

Web Application Development

HTML

Martina Freundorfer

Letztes Mal

- Client-Server-Modell
- URIs
 - **scheme:authority**[path][?query][#fragment]
- HTTP
 - Protokoll zur Übertragung von Daten über ein Rechnernetz
 - Zustandslos
 - Client-Server Kommunikation
 - Webseiten aus dem WWW, aber AUCH Webservices
 - Wie sieht eine HTTP-Nachricht aus?
- Was hat das alles mit Webanwendungen zu tun?
- An Umfragen anonym teilnehmen

<https://www.surveymonkey.de/r/FRCCNDP>

<https://www.surveymonkey.de/r/FX7Y8KZ>

Semesterplan

#	K W	Datum	Vorlesung	Das Labor nach der Vorlesung findet für beide Gruppen wöchentlich statt.
1	14	3.4.	Einführung, Scheinkriterien	npm im Selbststudium, Anlegen eines Webprojektes, Installation von Live Server
2	16	17.4.	Client-Server, Web Apps, URI, HTTP	Aufgabe von Beleg 1: AdViz (nur HTML und CSS)
3	17	24.4.	HTML, CSS	Prototyp AdViz
4	20	15.5.	CSS, JavaScript allgemein	Abgabe: Prototyp AdViz
5	22	29.5.	JavaScript: DOM, JSON, AJAX	Aufgabe von Beleg 2: Adviz mit JS
6	23	5.6.	React Framework	AdViz mit JS
7	24	12.6.	React Framework	AdViz mit JS
8	25	19.6.	React Framework	Abgabe: AdViz mit JS
9	26	26.6.	React Framework / NodeJS	Aufgabe von Beleg 3: AdViz mit Backend
10	27	3.7.	NodeJS	AdViz mit Backend

Heute

- HTML = **Hypertext** *Markup Language*
- Geschichtliches
- Aufbau einer Webseite
- Lernziel:
 - Sie können eine einfache, statische HTML-Seite und HTML-Form erstellen.

Hypertext

- Hyper (altgriechisch): über, oberhalb, darüber hinaus
- Wikipedia (de):
 - Ein **Hypertext** ist ein Text mit einer netzförmigen, dynamischen Struktur, in der die gewohnte Ordnung (lineare Sequenzialität) von statischen gedruckten Publikationen technisch aufgebrochen wird.
 - Vom typischen Buch unterscheidet er sich dadurch, dass er nicht dafür geschrieben ist, von Anfang bis Ende in der veröffentlichten Reihenfolge gelesen zu werden.
 - Er wird in Auszeichnungssprachen geschrieben, die neben für den Leser nicht sichtbaren Gestaltungsanweisungen auch Hyperlinks enthalten, also Querverweise zu entfernten Textpassagen oder anderen Dokumenten im Netzwerk.

Hypertext

- Wikipedia (eng):
 - **Hypertext** is text displayed on a computer display or other electronic devices
 - with references (hyperlinks) to other text that the reader can immediately access, or where text can be revealed progressively at multiple levels of detail.
 - Hypertext documents are interconnected by hyperlinks, which are typically activated by a mouse click, keypress set or by touching the screen.
- 'Hyper-' is used in the mathematical sense of extension and generality (as in 'hypercube') rather than the medical sense of 'excessive' ('hyperactivity'). There is no implication about size—a hypertext could contain only 500 words or so. 'Hyper-' refers to structure and not size.

Hypertext

Definitionsversuch des World Wide Web Consortium (W3C):

- Hypertext ist ein Text, der nicht linear sein muss (*not constrained to be linear*).
- Hypertext ist ein Text, der Links zu anderen Texten enthält.
- HyperMedia ist ein Hypertext, der auch Grafiken, Videos oder Klänge enthalten kann (*not constrained to be text*).
- Hypertext und HyperMedia sind Konzepte, keine Produkte.

Markup Language

- Deutsch: Auszeichnungssprache
- Wikipedia (de): Eine **Auszeichnungssprache** ist eine maschinen-lesbare Sprache für die Gliederung und Formatierung von Texten und anderen Daten.
- Mit Auszeichnungssprachen werden Eigenschaften, Zugehörigkeiten und Darstellungsformen von Abschnitten eines Textes (Zeichen, Wörtern, Absätzen usw.) oder einer Datenmenge beschrieben. Dies geschieht in der Regel, indem sie mit *Tags* markiert werden.
- Elemente eines Textes, also Zeichen, Wörter, Absätze, ggf. Graphiken werden durch Tags markiert, also ausgezeichnet

SGML

- Englisch: SGML = Standard Generalized Markup Language
- Deutsch: Genormte Verallgemeinerte Auszeichnungssprache
- SGML ist seit 1986 ein ISO-Standard: *ISO 8879:1986 Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML)*
- eine Metasprache, mit deren Hilfe man verschiedene Auszeichnungs-sprachen für Dokumente definieren kann
- D.h., SGML stellt Werkzeuge (Capabilities) und Regeln für die Erstellung von “Mengen von Tags” (Eng: sets of tags) zur Verfügung
- “Früher”: HTML ist ein solcher “set of tags”
- **Frühere Versionen von HTML** waren Anwendungen von SGML

XML

- XML = e**X**tensible **M**arkup **L**anguage
- Auch eine Meta Markup Language
- Entwickelt von einem Komitee des World Wide Web Consortium mit Jon Bosak als Vorsitzender
- 10. Februar 1998
- Grund für die Entwicklung von XML :
 - Vereinfachung von SGML
 - Fokussiert auf Dokumente im Internet
 - Anwendungsspezifische XML-Sprachen können leichter definiert werden, z.B., Nutzer können ihre eigenen Tags kreieren, und diese beschreiben (extensibility = Erweiterbarkeit)
- Definition dieser anwendungsspezifischen XML-Sprachen (Tags) erfolgt in ein Document Type Description (DTD) oder in einem XML Schema
- XML ist eine Untermenge von SGML
- **Heute: XML hat SGML in der Praxis abgelöst.**

HTML Versionen

- **HTML** wurde erstmals 1989 von Tim Berners-Lee am CERN in Genf vorgeschlagen.
- **HTML** (ohne Versionsnummer, 3. November 1992): Urversion, die sich nur an Text orientierte.
- **HTML** (ohne Versionsnummer, 30. April 1993): Zu Text kam neben Attributen wie fette oder kursive Darstellung die Bildintegration dazu
- **HTML+** (November 1993) Geplante Erweiterungen, die in spätere Versionen einfließen, aber nie als HTML+ verabschiedet wurden
- **HTML 2.0** (November 1995): Die mit [RFC 1866](#) definierte Version führte u. a. Formulartechnik ein. Der Status dieses Standards ist „HISTORIC“. Auch die Vorgänger sind veraltet.
- **HTML 3.0**: Nicht erschienen, weil sie mit der Einführung des Netscape-Browsers in der Version 3 bereits vor der geplanten Veröffentlichung veraltet war.
- **HTML 3.2** (14. Januar 1997): Neu waren zahlreiche Features wie Tabellen, Textfluss um Bilder, Einbindung von Applets.
- **HTML 4.0** (18. Dezember 1997): Einführung von Stylesheets, Skripten und Frames. Am 24. April 1998 erschien eine leicht korrigierte Version.
- **HTML 4.01** (24. Dezember 1999): Ersetzte HTML 4.0 mit vielen kleineren Korrekturen

HTML Versionen

- **XHTML 1.0** (26. Januar 2000): Neuformulierung von HTML 4.01 mit Hilfe von XML. Am 1. August 2002 erschien eine überarbeitete Version.
- **XHTML 1.1** (31. Mai 2001): Nachdem XHTML in Module aufgeteilt wurde, wurde mit XHTML 1.1 eine strikte Version definiert, bei der die mit HTML 4 eingeführten Varianten Frameset und Transitional entfielen.
- **XHTML 2.0** (26. Juli 2006, closed 2009): Diese Version sollte nicht mehr auf HTML 4.01 basieren und einige neue Elemente einführen, so z. B. `<nl>` für Navigationslisten. Die Trennung von Auszeichnung und Stil sollte in dieser Version vollendet werden. – Das W3C hat die Arbeiten an XHTML 2.0 im Sommer 2009 eingestellt, weil XHTML durch HTML5 ersetzt werden sollte.
- **HTML5** (Recommendation, 28. Oktober 2014): Schuf auf Basis von HTML 4.01 und XHTML 1.0 ein neues Vokabular. Die zu HTML gehörende DOM-Spezifikation wurde ebenfalls überarbeitet und erweitert.
- **HTML 5.1** (Recommendation, 1. November 2016)
- **HTML 5.2** (Recommendation, 14. Dezember 2017): **Aktuelle Version**
- HTML 5.3 (Working Draft, 18. Oktober 2018)

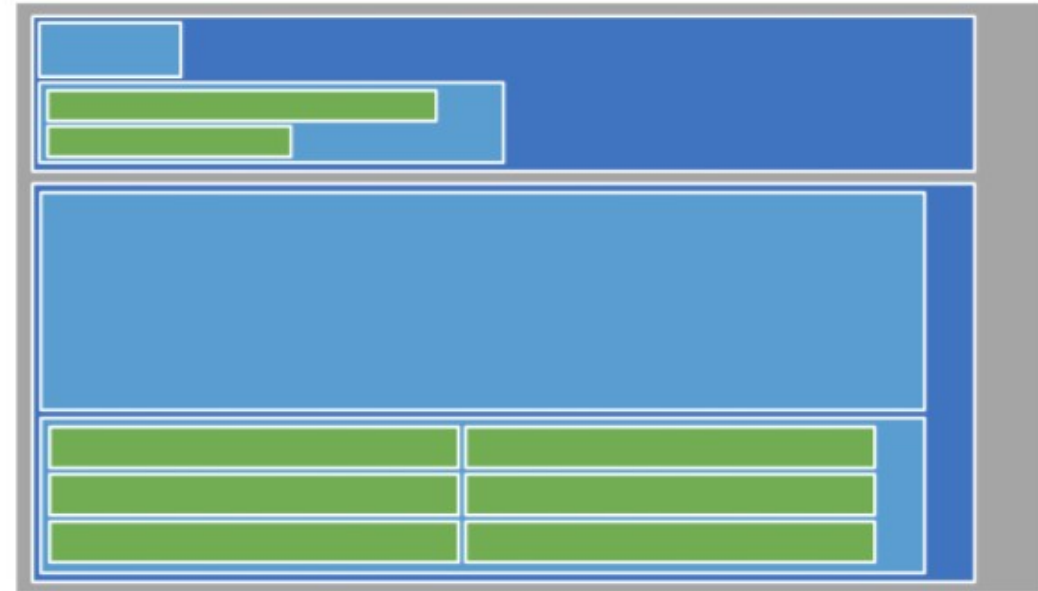
HTML 5

- 2000: Reformulierung von HTML 4.01 zu XML-basierten XHTML
- W3C konzentriert sich auf XHTML, neue Versionen sind nicht mehr abwärtskompatibel und XHTML wird zu kompliziert
- 2004: Mehrere Browserhersteller gründen die Web Hypertext Application Technology Working Group (**WHATWG**, pronounced “WHAT double you gee” oder WHAT wig) und erarbeiten einen ersten Vorschlag von HTML 5
- 2006: Tim Berners-Lee gründet neue **Arbeitsgruppe innerhalb des W3C für HTML-Weiterentwicklung**
- 2007: WHATWG und W3C arbeiten gemeinsam an HTML5
- 2009: W3C entscheidet Entwicklung von XHTML2.0 wird nicht mehr weitergeführt
- Oktober 2014: Spezifikation für HTML 5 vorgelegt vom W3C
- Ersetzt die Standards HTML 4.01, XHTML 1.0 und DOM HTML Level 2

HTML 5

- WHATWG verfolgt ein versionsloses Modell der Entwicklung:
 - Sie arbeitet an einem sogenannten *Living Standard*, also einer Spezifikation, die einer ständigen Korrektur und Erweiterung unterliegt
 - Standard wird fortgeschrieben, auf Basis von Rückmeldung von Webentwicklern und Browserherstellern
 - WHATWG verzichtet auf die Versionsangabe „5“
 - nur „HTML-Standard“: <https://html.spec.whatwg.org/multipage/>
- Ziele der HTML-Arbeitsgruppe des W3C:
 - Stabile Momentaufnahme dieser Spezifikation unter dem Namen HTML5 zu publizieren
 - Snapshots werden versioniert
 - Dafür muss vordefiniertes Prozedere durchlaufen werden: Working Draft → Candidate Recommendation → Proposed Recommendation → W3C Recommendation
 - 2014: W3C Recommendation (HTML 5)
 - 2016: W3C Recommendation (HTML 5.1)
 - Aktuell, seit Dezember 2017: W3C Recommendation (HTML 5.2)

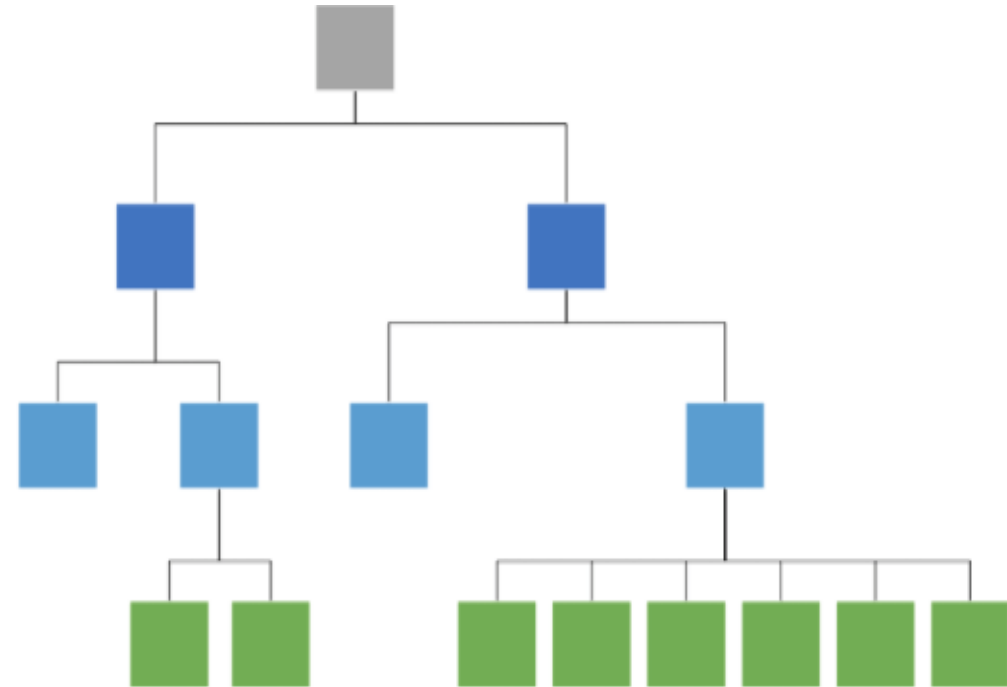
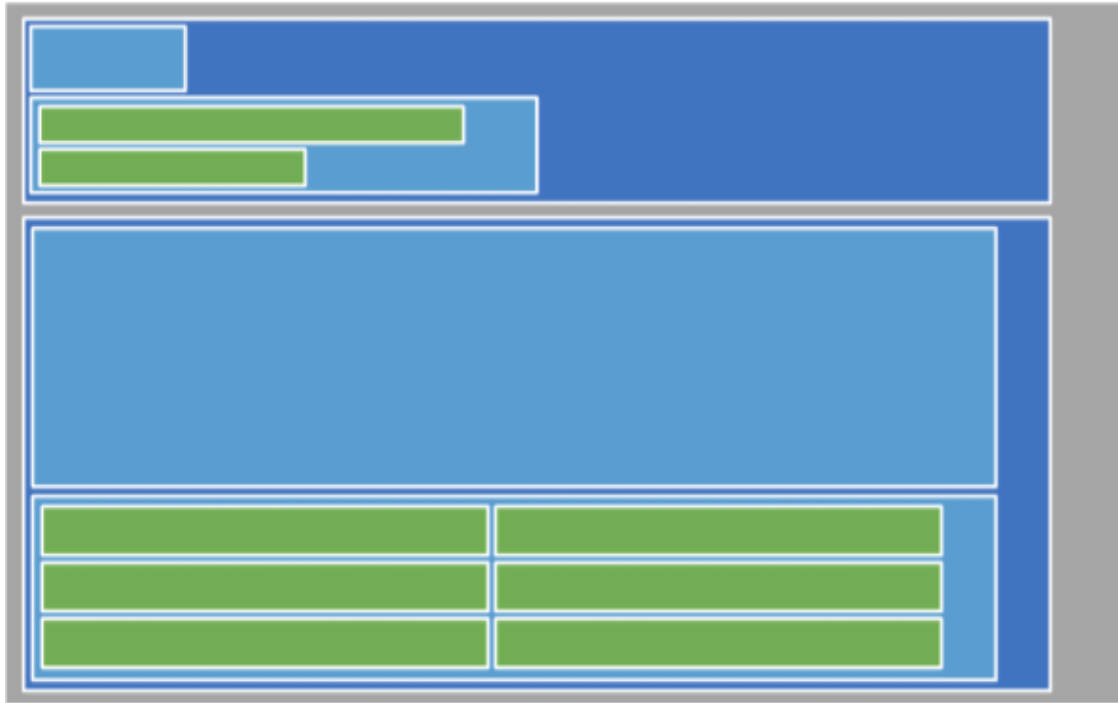
Webseiten



- Webseiten bestehen aus geschachtelten Boxen
- Geschachtelte Boxen entsprechen Elementen des HTML-Dokuments

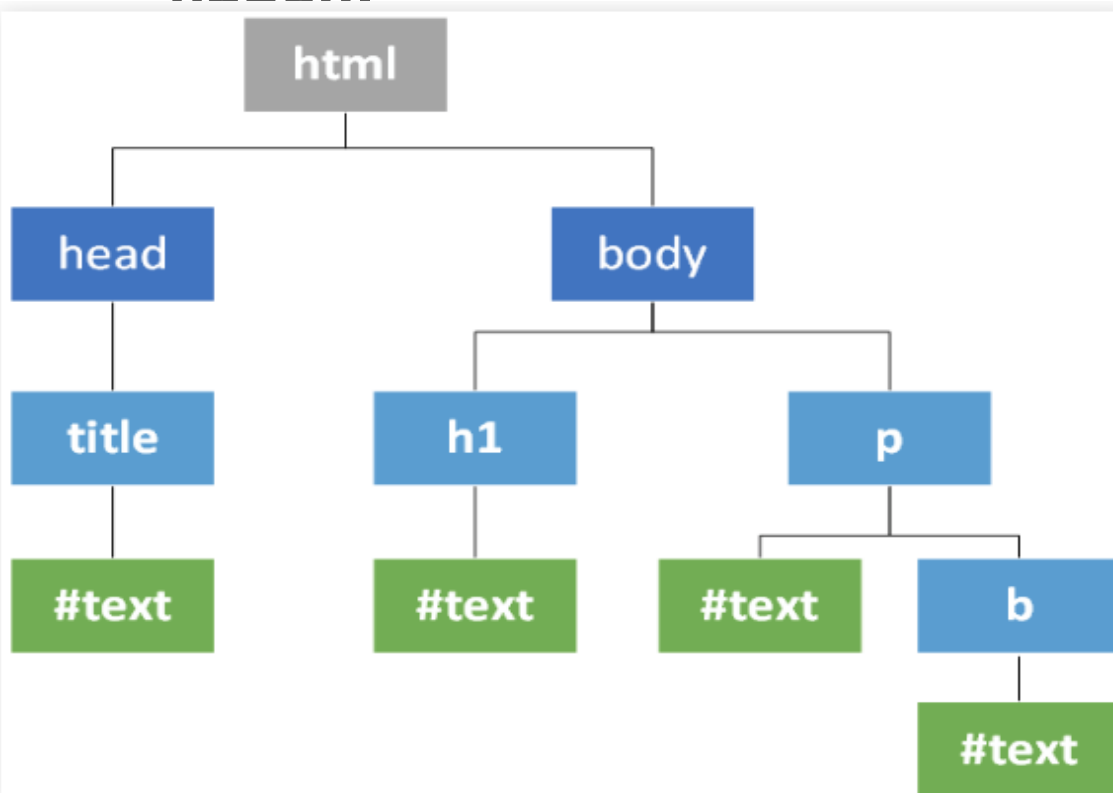
HTML: Baumstruktur

- Somit lassen sich die Boxen als Baum verstehen (Baum steht Kopf!):



HTML - Baumstruktur

- **HTML-Dokument:** Meist in einer .html- oder .htm-Datei gespeichert
 - konzeptionell (und im Arbeitsspeicher des Browsers): ein Baum, welcher durch **DOM**, dem Document Object Model, repräsentiert wird
 - besteht aus **Elementen**
 - **Das Wurzelement ist das “html”-Element.**
 - **Wurzelement hat Subelemente oder Children-Elements.**
 - **Jedes Element kann keine, eins oder mehrere Subelemente haben.**



```
<html>
<head>
  <title>Webentwicklung</title>
</head>
<body>
  <h1>Webentwicklung</h1>
  <p>
    Willkommen auf der Webseite
    zur Veranstaltung
    <b>Webentwicklung!</b>
  </p>
</body>
</html>
```

HTML: Root, Elements, Attributes

- Das Wurzelement: `<html> ...</html>`
- Beispiel für ein Element: `<h1>Webentwicklung</h1>`
- Ein HTML-Element mit Attributen:

`<input type="text" id="firstnameld" name="firstname"/>`

Attribute sollten einen
kleingeschriebenen Namen und
einen Wert haben.
Dieser Wert sollte von " "
umschlossen sein.

- Viele Browser sind sehr tolerant. Ein HTML-Dokument, wenn man es als XML betrachtet, muss nicht wohlgeformt sein (syntaktisch inkorrekt) wird vom Browser aber trotzdem in einen korrekten Baum transformiert und wird dargestellt.

Beispiel: https://developer.mozilla.org/de/docs/Learn/Getting_started_with_the_web/HTML_basics

Globale Attribute (Auswahl)

- Sind Attribute, die man für alle HTML-Elemente einsetzen kann, z.B.:
 - ▢ **id**: dokumentweit einzigartiger Name zur Identifizierung des Elements
 - ▢ **class**: Klassifizierung /semantische Gruppierung mehrerer Elemente (siehe CSS)
 - ▢ **title**: ergänzender Text zum Element (häufig für Tooltips benutzt)
 - ▢ **style**: zur Anpassung der Optik (siehe CSS)

Alle globalen Attribute sind hier aufgelistet:

https://developer.mozilla.org/de/docs/Web/HTML/Globale_Attribute

HTML-Dokument: Grundgerüst

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Webentwicklung</title>
  </head>
  <body>
    <h1>Webentwicklung</h1>
    <p>
      Willkommen auf der Webseite zur Veranstaltung
      <b>Webentwicklung!</b>
    </p>
  </body>
</html>
```

- <!DOCTYPE html> (Für HTML5: muss vorhanden sein)
- <html> - Wurzelelement (muss vorhanden sein)
- HTML-Header (kann vorhanden sein, kein “muss”)
- HTML-Body (kann vorhanden sein, kein “muss”)

<!DOCTYPE html>

- <!DOCTYPE html> ist kein HTML-Tag, sondern eine Anweisung für den Browser und gibt den “Typ” des Dokumentes an.
- Muss vor dem Rootelement <html> stehen.
- Früher häufig missachtet von Browsern
- Beispiel für XHTML: <https://www.w3.org/History.html> (open in Firefox or Chrome, “View Page Source”)
- **Für alle HTML5-Dokumente:**
<!DOCTYPE html> muss am Anfang stehen

HTML-Header

```
<head>
  <!-- Zusatzinformationen zum Dokument -->
  <meta charset="utf-8"> <!-- Kodierung -->
  <meta name="description" content="Test">

  <!-- Dokumententitel -->
  <title>Einführung in HTML</title>

  <!-- Verweis auf externe Ressourcen -->
  <link rel="stylesheet" href="style.css">

  <!-- Nicht zwingend nötig -->
  <link rel="shortcut icon" href="favicon.ico">
</head>
```

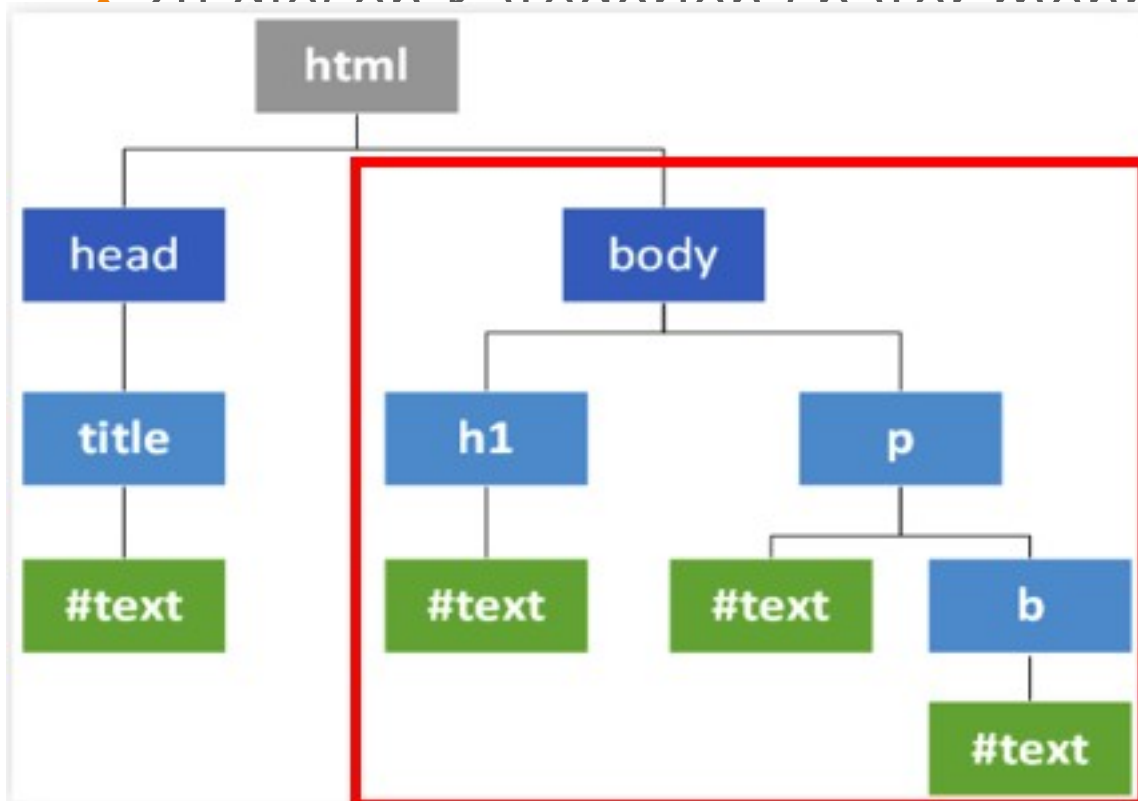
- Weitere meta-Elemente:
<https://developer.mozilla.org/de/docs/Web/HTML/Element/meta>
- Encoding/Kodierung ist wichtig! Siehe nächste Folie.

Encoding

- HTML-Dateien sind Textdateien, keine binären Dateien
- Zeichensatz oder Charset oder Encoding:
 - ▢ ASCII: 8-Bit-Zeichensatz, der aber nur 7 Bits verwendet und somit nur 128 Zeichen enthält (33 Steuerzeichen und 95 druckbare Zeichen)
 - ▢ ISO-8859-1: 8-Bit-Zeichensatz, 256 Zeichen inkl. äöüß, aber ohne €
 - ▢ UTF-8: 1 – 4 Bytes für die Kodierung, am weitesten verbreitete Kodierung für Unicode-Zeichen. Wird in einigen Betriebssystemen (GNU/Linux, Unix) und teilweise in verschiedenen Internetdiensten (E-Mail, Web) verwendet.
 - ▢ Und einige mehr
- <https://www.torsten-horn.de/techdocs/encoding.htm>
- **Hausaufgabe:**
<https://www.joelonsoftware.com/2003/10/08/the-absolute-minimum-every-software-developer-absolutely-positively-must-know-about-unicode-and-character-sets-no-excuses/>

HTML-Body

- `<body>`-Element enthält den Dokumentinhalt
- Die Subelements von *body* sind ineinander geschachtelte HTML-Elemente verschiedener Kategorien
- Ein HTML-Element kann keiner, einer oder mehreren Kategorien angehören
- Zu diesen Kategorien später mehr



```
<html>
<head>
  <title>Webentwicklung</title>
</head>
<body>
  <h1>Webentwicklung</h1>
  <p>
    Willkommen auf der Webseite
    zur Veranstaltung
    <b>Webentwicklung!</b>
  </p>
</body>
</html>
```


IDE: Ein Webprojekt zum Mitmachen

- Im IDE: Ein NPM-Webprojekt anlegen
- VS Code: index.html anlegen, Text kopieren



```
<!DOCTYPE html>
<html lang="de">
  <head>
    <!-- Zusatzinformationen zum Dokument -->
    <meta charset="utf-8"> <!-- Kodierung -->
    <meta name="description" content="Test">
    <!-- Dokumententitel -->
    <title>Einführung in HTML</title>

    <!-- Verweis auf externe Ressourcen -->
    <link rel="stylesheet" href="style.css">    </head>
  <body>
    <h1>Webentwicklung</h1>
    <p>
      Willkommen auf der Webseite zur Veranstaltung
      <b>Webentwicklung!</b>
    </p>
  </body>
</html>
```

HTML Tutorial

- https://www.w3schools.com/html/html_intro.asp

Textgestaltung und -strukturierung

- Überschriften: `<h1>... </h1>`, `<h2>... </h2>`, `<h3>... </h3>`, bis `<h6>`
- Absätze: `<p> Das ist ein ABSATZ </p>`
- Was passiert wenn ich viele Leerzeichen in einen HTML-Absatz einbaue?
- Apropos "Leerzeichen":
 - ▢ ` ` für einzelnes Leerzeichen (*non-breaking space*)
 - ▢ `
` für Zeilenumbruch (*break*)
- Leerzeichen können bewahrt werden:
`<pre> Jedes Leerzeichen ist
wichtig.</pre>`
- https://www.w3schools.com/html/tryit.asp?filename=tryhtml_elements

Textgestaltung und -strukturierung

Das muss dringend erledigt werden.
Du wolltest doch <mark>mindestens</mark>
70m²! Das Wort <i>das</i> ist ein Artikel.
<small>Copyright 2014 2015 Foo Inc.</small>

Das muss **dringend** erledigt werden.

Du wolltest doch **mindestens** 70m²! (high-lighted)

Das Wort *das* ist ein **Artikel**.

<https://lyrik.antikoerperchen.de/blog/sonstiges/bold-oder-wann-wird-was-benutzt-unterschiede-zwischen-semantisch-logischen-und-physisch-visuellen-elementen/>

Bilder und Links

```

```

```
<!-- explizite Größenangaben verhindern Sprünge beim Nachladen von  
Ressourcen --> 
```

```
<a href="http://de.wikipedia.org/">Wikipedia</a>
```

```
<a href="http://wikipedia.org/" target="_blank">Wikipedia</a>
```

```
<small>öffnet http://wikipedia.org in einem neuen Tab http://wikipedia.org  
</small>
```

Linkziele: Anker

```
<!-- Anker ("a") explizit setzen -->  
<h1><a name="overview"></a> Übersicht</h1>  
  
<!-- Elemente mit id-Attribut funktionieren auch -->  
<h2 id="faq">FAQ</h2>  
  
<!-- ... -->  
<a href="#overview">Zur Übersicht</a>  
  
<a href="#faq">zu den FAQ</a>
```

Listen

- Unnummerierte Listen `ul` (*unordered list*)
- Nummerierte Listen `ol` (*ordered list*)
- Definitions- oder Beschreibungsliste `dl` (*definition list*)

```
<ul>
  <!-- li = "list item" -->
  <li>Tick</li>
  <li>Trick</li>
  <li>Track</li>
</ul>
<ol> <li>Schlafen</li> <li>Essen</li> <li>Lernen</li> </ol>

<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

- Listen kann man auch schachteln
- Ausprobieren: https://www.w3schools.com/html/html_lists.asp

Tabellen

- Eine Tabelle table besteht aus Zeilen tr (*table row*)
- Zeilen bestehen aus Zellen td (*table data*)

```
<table>
  <tr>
    <td>A1</td> <td>A2</td> <td>A3</td>
  </tr>
  <tr>
    <td>B1</td> <td>B2</td> <td>B3</td>
  </tr>
</table>
```

A1	A2	A3
B1	B2	B3

- https://www.w3schools.com/html/html_tables.asp

Formulare

```
<form action="/survey" method="GET">
  <input type="text" id="stadt" name="stadt" value="Stadt"/> <br/>
  <input type="checkbox" id="nl" name="newsletter" checked/>
  <label for="nl">Newsletter?</label> <br/>
  <input type="number" id="alter" name="alter" value="23"/> <br/>
  <button type="submit">Absenden</button>
</form>
```

- "action" = an welches Script bzw. welchen Endpoint auf dem Server sollen die Daten des Inputs geschickt werden
- "method" = Welche HTTP-Methode soll genutzt werden?
- `<input type="text" id="stadt" name="stadt" value="Stadt"/>`

id:
eineindeutige
ID für dieses
HTML-Element.

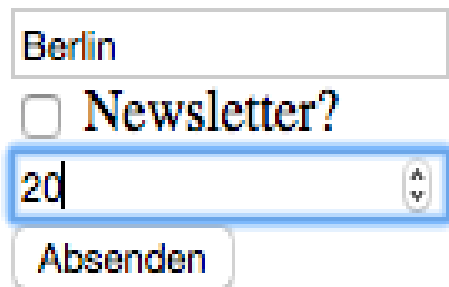
name: Name des
Elementes, wird
als Name des
HTTP-Parameters
verwendet

value: Wert des
Elementes, wird als Wert
des HTTP-Parameters
"name" an den Server
geschickt

Formulare

```
<form action="/survey" method="POST"> <!-- oder method="GET" -->
  <input type="text" id="stadt" name="stadt" value="Stadt"/> <br/>
  <input type="checkbox" id="nl" name="newsletter" checked/>
  <label for="nl">Newsletter?</label> <br/>
  <input type="number" id="alter" name="alter" value="23"/> <br/>
  <button type="submit">Absenden</button>
</form>
```

Unser Formular:



- Server erhält die Daten des Formulars in Form von HTTP-Parametern:

method="GET"	method="POST"
URL: http://my.host/somepath/survey? stadt=Berlin&newsletter=off&alter= 20	URL: http://my.host/somepath/survey mit HTTP-Body: stadt=Berlin newsletter=off alter=20

Input-Typen

```
Button: <input type="button"> HTML
Checkbox: <input type="checkbox">
Color: <input type="color">
E-Mail: <input type="email">
File: <input type="file">
Number: <input type="number">
Password: <input type="password">
Range: <input type="range">
Text: <input type="text">
URL: <input type="url">
```

Button:

Checkbox: ☐

Color:

E-Mail:

File: Choose File No file chosen

Number:

Password:

Range:

Text:

URL:

https://www.w3schools.com/html/html_form_input_types.asp

HTML 5: Neue Input-Typen

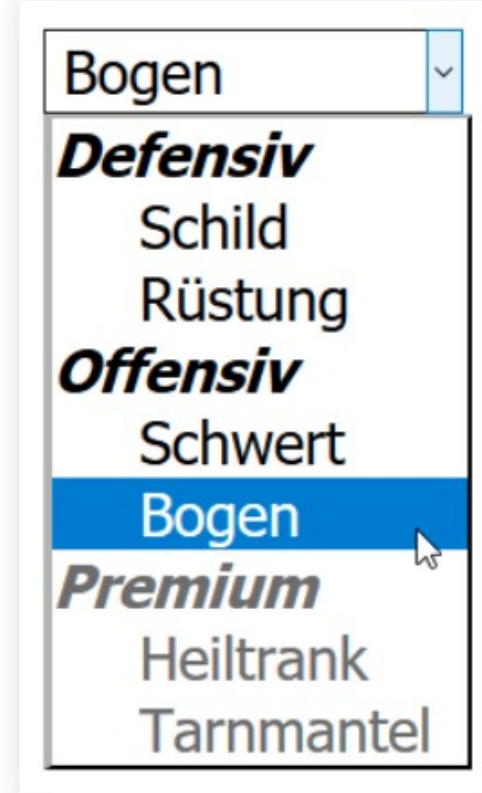
```
color: <input type="color"> <br/>
date: <input type="date"> <br/>
datetime-local: <input type="datetime-local"> <br/>
email: <input type="email"><br/>
month: <input type="month"> <br/>
number: <input type="number"><br/>
range: <input type="range"><br/>
search: <input type="search"><br/>
tel: <input type="tel"><br/>
time: <input type="time"><br/>
url: <input type="url"><br/>
week: <input type="week"><br/>
```

New input types that are not supported by older web browsers, will behave as `<input type="text">`.

- https://www.w3schools.com/html/html_form_input_types.asp

Auswahllisten

```
<select name="produkte">
  <optgroup label="Defensiv">
    <option>Schild</option>
    <option>Rüstung</option>
  </optgroup>
  <optgroup label="Offensiv">
    <option>Schwert</option>
    <option selected>Bogen</option>
  </optgroup>
  <optgroup label="Premium"
    disabled>
    <option>Heiltrank</option>
    <option>Tarnmantel</option>
  </optgroup>
</select>
```



Auswahllisten

```
<select name="produkte">
  <optgroup label="Defensiv">
    <option>Schild</option>
    <option>Rüstung</option>
  </optgroup>
  <optgroup label="Offensiv">
    <option>Schwert</option>
    <option selected>Bogen</option>
  </optgroup>
  <optgroup label="Premium" disabled>
    <option>Heiltrank</option>
    <option>Tarnmantel</option>
  </optgroup>
</select>
```

Kategorien der HTML-Elemente

- Ein HTML-Element kann keiner, einer oder mehreren Kategorien angehören
- Kategorien werden auch als “Content models” bezeichnet
- Folgende Kategorien werden in dieser Spezifikation (der von whatwg) unterschieden:
 - ▢ Metadata content: Definiert Präsentation oder Verhalten des restlichen Inhalts
 - ▢ Sectioning content: Semantische Unterteilung (z.B. Gültigkeitsbereiche von Headern und Footern)
 - ▢ Heading content: Definiert den Kopf eines Bereichs
 - ▢ Phrasing content: Zeichnet den Text des Inhalts aus
 - ▢ Embedded content: Bindet andere Ressourcen ein
 - ▢ Interactive content: Inhalt, der zur Benutzer-Interaktion gedacht ist
 - ▢ Flow content: Superset of all content modules, almost all elements
- Gucken Sie sich bitte den Link an. Er enthält eine schöne, interaktive Graphik der Kategorien:
<https://html.spec.whatwg.org/multipage/dom.html#kinds-of-content>

The End

- Questions?
- References:
 - ▢ HTML Living Standard: <https://html.spec.whatwg.org/multipage/>
 - ▢ HTML 5.2 W3C Recommendation: <https://www.w3.org/TR/html52/>
 - ▢ HTML5 Tutorial: <https://www.w3schools.com/html/default.asp>
 - ▢ Dane Cameron: “A Software Engineer Learns HTML5, JavaScript & jQuery”, Cisdal Publishings, 2015
 - ▢ Frank Zieris, Webentwicklung, WiSe2017/18, HTW:
https://www.zieris.net/teaching/htw-berlin/webdev-slides/?p=11_html