Web-Grundlagen Teil 2:
Einführung in HTML und CSS

Aufbau eines HTML-Dokuments

In Teil 1 haben wir bereits ein (fast) komplettes HTML-Dokument erstellt. Was noch gefehlt hat ist die Doctype-Angabe in der ersten Zeile des HTML-Dokuments. Diese wurde früher genutzt, um die verwendete HTML- oder XHTML-Version zu spezifizieren. Heute gibt sie einfach an, dass es sich um eine HTML-Datei nach dem aktuellen (Living) Standard handelt:

```
<!DOCTYPE html>
```

Um schnell ein erstes Gerüst einer HTML-Seite zu erstellen, können Sie im VSCode auf einer leeren HTML-Seite einfach `html:5` gefolgt von der Enter-Taste eingeben. Das Ergebnis sieht etwa so aus:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

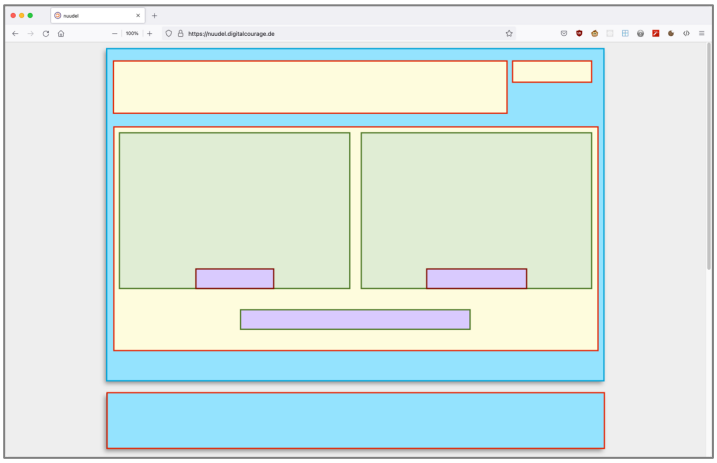
Nun muss nur noch der Titel angepasst werden und man kann beginnen, den Inhalt der HTML-Datei aufzubauen. Das hierfür in VSCode eingebaute Werkzeug heisst übrigens Emmet[1]. Es stellt zahlreiche weitere Möglichkeiten zur Verfügung, um HTML-Code zu erzeugen, und kann auch in einige andere Code-Editoren integriert werden.

Das Meta-Tag mit dem *http-equiv*-Attribut definiert einen bestimmten Kompatibilitätsmodus für Internet-Explorer-Browser. Es ist vor allem wichtig, wenn unsere Seite auch mit älteren Browsern kompatibel sein soll. Und das Meta-Tag mit dem *viewport*-Attribut ist wichtig für Geräte mit kleinem Bildschirm. Es besagt, dass die Seite sich selbst um Geräte mit kleinem Bildschirm kümmert und der Browser die Seite daher nicht verkleinert darstellen soll. Hier sind wir im Bereich des *responsiven Webdesigns*, einem Thema, mit dem wir uns in diesem Tutorial und in WBE nicht beschäftigen können. Es wird aber im Wahlfach *Mobile Applications* behandelt.

[1] https://emmet.io

Inhalt strukturieren

Die logische Struktur einer Webseite besteht aus Blocks, welche ineinander verschachtelt sein können. Dadurch entsteht eine hierarchische Struktur, die normalerweise auch im Design einer Webseite sichtbar ist. Hier ein Beispiel:



Wichtig: Die visuelle Struktur der gestalteten Website ist eigentlich unabhängig von der logischen Struktur des HTML-Codes, da jedes Element des HTML-Codes mittels CSS beliebig auf der Seite platziert werden kann. Trotzdem gibt es natürlich in der Regel gewisse Entsprechungen zwischen logischer und visueller Struktur.

Vor HTML5 stand für die Grobstruktur eines HTML-Dokuments hauptsächlich das *div*-Element zur Verfügung. Um näher zu spezifizieren, was mit den einzelnen *div*-Elementen gemeint ist, behalf man sich mit *class*- oder *id*-Attributen, zum Beispiel:

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body>
    <div class="header">...</div>
    <div class="nav">...</div>
    <div class="aside">...</div>
    <div class="main">
      <h1>Überschrift</h1>
      <p>Absatz</p>
    </div>
    <div class="footer">...</div>
  </body>
</html>
```

Die *div*-Elemente bilden hier die Grobstruktur des Dokuments. Sie können selbst weitere *div*-Elemente enthalten. Das *div*-Element dient einfach zur Gruppierung weiterer Elemente, es hat keine weitere Bedeutung. Für Überschriften werden *h1* (oberste Überschrift) bis *h6* (Überschrift mit niedrigstem Rang) und für Absätze *p*-Elemente verwendet.

Die *class*-Attribute – deren Wert frei gewählt werden kann – werden im Beispiel zur näheren Beschreibung der *div*-Elemente verwendet. Über die Attribute kann ein Element für CSS oder JavaScript bei Bedarf von anderen *div*-Elementen unterscheiden. Seit HTML5[2] stehen für die grobe Struktur eines Dokuments spezielle Elemente zur Verfügung. Das Beispiel sieht dann so aus:

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body>
    <header>...</header>
    <nav>...</nav>
    <aside>...</aside>
    <main>
      <h1>Überschrift</h1>
      <p>Absatz</p>
    </main>
    <footer>...</footer>
  </body>
</html>
```

Im Gegensatz zu *div* haben diese Elemente eine Bedeutung, sie werden daher auch als *semantische* Elemente bezeichnet. Wann immer ein passendes semantisches Element existiert, sollte dies der Verwendung eines *div*-Elements vorgezogen werden.

Aufgaben:

- Verschaffen Sie sich einen Überblick über die semantischen Elemente, zum Beispiel hier: https://www.w3schools.com/html/html5_semantic_elements.asp
- Nehmen Sie das Beispieldokument oben (Sie finden es unter in der Datei *struktur.html*) und füllen Sie es mit etwas Inhalt: Texte, weitere Überschriften, ein Bild. Wenn Sie Text benötigen, können Sie im VSCode einfach «lorem» eingeben. Emmet wird dies zu einem ganzen Absatz «Lorem ipsum dolor sit…» erweitern. Das nennt man *Blindtext*. Solche Platzhaltertexte werden häufig verwendet, wenn der definitive Text noch nicht zur Verfügung steht.[3]
- Testen Sie die Seite im Browser. Sie werden feststellen, dass das *aside*-Element nicht etwa dazu führt, dass das Element seitlich verschoben wird. Es ist zunächst nur ein Hinweis auf die Bedeutung des Elements. Wie es gestaltet und wo es positioniert wird, ist dann mit CSS zu regeln.

Kommentare

In den HTML-Code einer Seite können Kommentare eingefügt werden. Solche Kommentare werden mit der Seite ausgeliefert, der Browser wird sie aber nicht anzeigen. Kommentare eignen sich zur

[2] Ursprünglich wurden die einzelnen HTML-Versionen mit Versionsnummern versehen. HTML 5.3 war die letzte Versionsnummer, welche vom W3C (World Wide Web Consortium) für HTML vergeben wurde. Mittlerweile wird die Bezeichnung *HTML – Living Standard* verwendet oder einfach nur *HTML*.

[3] Weitere Blindtexte finden Sie hier: https://www.blindtextgenerator.de

Beschreibung bestimmter Teile des HTML-Codes. Sie eignen sich nicht, um Informationen zu verbergen, da jederzeit der Quellcode zu einer Seite angezeigt werden kann. Ein Kommentar wird folgendermassen eingefügt:

```
<!-- Das ist ein Kommentar. -->
```

Aufgabe:

- Fügen Sie Ihrem Dokument einen Kommentar hinzu und testen Sie die Ausgabe im Browser.

Verweise

HTML-Dokumente sind Hypertext-Dokumente. Damit ist gemeint, dass es einen Mechanismus gibt, auf andere Stellen innerhalb eines Dokuments oder in einem anderen Dokument zu verweisen und mit diesen Verweisen auf einfache Weise zu navigieren.

Ein Verweis wird in HTML mit dem *a*-Element angelegt (*a* steht für *anchor*). Das *href*-Attribut gibt das Verweisziel an. Beispiel:

```
<a href="andere_datei.html">Zum anderen Dokument</a>
```

Das Verweisziel kann als relativer Pfad oder absoluter Pfad angegeben werden. Wenn nur der Dateiname angegeben ist bedeutet das, dass die Zieldatei im gleichen Verzeichnis liegen muss. Als Pfad-Trennzeichen wird wie unter Unix der Schrägstrich verwendet. Die Wurzel eines absoluten Pfads ist ein Verzeichnis, das im Webserver als Web-Root konfiguriert ist. Als Verweisziel kann auch eine komplette Web-Adresse angegeben werden:

```
<a href="https://www.zhaw.ch/de/studium/bachelorstudiengaenge/">Zu den
Bachelorstudiengängen</a>
```

Was hier zum Thema Pfade geschrieben wurde, gilt auch für Bilder. Angenommen ein Logo befindet sich in einem Verzeichnis *images*, welches im übergeordneten Verzeichnis liegt. Dann kann es so mit einer relativen Pfadangabe referenziert werden.

```
<img src="../images/logo.jpg">
```

Aufgaben:

- Duplizieren Sie Ihre Übungsdatei und machen Sie in der neuen Datei ein paar Anpassungen.
- Ergänzen Sie im Navigationsbereich der beiden Dateien Verweise, um durch Anklicken der Verweise zwischen den beiden Dateien hin- und herspringen zu können.
- Testen Sie das Resultat im Browser.

Block- und Inline-Elemente

Die in den vorangehenden Aufgaben und Beispielen verwendeten HTML-Elemente lassen sich in zwei Gruppen einteilen:

- **Block-Elemente** nehmen die ganze verfügbare Breite in Anspruch. Zu ihnen gehören: Absätze (*p*) sowie die Überschriften (*h1...h6*). Ausserdem gehören die beschriebenen semantischen Elemente wie *article* zu den Block-Elementen.
- **Inline-Elemente** können innerhalb des Textflusses auftreten. Dazu gehören Textauszeichnungen wie *b* und *i* sowie Bilder (*img*) und Verweise (*a*).

Aufgaben:

- Erstellen Sie in einem der Beispieldokumente einen Absatz, der sowohl Text als auch ein kleines Bild enthält (notfalls nehmen Sie ein grosses Bild und verkleinern es mit den beschriebenen Attributen). Können Sie verifizieren, dass *img* ein Inline-Element ist?
- Fügen Sie mitten im Text eines Absatzes eine Überschrift ein. Wie wird das im Browser dargestellt? Ist es noch korrektes HTML? Überprüfen Sie es mit dem HTML-Validator, den Sie über die Web-Developer-Erweiterung leicht aufrufen können.[4]

Listen

In HTML gibt es verschiedene Varianten von Listen. Eine einfache Punkteliste könnte wie folgt dargestellt werden:

- eins
- zwei
- drei

Im Quellcode wird dies mit dem *ul*-Element (steht für „unordered list“) erreicht. Die einzelnen Listenpunkte werden mit *li* („list item“) ausgezeichnet:

```
<ul>
  <li>eins</li>
  <li>zwei</li>
  <li>drei</li>
</ul>
```

Das Einrücken des Codes dient der Lesbarkeit. Auf die Darstellung im Browser hat es keine Auswirkungen. Dennoch ist es empfehlenswert, lesbaren und übersichtlichen HTML-Code zu schreiben.

Neben den ungeordneten Listen gibt es für geordnete Listen das Element *ol* („ordered list“). Normalerweise werden diese nummeriert dargestellt. Ferner gibt es Definitionslisten mit *dl*, *dd* und *dt*.

[4] Oder über die Website https://validator.w3.org

Mit Hilfe von geordneten oder ungeordneten Listen werden auch Navigationsleisten umgesetzt. Die Darstellung (ohne Listenpunkte, ggf. horizontal statt vertikal angeordnet) wird mit CSS beschrieben.

Aufgaben:

- Sehen Sie sich auf der Seite *nuudel.digitalcourage.de* den Quellcode der FAQ an. Hier wird eine Definitionsliste verwendet.
- Es ist üblich, die Links im Navigationsbereich als Liste von Verweisen anzulegen. Passen Sie den Navigationsbereich Ihrer Beispielseiten entsprechend an.

Attribute

In HTML-Elementen sind uns schon an verschiedenen Stellen Attribute begegnet, etwa *href* in Verweisen, um das Ziel des Verweises anzugeben:

```
<a href="andere_datei.html">Zum anderen Dokument</a>
```

Die korrekte Schreibweise ist: Name des Attributs, Gleichheitszeichen, Wert des Attributs in Anführungszeichen. Vor und nach dem Gleichheitszeichen darf kein Leerzeichen stehen.

Manche Attribute haben keinen Wert, sie sind einfach vorhanden oder eben nicht. Das Attribut *contenteditable* zum Beispiel sorgt dafür, dass ein Element im Browser änderbar ist:

```
<p contenteditable>Dieser Absatz kamm geändert werden.</p>
```

Viele Attribute können nur in bestimmten Elementen vorkommen. Teilweise sind Attribute für bestimmte Elemente auch vorgeschrieben. Ein Bild-Element *img* muss zum Beispiel immer ein *src*- und ein *alt*-Attribut haben.

Die Attribute *id* und *class* sind sogenannte *Universalattribute*, sie können praktisch in jedem HTML-Element vorkommen. Auch ein Bild, eine Überschrift oder ein Absatz kann eine *id* oder eine *class* haben. Mit *id* wird ein Element eindeutig identifiziert. Jede *id* darf daher nur einmal in einem Dokument vorkommen. Mit dem *class*-Attribut wird ein Element einer Klasse von Elementen zugeordnet. Im Gegensatz zur *id* kann eine Klasse auch mehrfach in einem Dokument vorkommen. Weitere Universalattribute sind *style* und *title*.

Aufgaben:

- Öffnen Sie die Seite https://htmlreference.io. Die Seite enthält eine Übersicht von HTML-Elementen. Sehen Sie sich die Beschreibung des a-Elements an. Welche Attribute sind vorgeschrieben, welche sind fakultativ? Sehen Sie auch die Beschreibung für ein paar weitere bereits verwendete Elemente an.
- Eine weitere Seite mit Referenzen ist https://devdocs.io/. Öffnen Sie die Seite und suchen Sie nach Informationen über das *title*-Attribut (Suchfeld: html → TAB → title). Das *title*-Element kennen wir bereits. Es gibt aber auch ein Attribut *title*.

- Machen Sie einen Absatz in einem Ihrer Beispieldokumente editierbar und testen Sie die Seite im Browser.

Das span-Element

Wie *div* ist auch *span* ein Element ohne vorgegebene Bedeutung. Im Unterschied zu *div* ist *span* aber ein Inline-Element. Es dient dazu, Textteile mit einer Bedeutung zu versehen, für die kein vordefiniertes Element existiert. Beispiel:

```
<p>Dieses Beispiel ist <span class="warnung">Vorsicht</span> zu geniessen</p>
```

Weitere Inline-Elemente zur Textauszeichnung:

strong	Defines **strong importance** on text.
em	Defines **emphasis** on text. Is usually rendered as *italic* text.
mark	Defines **highlighted text**.

Aufgabe:

- Ergänzen Sie eine Ihrer Beispielseiten um Elemente zur Textauszeichnung. Wie werden diese standardmässig im Browser dargestellt? Hat auch *span* eine vorgegebene Darstellung?

Seite in HTML umsetzen

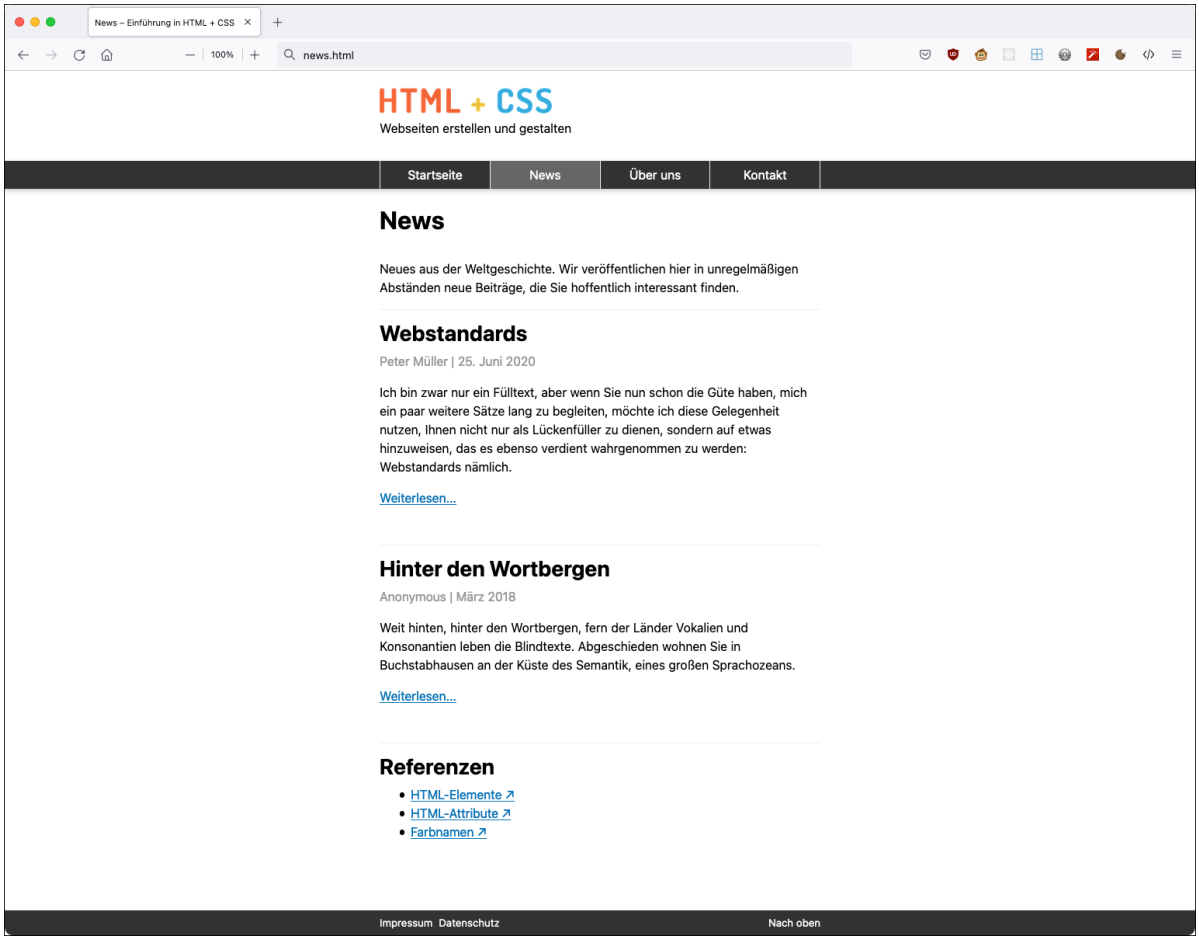
Den HTML-Code für eine Seite schreiben heisst: Den Inhalt strukturieren, die logische Struktur des Dokuments herausarbeiten. Dabei sollte man noch nicht an die Darstellung denken. Eine Aussage wie „Das Firmenlogo soll oben rechts erscheinen“ spielt beim Schreiben des HTML-Codes keine Rolle. Hier geht es darum, ob das Firmenlogo inhaltlich zum Header der Firmenbeschreibung gehört oder zu einem anderen Teil der Seite.

Häufig bekommt man als Webdesigner einen Seitenentwurf (zum Beispiel als Photoshop-Dokument) vom Grafiker oder Kunden. Die erste Aufgabe des Webdesigners ist dann, Inhalt und Darstellung voneinander zu trennen. Beim Schreiben des HTML-Codes muss man die Gestaltung der Seite also „wegdenken“, was nicht immer ganz einfach ist.

Aufgabe (empfohlen):

- An dieser Stelle ist es sinnvoll, den HTML-Code für eine Seite zu schreiben. Nehmen Sie sich als Beispiel eine Seite vor, wie sie das folgende Bild zeigt. Versuchen Sie, den HTML-Code dafür zu schreiben. Erstellen Sie auch einfache Seiten «Startseite», «Über uns» und «Kontakt», welche über die Navigation erreichbar sind. Konsultieren Sie bei Bedarf die Referenzen im letzten Abschnitt dieses Dokuments.

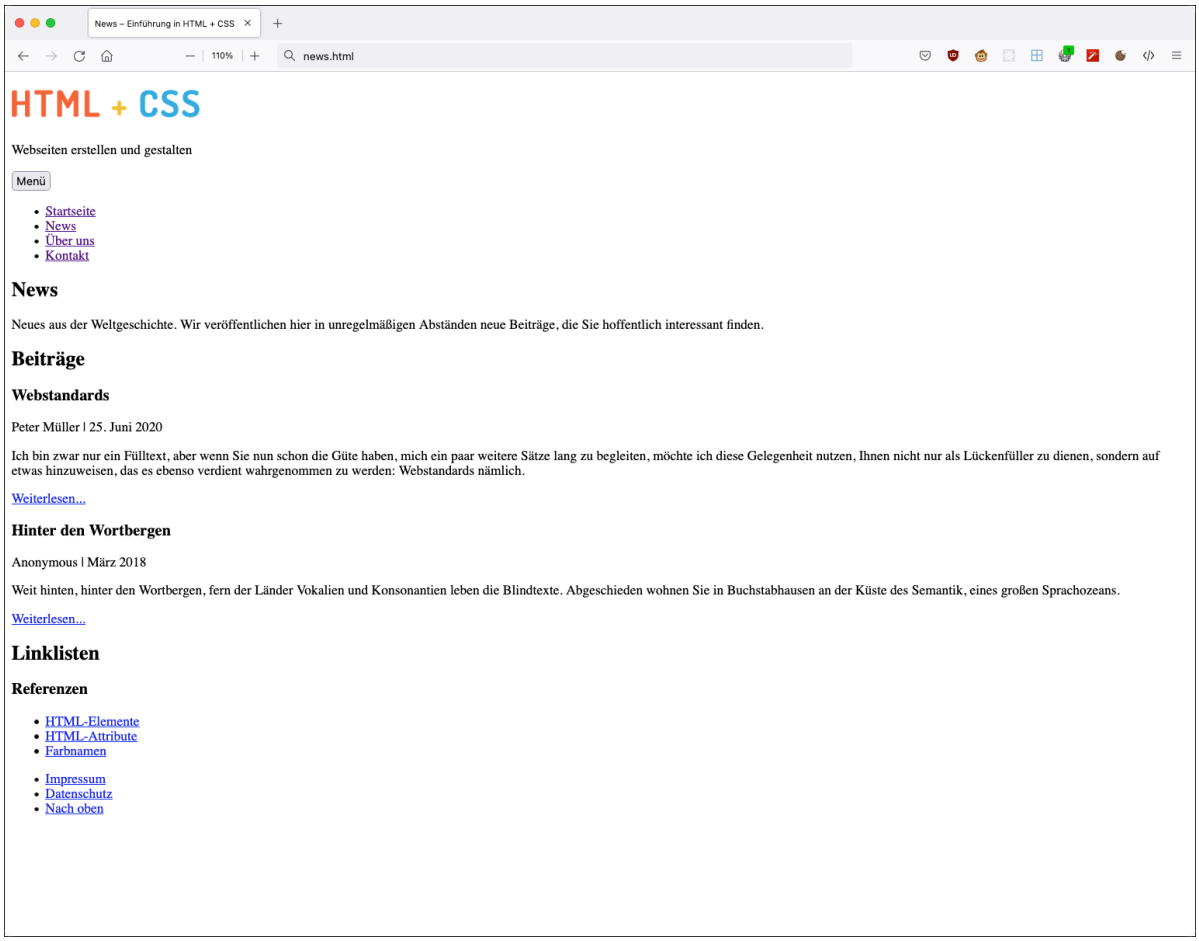
Die Seite soll einmal so aussehen, wie auf diesem Bild gezeigt. Das Beispiel stammt aus dem Buch «HTML + CSS – Webseiten erstellen und gestalten» von Peter Müller (Rheinwerk-Verlag, 2020): https://html-und-css.de/buch/



Natürlich erlaubt reiner HTML-Code noch keine solche Darstellung. Dafür ist sicher CSS nötig. Ohne CSS wird ihre Seite erst einmal eher so aussehen, wie auf dem folgenden Bild gezeigt. Folgendes ist festzustellen:

- Die Seite ist auch ohne CSS schon verwendbar, auch wenn sie natürlich nicht so schön aussieht.
- Mit ausgeschaltetem CSS-Code kann man die Struktur der Seite gut erkennen.

Die gleiche Seite ohne CSS:



HTML-Referenzen

- https://htmlreference.io
  Liste von HTML-Elementen mit Beschreibung
- https://devdocs.io/
  Verschiedene Referenzen, unter anderem auch für HTML
- https://developer.mozilla.org/en-US/docs/Web/HTML/Reference
  HTML-Referenzen des Mozilla Developer Network
- https://html.spec.whatwg.org/multipage/
  HTML Living Standard, der aktuelle Stand von HTML