Elementen.

Tabellen

- <table> für zusammengehörige Daten, die eine Bedeutung haben
- <tbody> umschließt den eigentlichen Tabellen-Inhalt
- Tabellen sind zeilenorientiert!
- <tbody> hat als Kind-Elemente eine oder mehrere Tabellenzeilen (<tr>)
- Jede Zeile enthält eine oder mehrere Tabellenzellen (<td>)

Tabellen – Achtung!

- Kein Element für Spalten!
- Es gibt nur Zeilen und Zellen!
- Der Browser sorgt in der Darstellung dafür, dass die jeweils erste Zeile jeder Zeile zu einer Spalte wird, die jeweils zweite Zelle zu einer Spalte!
- Alle Zeilen MÜSSEN die gleiche Anzahl an Zellen haben!
- Auch leere Zellen müssen unbedingt angegeben werden!

Tabellen - Übung

Web-Technologien	CPWT	28 LV	60 Min.	5 ECTS
Datenbanken	DB	34 LV	120 Min.	10 ECTS
Theortische Informatik	TI	40 LV	120 Min.	10 ECTS
Mathematik	MA	120 LV	240 Min.	25 ECTS

Bauen Sie diese Tabelle mit HTML nach! Achtung! Die Creditpoints sind rechtsbündig ;-)

Tabellen - Lösung

```
…
<table>
 <tbody>
  <tr>
   <td>Web Techn..</td>
   <td>CPWT</td>
   <td>28 LV</td>
   <td>60 Min.</td>
   <td style=“text-align: right;“>5 ECTS</td>
  </tr>
  …
 </tbody>
</table>
```

Formulare

- Bisher: vom Server zum Client, oder?
- Jetzt: Informationen zum Server senden, z.B. für
  - Login-Masken
  - Kontaktformulare
  - Wikipedia-Seiten
  - …
- Wir benötigen: <form>-Tag!

Dieser Teil erfordert einen eigenen WebServer (z.B. XAMPP).

<form>-Tag

- Action gibt an, wohin die Daten gesendet werden
- Method gibt an, wie die Daten versendet werden sollen
- Accept-charset gibt den Character-Encoding an

```
<form action=“auswertung.php“ method=“get“ accept-charset=“utf-8“>
```

<input>-Tags

- Input deckt fast alle Arten von Eingaben ab.
- Der einfachste Typ: type=“text“ für ein einfaches Eingabefeld
- <input type=“text“ name=“vorname“>
- Der Name ist auch der Name, unter dem der Server diese Daten abrufen kann
- Funktioniert in ALLEN Browsern

Formulare - Übung

- Legen Sie ein neues HTML-Dokument an, mit einem <form>-Tag und drei <input>-Tags als Listen-Elemente an
- Vorname, Nachname, E-Mail sollen als Text übertragen werden (action="register.php" method="get")
- Mit <button>absenden</button> fügen Sie noch einen Button vor das schließende <form>-Tag ein.

- Ergänzen Sie das Formular um ein Passwort-Eingabefeld!

Requests

- GET als URL-Parameter in der Browserleiste
- method=“post“ ändert den Request-Typ
- Auf dem Server ist die Umstellung u.U. schwieriger!
- GET-Requests um Seiten zu holen (URL-Parameter sind nützlich für Lesezeichen!)
- POST-Request um Daten an den Server zu senden, die dann ggf. gespeichert werden sollen

Weitere Typen

- Checkbox / Radio-Buttons:
- <input type=“radio“ name=“geschlecht“ value=“w“ id=“geschlecht_w“>
- <input type=“radio“ name=“geschlecht“ value=“m“ id=“geschlecht_m“>

- <input type=“checkbox“>

- Select-Boxen:
- <select name=“studium“>
  - <option value=“info“>Informatik</option>
  - <option value=“contr“>Controlling</option>
  - …
  - </select>

Formulare 2.0 – HTML5

Typ	IE 11	Edge	Firefox	Chrome	Safari	Bemerkung
type=“email“	Ja	Ja	Ja	Ja	Ja	Lässt nur gültige E-Mails zu
type=“date“	Nein	Ja	Ja	Ja	Nein	Eingabe eines Datums.
type=“number“	Ja	Ja	Ja	Ja	Ja	Lässt nur Zahlen zu
type=“search“	Ja	Ja	Ja	Ja	Ja	Wird vom Browser wie ein Suchfeld dargestellt
Required	Ja	Ja	Ja	Ja	Ja	Markiert ein Feld als Pflichtfeld

Block- und Inline-Elemente

- Block-Elemente sind HTML-Elemente, die nicht innerhalb des Textes vorkommen, aber Text umfassen können.
- Blockelemente sind immer rechteckig
- z.Bsp.: <p>, <table>

- Inline-Elemente sind HTML-Elemente, die mitten im Text vorkommen können.
- z.Bsp.: <a>,

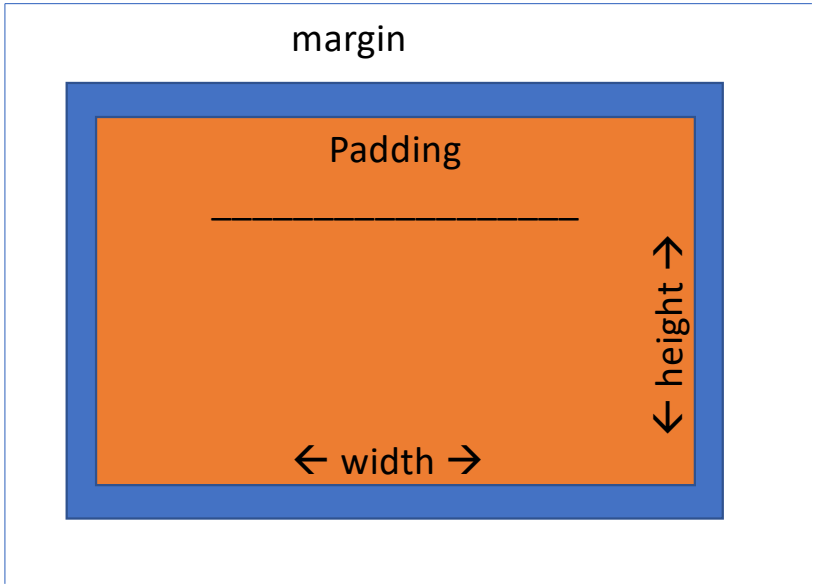
BOX-Model

- Das -Tag
- Ist ein generischer Container und kann alle Elemente enthalten
- funktioniert in allen Browsern.

Fügen Sie einen -Container in Ihren HTML-Code ein und beschreiben Sie das Layout-Verhalten.

BOX-Model 2

- width und height beziehen sich auf den Platz, der dem Inhalt einer Box zur Verfügung steht. Sie enthalten weder margin noch padding.
- Padding: Abstand zw. dem Inhalt und dem Rahmen
- Margin: Mindestabstand zw. dieser Box und der nächsten



Übung

- Platzieren Sie auf einer neuen Seite fünf s mit Text darin und den Ids eins, zwei, drei, vier, fuenf. Geben Sie verschiedene Hintergrundfarben an.

- Div 1 ohne Größenangaben
- Div 2 eine Höhe in Pixeln
- Div 3 Breite und Höhe in Pixeln (px)
- Div 4 Breite und Höhe in %
- Div 5 Breite und Höhe in em

Diskussion zur Übung

- Zwischen den s gibt es keinen Abstand
- Wenn der Text zu lang ist, geht er einfach über den Rand
- Bei div Nr. 4 wird die Box so hoch, bis der Text passt. Höhenangaben in % funktionieren nur zuverlässig, wenn für das umgebende Element eine Größe gesetzt ist.
- Ist keine Größe angegeben, wird das Element gerade so groß, dass sein Inhalt komplett hineinpasst.

Abstände

- padding: 10px; – in allen Himmelsrichtungen 10px Abstand
- padding: 10px 20px; Nord + Süd: 10px, West + Ost: 20px;
- padding: 10px 20px 5px: Nord: 10px; West+Ost: 20px; Süd: 5px;
- padding: 10px 20px 5px 15px: …?

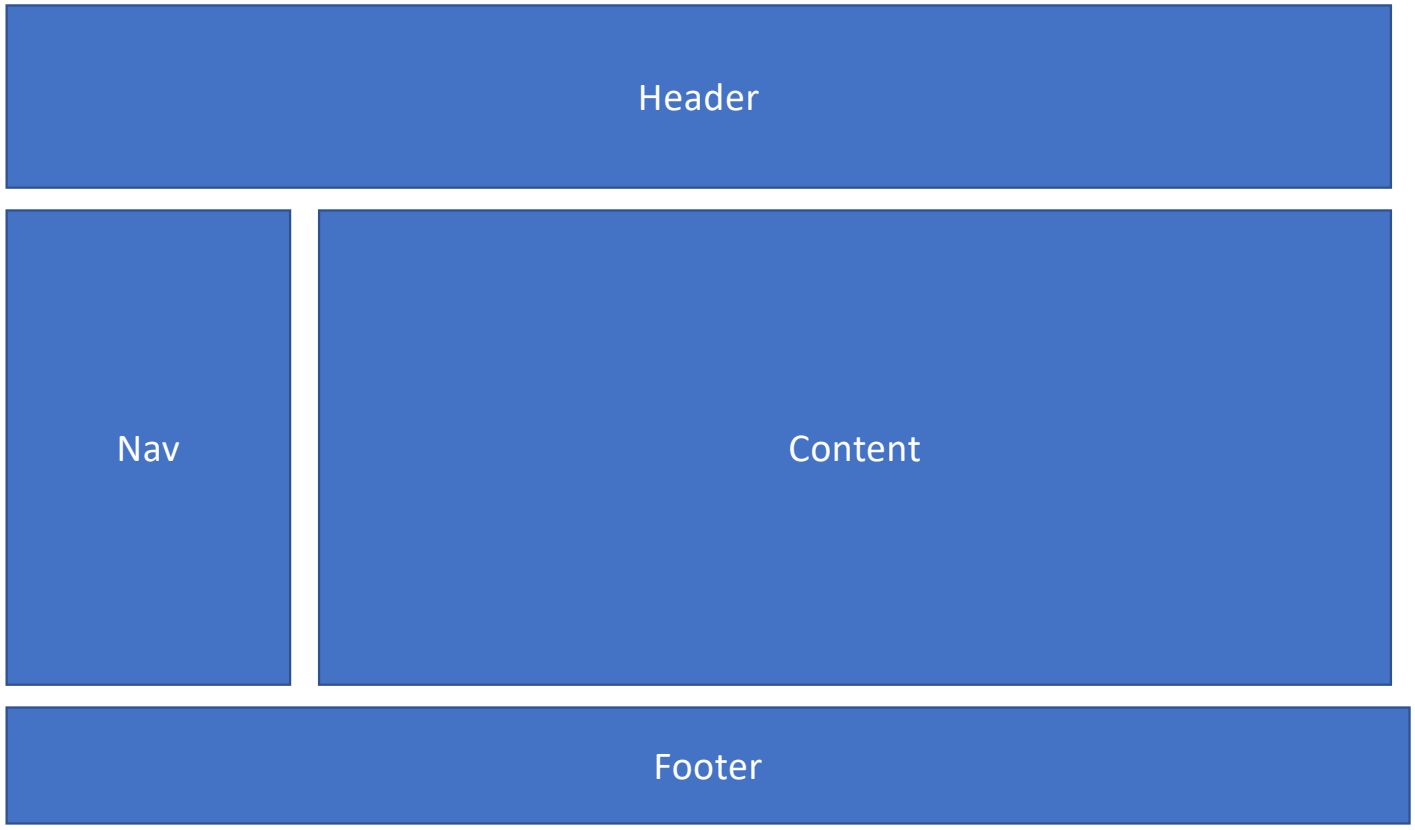
Das Fenster im Fenster

- <iframe> lädt eine HTML-Seite, deren URL im src-Attribut angegeben wird
- <iframe> sind komplett unabhängig

<iframe src="http://google.de">

Fügen Sie einen iframe ein und färben Sie allen Text grün!

Komplette Webseite



Web-Technologien

Teil 4

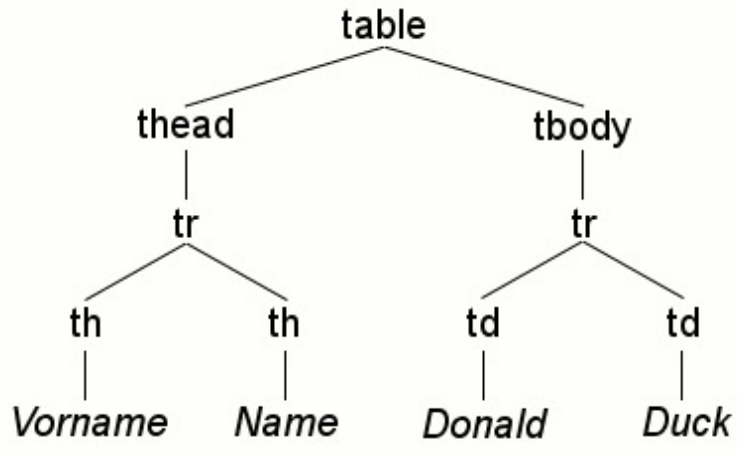
Ajax

- Asynchrones JavaScript and XML
- Asynchron: Webseite bleibt „nutzbar“, während im Hintergrund Daten geladen werden
- Request-Response-Modell wird abgelöst
- Seite muss nicht mehr komplett neugeladen werden
- Z.B.: Google Suchvorschläge
- XMLHTTP seit IE 5 (1999) verfügbar

Document Object Model

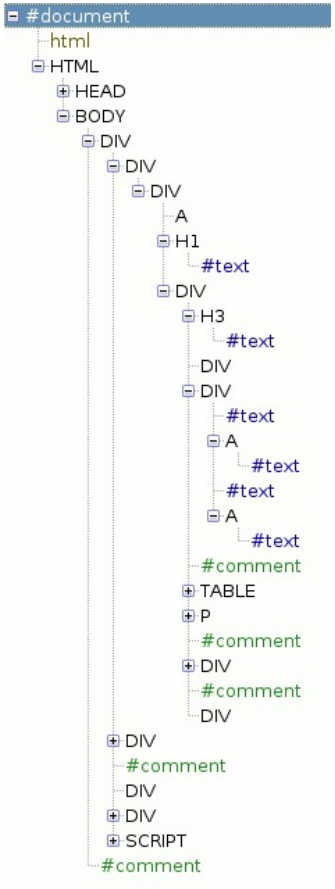
- API für Zugriff auf Web-Dokument
- Möglichkeiten: Knoten erstellen, ändern, löschen
- Anwendbarkeit: auf Client- aber auch Server-Seite

```
<table>
  <thead>
    <tr>
      <th>Vorname</th>
      <th>Name</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Donald</td>
      <td>Duck</td>
    </tr>
  </tbody>
</table>
```



Element Node Access

- getElementById()
- getElementsByName()
- getElementsByTagName()



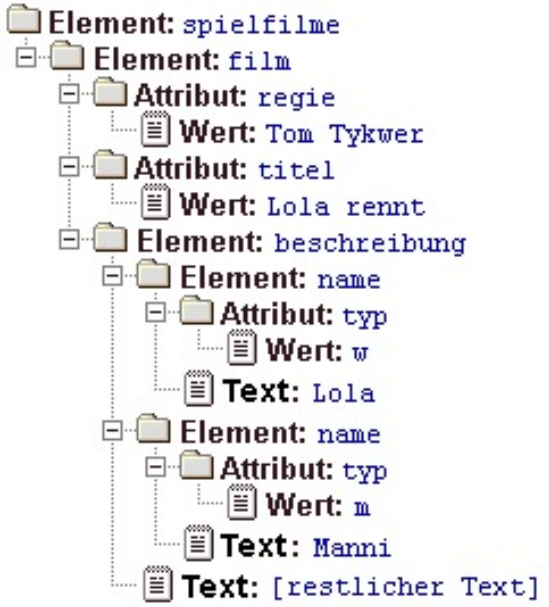
JavaScript Object Notation (JSON) versus XML

```
{
        "Herausgeber": "Xema",
        "Nummer": "1234-5678-9012-3456",
        "Deckung": 2e+6,
        "Waehrung": "EURO",
        "Inhaber":
        {
                "Name": "Mustermann",
                "Vorname": "Max",
                 Hobbys": [ "Reiten", "Golfen", "Lesen" ],
                …
        }
}
```

```
<Kreditkarte
        Herausgeber="Xema"
        Nummer="1234-5678-9012-3456"
        Deckung="2e+6"
        Waehrung="EURO">
        <Inhaber
        Name="Mustermann"
        Vorname="Max"
        maennlich="true"
        Alter="42"
        Partner="null">
                <Hobbys>
                        <Hobby>Reiten</Hobby>
                        <Hobby>Golfen</Hobby>
                        <Hobby>Lesen</Hobby>
                </Hobbys>
        </Inhaber>
</Kreditkarte>
```

Extensible Markup Language (XML)

```
<spielfilme>
        <film regie="Tom Tykwer" titel="Lola rennt">
                <beschreibung>
                        <name typ="w">Lola</name> rennt für <name typ="m">Manni</name>,
                        der 100000 Mark liegengelassen hat und noch 20 Minuten Zeit hat,
                        das Geld auszuliefern.
                </beschreibung>
        </film>
</spielfilme>
```



XML 2

- universelles Konzept für Datenspeicherung und –kommunikation
- nicht beschränkt auf Internet oder WWW
- verlustfreie Übertragbarkeit von Daten
- trennt Daten und Layout, d.h. dieselben Daten können zur Erstellung einer Website, eines Ausdrucks, eines akustischen Signals usw. verwendet werden

XML Derivate

- Extensible HTML (XHTML)
- Scalable Vector Graphics (SVG)
- Mathematical Markup Language (MathML)
- Wireless Markup Language (WML)
- …

XML Dateien

- Text-Dateien
- Tabulatoren und Leerzeichen können verwendet werden
- Kommentare:
  - <!-- The recommended storage temperature -->
    <storagetemperature>4°</storagetemperature>

SVG

```
<svg width="4in" height="3in">

<!-- The element g is used for grouping elements to complex units. --> <g>

<rect x="50" y="80" width="200" height="100" style="fill: #FFFFCC"/> </g>

</svg>
```

Case-Sensitivity

If DTD defines:
<!ELEMENT Chiptype (#PCDATA)>
then the XML file has to use:

<!-- Wrong -->
<chiptype>...</chiptype>
<CHIPTYPE>...</CHIPTYPE>

<!-- Correct -->
<Chiptype>...</Chiptype>

Anführungszeichen

```
<!-- Wrong -->
<area language=german dialect=saxonian>
```

```
<!-- Correct -->
<area language="german" dialect="saxonian">
<!-- Replace ' with &apos; -->
<bar drink="A German&apos;s beer">
```

Internal Document Type Definition

```
<?xml version="1.0"?>
<!DOCTYPE Greeting [ <!ELEMENT Greeting (#PCDATA)>
 ]>
<Greeting>Hello Jupiter!</Greeting>
```

Well-formed XML

- befolgt XML-Syntax-Regeln
- mindestens ein Datenelement (Wurzelelement)
- alle Elemente haben schließendes Tag
- keine sich überlappenden Tags
- beliebige Inhalte sind möglich

Well Formed	Not Well formed
<Personnel> <Employee> <Name>Seagull</Name> < ID> 3674 </ID> <Age>34</Age> </Employee> </Personnel>	<Personnel> <Employee> <Name>Seagull</Name> < ID> 3674 </ID> <Age>34 </Employee> </Personnel>

Valides XML

```
<?xml version="1.0"?>
<!DOCTYPE source [
        <!ELEMENT source (address, description)>
        <!ELEMENT address (#PCDATA)>
        <!ELEMENT description (#PCDATA)>
]>
<source>
        <address>http://www.willy-online.de/</address>
        <description>All interesting things concerning Men</description>
</source>
```

Web-Storage

- Zur Speicherung von Daten am Client
- Besser als Cookies ;-)
- Funktioniert in allen Browsern

Web Storage – So geht‘s

```
localStorage.setItem(“Zucker“, “2kg“);
localStorage[“Eier“] = 12;
alert(“Kaufe Eier“ + localStorage.getItem(“Eier“);
alert(“Kaufe Zucker“ + localStorage.getItem(“Zucker“);
```

Das localStorage-Objekt erlaubt den Zugriff auf Web Storage

Web Storage arbeitet mit Name-Wert-Paaren.

setItem nimmt einen Namen und einen Wert an und legt dieses Paar im Web Storage an

Zugriff auch mit eckigen Klammern möglich

getItem erwartet nur als Parameter nur den Namen des Eintrags

Web Storage auslesen

```
for (var i=0; i < localStorage.length; i++) {

        var name = localStorage.key(i);
        var wert = localStorage[name];
}
```

Web Storage auslesen

```
for (var i=0; i < localStorage.length; i++) {

        var name = localStorage.key(i);
        var wert = localStorage[name];
}
```

length wie viele Elemente liegen im Storage

Die Key-Methode gibt den Key des n-ten Key-Value-Paares zurück.

Mit dem key kann dann der dazugehörige Value geholt werden

Einkaufsliste

- Erstellen Sie eine einfache Einkaufsliste

Ajax 2

- Ablauf:
  - Request-Objekt erzeugen
  - Eventlistener registrieren
  - Request absenden

```
var ajaxRequest = new XMLHttpRequest();

ajaxRequest.addEventListener(“load“, ajaxGeladen);
ajaxRequest.addEventListener(“error“, ajaxFehler);
```

Ajax 3

```
ajaxRequest.open(“get“, “/ajaxURL“);
ajaxRequest.send();
```

Die open-Methode legt fest, was per Ajax aufgerufen werden soll

Die Request-Methode, analog zu den Formularen, ist entweder get oder post

URL, gegen die der Request ausgeführt werden soll

Send führt den Request aus