HTML

Einführung

HTML

- **H**yper **T**ext **M**arkup **L**anguage

- Beschreibt die Struktur von Webseiten (HTML-Dokumenten) mithilfe von *Markups (HTML-Elementen)*
- HTML-Elemente werden durch *Tags* repräsentiert, Tags stehen in spitzen Klammern **<tag>**
- Tags zeichnen Teile des Textes aus (Markupsprache, Auszeichnungssprache), z.B. "heading“ <h1></h1>, "paragraph“ , "table“ <table></table> usw.
- Browser zeigen nicht die Tags an, sondern nutzen diese zur Darstellung (Rendering) des Inhaltes

HTML – erstes Beispiel

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Erste Seite</title>
</head>
<body>
  <h1>eine große Überschrift</h1>
  <p>ein Absatz</p>
  <h3>eine kleinere Überschrift</h3>
  <p>noch ein Absatz</p>
</body>
</html>
```

HTML – erstes Beispiel

```
<html>
    <head>
        <title> Seitentitel </title>
    </head>
    <body>
        <h1> eine große Überschrift </h1>
        <p> ein Absatz </p>
        <h3> eine kleinere Überschrift </h3>
        <p> noch ein Absatz </p>
    </body>
</html>
```

Auszeichnungssprache
allgemeine Merkmale

- Strukturinformation wird in den „eigentlichen“ Inhalt integriert
  - Meta-Sprache zur Auszeichnung von Strukturinformation

- Strukturinformationen sind z.B.
  - Gliederungsebenen (Überschriften, Absätze etc.)
  - Layout (Zeichen, Schriften, Farben, Farbverlauf, Größe und Relation)

- Begriffe
  - Auszeichnung = **Markup**
  - Auszeichnungssprache = Markup-Sprache
  - Markup-Symbol (**Tag**) = Wort aus der Markup-Sprache
    - Tag = Marke = Etikett
    - insbesondere: Unterscheidung von Start-Tags und End-Tags

```
<h1>HTML – die Sprache des Web</h1>
```

Auszeichnungssprache
allgemeine Anforderungen

- Syntax und Semantik von Markup-Symbolen definierbar
- erweiterbar hinsichtlich neuer Strukturelemente und Dokumententypen
- von Menschen schreib- und lesbar
- in Programmiersprachen einbettbar
- offen für zukünftige Entwicklungen (neue Medientypen, Medienintegration)

HTML
Tags

- Tag in HTML: Zeichenkette innerhalb von **<>**

- Inhalt wird von einem Start- und Ende-Tag eingerahmt: **<tag>Inhalt</tag>**

- Start-Tag: **<xx>**
- Ende-Tag: **</xx>** mit xx als Tag-Name
- Beispiel: ** Text im Absatz **

- Wirkung des Tags bezieht sich auf den umschlossenen Bereich
- Tags ohne Inhalt haben auch kein Ende-Tag, z.B. **
** (Standalone-Tags)
  - es geht auch
 (HTML5)
  - aber <br /> ist auch XHTML-gerecht (ist auch HTML5)
  - es geht sogar
</br> (aber völlig unüblich))

HTML
verschachtelte Elemente

- Tags können geschachtelt werden

- es entsteht eine hierarchische Struktur

- Wirkung wird von außen nach innen vererbt
- Beispiel:

```
<small>kleiner
        <strong>fetter Inhalt
        </strong>
</small>
```

- Elemente werden in umgekehrter Reihenfolge geschlossen, in der sie geöffnet werden
- Fehler:

```
<small>kleiner
        <strong>fetter Inhalt
</small>
        </strong>
```

HTML
Geschichte

- Projekt für Austausch innerhalb des CERNs von strukturierten Daten → 1989
- HTML 1, 1992
  - Vorstellung von frühen Versionen von HTML, HTTP, und URL
- **…**
- **HTML 4.01**, 1999
  - Style Sheets (CSS), Einbindung von Scripts, ... (vieles nie implementiert)
- XHTML 1.0, 2000: Neudefinition von HTML 4.0.1 auf XML-Basis
- **HTML 5**, 2014:
  - Weiterentwicklung von HTML4 und XHTML
  - „lebender“ Standard;
  - aktuell 5.2 (Recommendation 14. Dezember 2017)

- wird vom World Wide Web Consortium (**W3C**) und der Web Hypertext Application Technology Working Group (**WHATWG**) weiterentwickelt

HTML
Versionen

- **HTML 4**
  - Stabile Version, von allen Browsern gut unterstützt
  - robust gegenüber kleinen Fehlern
  - Sicherheit, dass alle Browser die Inhalte richtig darstellen

- **XHTML**
  - „strenge“ Variante von HTML gemäß XML

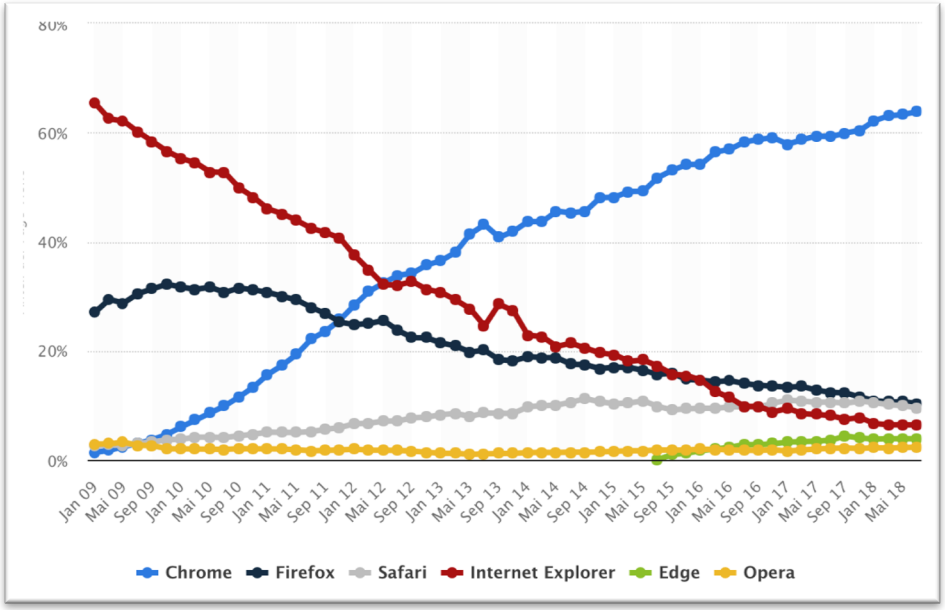
- **HTML 5**
  - aktuelle HTML-Version
  - Browser unterstützen die Neuerungen unterschiedlich
  - Permanent Neuerungen

  - *Wir: HTML 5 → XHTML-konform (kleine Tags, Beenden der Tags)*

HTML

Inkompatibilitäten und Weiterentwicklungen

- Permanent neue Entwicklungen (neue HTML-ELemente) durch die Browser-Hersteller
- Test, ob W3C-Standard-konform: http://validator.w3.org/
- Eigenentwickelte Webseiten müssen auf mehreren Browsern getestet werden



HTML

Gestaltung von Webseiten

- Webseiten können grundsätzlich über folgende Elemente erstellt und gestaltet werden:
  - HTML-Tag
  - HTML-Tag mit Attributen (gleich)
  - HTML-Tag mit Stylesheets (CSS)
- Formatierung sollte mithilfe von Stylesheets vorgenommen werden
- Besondere HTML-Tags zum Festlegen des Aussehens der Webseite, z.B.
  `<strong>Wichtig</strong>` Auszeichnung als Fettdruck darstellen

- **Am besten:** Trennung von Form (Aussehen) und Inhalt
  - Inhalt mit HTML **`<p>`**
  - Form (Aussehen) *zentral* mit CSS **`p {color: maroon;}`**

- CSS später

HTML
Grundgerüst

```
<!DOCTYPE html>
<html>
  <head>

     <!-- Kopf-(Meta-)daten (Vereinbarungen) -->

  </head>
  <body>

   <!-- im Broser sichtbarer Dokumentbereich -->

  </body>
</html>
```

HTML
Dokumententyp

```
<!DOCTYPE html>
<html>
  <head>
```

- Dokumenttyp-Deklaration (hier für HTML5)
- für XHTML ist XML-Deklaration erforderlich:
  - `<?xml version="1.0" encoding="utf-8"?>`
  - `<html xmlns="http://www.w3.org/1999/xhtml">`
- alte Dokumenttyp-Deklarationen:
  - `<!DOCTYPE HTML PUBLIC "-//W3C/DTD HTML 4.01//EN" "http://www.w3.org/TR/HTML/ 4/strict.dtd">`
  - Regeln mit Hilfe des Markup-Rahmenwerkes SGML formuliert
  - dort für jeden Dokumententyp geregelt, welche Elemente enthalten sein dürfen (in den Dokumenttyp-Definitionen – DTD)
  - W3C ist Herausgeber der DTD

HTML
Metadaten

```
<!DOCTYPE html>
<html>
  <head>
     <!-- Kopf-(Meta-)daten (Vereinbarungen) -->
  </head>
  <body>
```

- Head-Element ist Container für (Meta-)Daten über das Webdokument. Das <head>-Element kommt in das <html>-Element und vor das <body>-Element
- Die Metadaten werden nicht dargestellt
- Typische HTML-Elemente für Metadaten sind:
  - **`<title>`** Titel des Dokumentes (im Tab und in der Such-Ergebnisliste,
  - **`<style>`** für Format-Angaben (CSS),
  - **`<meta>`** für die Festlegung von Zeichenkodierungen, Schlüsselwörter, Autor usw.,
  - **`<link>`** zum Einbinden externer CSS-Dateien,
  - **`<script>`** zum Definieren von Client-seitigen JavaScript-Funktionen,
  - **`<base>`** zum Festlegen, der URL, von der aus alle Pfadangaben relativ sind.

HTML

Metadaten – Beispiel (siehe metabeispiel.html)

```
<head>
   <meta charset="UTF-8">
   <meta name="description" content="meta data">
   <meta name="keywords" content="HTML, head, title, meta, link, style">
   <meta name="author" content="Jörn Freiheit">
   <meta http-equiv="refresh" content="30">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="mystyle.css">
   <style>
      body {background-color: #29e0e6;}      h1 {color: #ff6a3b;}      ul {color: #0000ff;}
   </style>
   <script>
      function myFunction() { document.getElementById("demo").innerHTML = "Hallo
FIW!";
      }
   </script>
   <base href="localhost/WT19" target="_blank">
   <title>Metadaten</title>
</head>
```

HTML
Attribute

- Alle HTML-Elemente können **Attribute** haben
- Attribute liefern zusätzliche Informationen zum Element bzw. spezifizieren es genauer
- Attribute gehören immer in den Start-Tag
- Attribute werden als Schlüssel-Werte-Paare angegeben (**key="value"**)

- allgemein: `<tag key1="value1" key2="value2"> inhalt </tag>`

- Beispiele: `<a href="http://www.htw-berlin.de">HTW</a>`

```
<img src="../img/fiw.jpg" alt="FIW-Logo" style="width:42px;"/>
<button onclick="myFunction()">Klick</button>
```

HTML
Attribute

- Arten von Attributen:
  - Attribute mit Wertzuweisung (es gibt bestimmte erlaubte Werte)

    z.B.: `<input type="button">`
  - Attribute mit freier Wertzuweisung aber mit Typkonvention

    z.B. `<style type="text/css">` (MIME-Typ Typ/Untertyp)

    `<select size="8">` (Auswahlliste num. Länge 8)
  - Attribute mit freier Wertzuweisung und ohne Typkonvention

    z.B. `<p title="Mouseover">` (beliebiger Text)
  - Alleinstehende Attribute (nur HTML)

    z.B. `<input type="checkbox" checked>` (HTML-Form)

    `<input type="checkbox" checked="checked">` (XHTML)

  - am besten immer XHTML-Form (Attribute klein schreiben, Anführungsstriche verwenden und keine alleinstehenden Attribute)

HTML
Kommentare

- keine Unterscheidung zwischen Zeilen- und Blockkommentar:
  - `<!-- Kommentar -->`
  - beginnt mit `<!--`
  - und endet mit `-->`

HTML
Hyperlinks

- Tag <a>
- Muss kein Text sein, kann auch Bild oder jedes beliebige andere HTML-Element
- allgemein: `<a href="url"> Text </a>`
- Beispiele (index.html)

```
<a href="http://www.htw-berlin.de">HTW Berlin (gleicher tab)</a>
<a href="http://fiw.htw-berlin.de" target="_blank" title="BSGE">FIW (neuer tab und tooltip)</a>
<a href="./seite1.html">Seite 1</a>
<a href="http://fiw.htw-berlin.de" target="_blank" title="BSGE">
    <img src="../img/fiw.jpg" alt="Logo SG FIW" style="width:42px;"/>
</a>
<a href="http://www.htw-berlin.de" target="_self">
    <img src="../img/htw.jpg" alt="Logo HTW Berlin“ style="width:42px;border:0;"/>
</a>
```

HTML
Verzeichnisse und Dateinamen

- wählen Sie eine sinnvolle Verzeichnisstruktur!
  - die Verzeichnisnamen tauchen häufig in der URL wieder auf
  - selbsterklärende Verzeichnisnamen

- Dateinamen:
  - möglichst nur **a-z**, **A-Z** und **0-9** verwenden (siehe URL – erlaubte Zeichen)
  - evtl. auch Bindestriche, Unterstriche und Punkt für Abtrennung der Dateiendung
  - keine Umlaute und keine Sonderzeichen (erlaubt, aber gefährlich)

- Relative Pfade! (Achtung bei absoluten Pfadangaben → ändern sich auf dem Server)

HTML
Zeichencodierung

- schalten Sie in Ihrem Editor die Zeichenkodierung UTF-8 ein

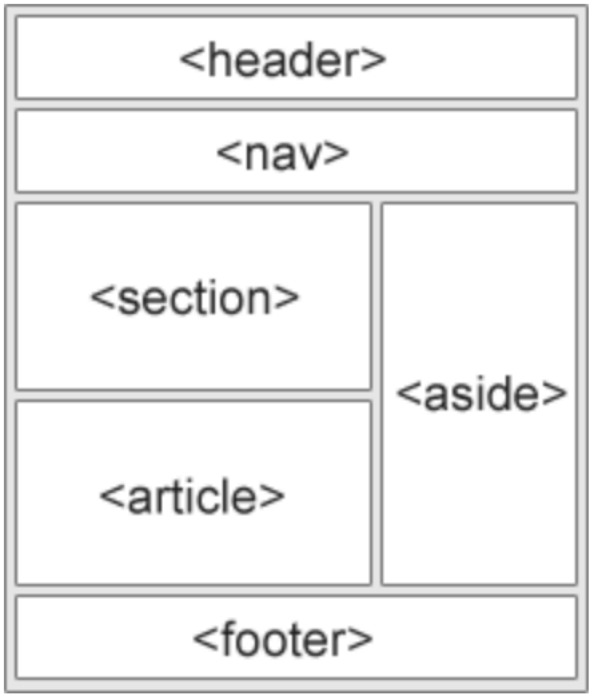
- wenn in Ihrem HTML-Text Zeichen aus der HTML-Syntax vorkommen:
  - Zeichen maskieren (named entities – benannte Zeichen)
  - z.B. **`<`** (lower than <)
    `>` (greater than >)
    `&` (ampersand &)
    `&quot;` (quot ")

- wenn Sie ein Zeichen brauchen, das nicht in UTF-8 ist:
  `&#xNummer` hexadezimale Nummer
  `&#Nummer` dezimal

siehe Unicode 7.0 http://www.unicode.org/charts/

HTML
Dokumentenstruktur

- Ein Container (ein Bereich eines Webdokumentes) war füher typischerweise ein
- Es entstanden sehr viele -Elemente mit einer eigenen Id, z.B. **`<div id="header">`**
- Mit HTML5 kamen weitere Strukturelemente hinzu:



- Außerdem noch (wachsend):
  - <details>
  - <figcaption>
  - <figure>
  - <main>
  - <mark>
  - <summary>
  - <time>

Quelle: w3schools.com - http://www.w3schools.com/html/html_layout.asp

HTML

Dokumentenstruktur – bereits mit Auswirkungen!

```
<body>
    <h1>Titel eines Buches</h1>
    <section>
        <h1>Erstes Kapitel</h1>
        <p>Inhalt des ersten Kapitels</p>
    </section>
    <section>
        <h1>Zweites Kapitel</h1>
        <p>Inhalt des zweiten Kapitels</p>
    </section>
</body>
```

- **<h1>** wird unterschiedlich groß dargestellt!
- mit **<section>** entsteht eine Art Unterdokument
- logische Hierarchie des Dokumentes

HTML

Dokumentenstruktur – Beispiel

```
<h1>Mein Blog</h1>
<section>
    <header>
        <h1> Kommentare </h1>
    </header>
    <article>
        <header>
            <h1> Artikel&uuml;berschrift</h1>
        </header>
            <p> Artikelinhalt</p>
        <footer>
            <p>Kommentar zum Artikel</p>
        </footer>
    </article>
</section>
```

- **<section> Abschnitt </section>**
  - bestehen typischerweise aus Überschrift und Inhalt
  - können verschachtelt werden
  - beeinflussen die hierarchische Logik

HTML

Dokumentenstruktur – Beispiel

- **<header> Kopfbereich </header>**
- **<footer> Footer </footer>**
  - beeinflussen nicht die hierarchische Logik
  - können auch jeweils für Sections definiert werden

- **<nav> Navigation </nav>**
  - Verhalten wie (neue Zeile)

- **<aside> Sidebar </aside>**
  - nicht für Navigation gedacht, eher für kurze Informationen, Nachrichten oder Werbung an der Seite
  - nicht automatisch an der Seite!

- **<article> Inhalt </article>**
  - eigenständiger logischer Teil einer Webseite (z.B. Blogeintrag)

- ** Paragraph **
  - Sollte kleinster Textblock sein (nur noch kleiner – inline)

HTML
Listen

- Wichtige Elemente zur Übersichtlichkeit (insb. lange Aufzählungen)
- Zur Erstellung zwei verschiedene Tags notwendig
  - Definition der Art der Liste
  - Definition der einzelnen Listenpunkte
- Listenarten
  - Ungeordnete Liste **`<ul>`**
  - Geordnete Liste **`<ol>`**
  - Definitionsliste **`<dl>`**
- Listeneinträge (list items) **`<li>`**

HTML
verschachtelte Listen

- **<ul>** zeichnet eine ungeordnete Liste aus
  - Listenpunkte **<li>** unterliegen keiner bestimmten Ordnung (gleichwertig)
  - Durch Verschachteln der Listenpunkte sind mehrere Ebenen möglich

```
<ul>
  <li>Liste</li>
  <li>mit</li>
  <li>Punkten</li>
</ul>
```

- Liste
- mit
- Punkten

```
<ul>
  <li>Liste</li>
  <li>mit
  <ul>
    <li>mehreren</li>
    <li>Unter-</li>
  </ul> </li>
  <li>Punkten</li>
</ul>
```

- Liste
- mit
  - mehreren
  - Unter-
- Punkten

*Listen enthalten nur -Elemente! (
…)*

HTML

Beispiel unordered list

```
<h3>Ungeordnete Liste</h3>
 <ul>
  <li>Kaffee
<ul>
<li>mit Milch</li>
<li>ohne Milch</li>
</ul>
</li>
  <li>Tee</li>
  <li>Milch</li>
</ul>
```

Ungeordnete Liste

- Kaffee
  - mit Milch
  - ohne Milch
- Tee
- Milch

```
<h3>Ungeordnete Liste</h3>
 <ul style="list-style-type:square">
  <li>Kaffee
<ul style="list-style-type:none">
<li>mit Milch</li>
<li>ohne Milch</li>
</ul>
</li>
  <li>Tee</li>
  <li>Milch</li>
</ul>
```

Ungeordnete Liste

- Kaffee
  - mit Milch
  - ohne Milch
- Tee
- Milch

- `list-style-type:disc` bullets (default)
- `list-style-type:circle` Kreise
- `list-style-type:square` Quadrate
- `list-style-type:none` keine

HTML

Beispiel ordered list

```
<h3>Geordnete Liste</h3>
 <ol type="A">
  <li>Kaffee
        <ol type="i" start="2">
              <li>mit Milch</li>
              <li  value="4">ohne Milch</li>
        </ol>
        </li>
  <li>Tee</li>
  <li>Milch
        <ol type="I" reversed="reversed">
              <li>pur</li>
              <li>mit Honig</li>
              <li>mit Kakao</li>
        </ol>
  </li>
</ol>
```

Geordnete Liste

A. Kaffee
   ii. mit Milch
   iv. ohne Milch
B. Tee
C. Milch
   III. pur
   II. mit Honig
   I. mit Kakao

HTML
Text formatieren (hervorheben)

- Formatierungen sind grundsätzlich Aufgabe von CSS → Trennen von Inhalt und Aussehen
- Aber Hervorhebungen von Text möglich:
  - **<b>** fettgedrucker Text (bold)
  - **** hervorgehobener Text (wichtig)
  - **<i>** kursiv geschriebener Text (italic)
  - **** hervorgehobener Text (emphasized)
  - **<mark>** markierter Text
  - **<small>** kleinerer Font als der umgebende Text
  - **<del>** durchgestrichener Text (deleted text)
  - **<ins>** eingefügter Text (unterstrichen)
  - **<sub>** tiefgestellter Text
  - **<sup>** hochgestellter Text
- → vordefinierte Einstellungen (Browser-Styles) werden verwendet
  → können mit eigenen Styles leicht überschrieben werden

HTML
Tabellen

Vorname	Nachname	Alter
Maria	Mustermann	50
Egon	Eimer	24

- Tabellen werden mithilfe des HTML-Elementes **<table>** erstellt
- Tabellenzeile: **<tr>**
- Eintrag in Überschrift: **<th>**
- Eintrag in Tabelle: **<td>**

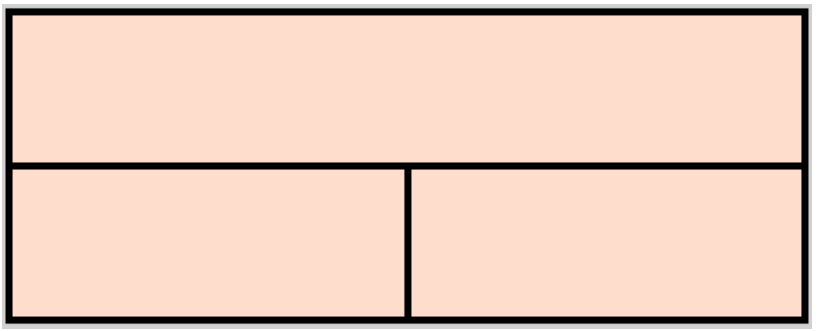
```
<table>
 <tr>
   <th>Vorname</th>
   <th>Nachname</th>
   <th>Alter</th>
 </tr>
 <tr>
   <td>Maria</td>
   <td>Mustermann</td>
   <td>50</td>
 </tr>
 <tr>
   <td>Egon</td>
   <td>Eimer</td>
   <td>24</td>
 </tr>
</table>
```

HTML

Tabellen – Attribute für Zellen

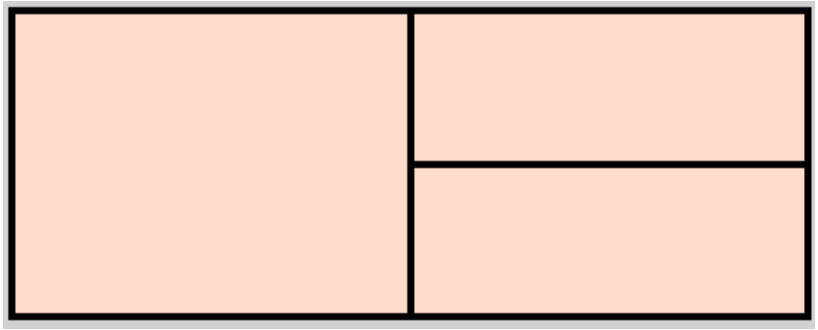
- **<th>** und **<td>** können als Attribute haben:
  - **NOWRAP**: Abstellen des automatischen Zeilenumbruchs
  - **COLSPAN**: Anzahl der Spalten, über die eine Zelle reicht
  - **ROWSPAN**: Anzahl der Zeilen, über die eine Zelle reicht
- Verbindung zweier Spalten:

```
<table>
  <tr>
    <td colspan="2"> … </td>
  </tr>
  <tr>
    <td> … </td>
    <td> … </td>
  </tr>
</table>
```



- Verbindung zweier Reihen:

```
<table>
  <tr>
    <td rowspan="2"> … </td>
    <td> … </td>
  </tr>
  <tr>
    <td> … </td>
  </tr>
</table>
```



HTML

Block- vs. Inline-Elemente

- Block-Elemente beginnen stets in einer neuen Zeile und nutzen die gesamte verfügbare Breite
- Alle Block-Elemente in HTML:

```
<address>
<article>
<aside>
<blockquote>
<canvas>
<dd>
<div>
<dl>
<dt>
<fieldset>
<figcaption>
<figure>
<footer>
<form>
<h1>-<h6>
<header>
<hr>
<li>
<main>
<nav>
<noscript>
<ol>
<output>
<p>
<pre>
<section>
<table>
<tfoot>
<ul>
<video>
```

HTML

Block- vs. Inline-Elemente

- Inline-Elemente starten nicht in einer neuen Zeile und nutzen nur soviel Breite wie notwendig
- Alle Inline-Elemente in HTML:

```
<a>
<abbr>
<acronym>
<b>
<bdo>
<big>
<br>
<button>
<cite>
<code>
<dfn>
<em>
<i>
<img>
<input>
<kbd>
<label>
<map>
<object>
<q>
<samp>
<script>
<select>
<small>
<span>
<strong>
<sub>
<sup>
<textarea>
<time>
<tt>
<var>
```

HTML
Konventionen

- Tag-Namen immer klein
- Attribute immer klein
- Alle HTML-Elemente schließen (Ende-tag oder **/>**)
- Attribute immer als Schlüssel-Werte-Paar
- Werte der Attribute immer in doppelten Hochkomma
- **<img>**-Element immer mit **alt**-Attribut