1. Grundaufbau

HTML (Hyper Text Markup Language) ist eine Auszeichnungssprache, mit der die einzelnen Bestandteile eines WEB-Dokuments formatiert werden. Die hier vorgestellte (X)HTML5 Version bezieht sich auf die W3C-Empfehlung. Zur Beschreibung dienen (auch verschachtelt) Tags, die durch spitze Klammern gekennzeichnet werden. Zu jedem Start-Tag (außer Standalone Tag's) gehört ein End-Tag.
Eine Auszeichnung (Markup) sieht schematisch folgendermaßen aus:

```
<Element-Name [attribut="wert" ...] > Inhalt </Element-Name>
```

Jede HTML – Seite sollte mindestens nebenstehenden Grundaufbau haben. Zeile ① beschreibt hierbei die DTD für HTML5 und kann auch durch andere gültige DTD's ersetzt werden (in weiten Bereichen ist valides HTML5 auch HTML 4 konform).

WICHTIG : Nicht alle Browser unterstützen die W3C-Empfehlung gleichermaßen, so dass gleiche Seiten auf verschiedenen Browsern zu unterschiedlichen Darstellungen führen können.

```
① <!DOCTYPE html>
② <html lang="de" >
③     <head>
④         <meta charset="utf-8" >
⑤         <title>Seitentitel</title>
⑥     </head>
⑦     <body>
⑧         Inhalt der Seite
⑨     </body>
⑩ </html>
```

start.html

2. Textformatierungen und Strukturierung

Mit den nachfolgenden Formatierungs-Tags kann die inhaltliche Struktur eines Dokumentes beschrieben werden. Dabei dient die optische Darstellung nur der inhaltlichen Unterscheidbarkeit. Zur optischen Gestaltung (Präsentation) von WEB-Dokumenten dient das style-Attribut bzw. eine CSS-Formatierung.

Textformatierung	Bedeutung	Darstellung
Text **** Beispiel **** Text	fette Darstellung	Text **Beispiel** Text
Text **~~** Beispiel **~~** Text	gelöschter Text	Text ~~Beispiel~~ Text
Text **** Beispiel **** Text	kursive Darstellung	Text *Beispiel* Text
Text **<mark>** Beispiel **</mark>** Text	markiert Text	Text Beispiel Text
Text **<pre>** Beispiel **</pre>** Text	nichtproportionale Schrift	Text ↵ Beispiel ↵ Text
Text **** Beispiel **** Text	kleiner Text	Text Beispiel Text
Text **_{** Beispiel **}** Text	tiefer gestellte Darstellung	Text $_{\text{Beispiel}}$ Text
Text **^{** Beispiel **}** Text	höher gestellte Darstellung	Text $^{\text{Beispiel}}$ Text

Textstrukturierung	Bedeutung
…	Kennzeichnung von Textabsätzen
  und <wbr />	tatsächlicher bzw. optionaler Zeilenumbruch
<hr />	horizontale Linie
… bzw. …	Kennzeichnung von Abschnitten (div → Block / span → Inline)
… bis …	Überschriftenhierarchien
<code>…</code> und <kbd>…</kbd>	Code (Quelltext) und Tastatureingabe

3. Tabellen

Tabellen (eingeleitet mit <table>) bestehen aus Zeilen <tr> und innerhalb jeder Zeile der gleichen Anzahl von Zellen <td>, die optional auch als Überschrift <th> gekennzeichnet werden können. Zellen können auch leer sein, sollten aber zur korrekten Darstellung ein Leerzeichen (<td> </td>) enthalten. Nachstehend werden die Deklaration und das Ergebnis einer HTML-Tabelle dargestellt.

```
① <table>
② <tr><th> AA </th><th> BB </th><th> CC </th></tr>
③ <tr><td> 11 </td><td> 12 </td><td> 13 </td></tr>
④ <tr><td> 21 </td><td> 22 </td><td> 23 </td></tr>
⑤ <tr><td> 31 </td><td> 32 </td><td> 33 </td></tr>
⑥ </table>
```

tabelle.html

AA	BB	CC
11	12	13
21	22	23
31	32	33

Zur weiteren logischen Unterteilung (besonders bei umfangreichen Tabellen) dienen nachfolgende Tags.

Tabellen-Tags	Bedeutung
<thead>…</thead>	Kennzeichnet einen Tabellenkopf und kann mehrere Zeilen enthalten.
<tfoot>…</tfoot>	Der Tabellenfuß muss direkt nach dem Tabellenkopf deklariert werden.
<tbody>…</tbody>	Kennzeichnet den eigentlichen Datenbereich (auch mehrfach möglich).

In den Tags **<tr>** und **<td>** sind die Attribute **colspan=**"integer" und/oder **rowspan=**"integer" zulässig, mit denen mehrere Zellen zeilenweise und/oder spaltenweise zusammengefasst werden können.

4. Nummerierungs-, Aufzählungs- und Definitionslisten

Mit den unterschiedlichen Listenarten wird die Strukturierung von Aufzählungen erreicht.

Nebenstehend sind einige Gestaltungsmöglichkeiten abgebildet.

Listen-Tags	Bedeutung
...	umschließt eine nummerierte Liste
	zulässige Attribute:
reversed = "reversed"	rückwärts zählend
start = "nummer"	Startwert
value = "nummer"	Nummerierung festlegen
...	umschließt eine Aufzählungsliste
<menu> ... </menu>	Umschließt eine Liste mit inter-aktiven Optionen oder Befehlen
...	Kennzeichnet die Listenelement
<dl> ... </dl>	Umschließt eine Definitionsliste
<dt> ... </dt>	Kennzeichnet die Abkürzung
<dd> ... </dd>	Kennzeichnet die Beschreibung

Bild - Formate im Netz

1. JPG - am weitesten verbreitet
2. PNG - nimmt langsam zu
3. GIF - kommt seltener vor

- JPG (bei hoher Farbtiefe)
- PNG (für Rastergrafiken)
- GIF (bei geringe Farbtiefe)

- JPG-Test
- PNG-Test
- GIF-Test

JPG — Joint Photographic Experts

PNG — Portable Network Graphics

GIF — Graphics Interchange Format

5. Verweise auf Ressourcen (Hyperlinks)

Durch die Verwendung von Links (Verweisen) wird das Strukturieren von bzw. Navigieren in größeren Dateien erleichtert und vereinfacht. Dabei verweist ein Hyperlink auf eine Ressource, die durch eine URL adressiert wird.

5.1 Aufbau einer URL

Eine URL (Uniform Resource Locater) hat nachfolgenden schematischen Aufbau mit optionalen Angaben.

```
Schema://Server[:Port][/Pfad ...][Datei][#Anker][?GET-Parameter]
```

Bei dieser Angabe spricht man von absoluter Adressierung, da das Ziel unabhängig von der Adressposition angesprochen wird. Dem gegenüber wird bei der relativen Adressierung nur der Pfad von der Adressposition zur Zielposition angegeben. Dabei kann mit /Verzeichnis/Verzeichnis/... in Unterverzeichnisse, bzw. mit ../../ in Überverzeichnisse navigiert werden. Bei WEB-Projekten ist der relativen Adressierung den Vorzug zu geben.

5.2 Hyperlinks im selben Dokument (Anker)

Um innerhalb eines Dokumentes zu verlinken, muss das Ziel gekennzeichnet werden. Dazu wird an dieser Textstelle ein Anker gesetzt. Zeile ① zeigt eine solche Definition einer Textmarke, die mit Zeile ② referenziert wird (Anker und Adressen sind CASE-sensitiv!).

```
① <a id = "Anker_01"> Zieltextstelle </a>
② <a href = "#Anker_01"> Hier geht es zur Zieltextstelle </a>
```

Link1.html

5.2 Links in andere Dokumente

Ist das Linkziel eine andere Ressource auf dem selben Server (ggf. in anderen Verzeichnissen), kann dieses mit einer relativen Adressierung ① referenziert werden. Handelt es sich dabei um ein HTML-Dokument sind auch hier Verweise auf Ankerziele ② möglich, die gekennzeichnet seine müssen ③.

```
① <a href = "verzeichnis/Link3.html"> Beschreibung des Verweises</a>
② <a href = "verzeichnis/Link3.html#Anker_01"> Beschreibung des Verweises</a>
```

Link2.html

```
③ <a id = "Anker_01"> Zieltextstelle </a>
```

Link3.html

5.3 Links auf externe Ressourcen

Für alle anderen Verweise wird die absolute Adressierung verwendet (bei der die URL dem angegebenen Schema entsprechen muss).

```
① <a href = "http://adresse.de/Link4.html" > Textausgabe für Link 4 </a>
② <a href = "http://192.168.10.16/~user/ Link4.html " > Textausgabe für Link </a>
```

Link4.html

Auch andere Ressourcen können nach diesem Adressierungsprinzip referenziert werden:

TAG – Elemente zur Ressourceneinbindung	Bedeutung
`<img src = "bild.gif" />`	Lädt ein Bild.
`<script type = "text/javascript" src = "script.js" />`	Importiert eine externe JavaScript-Datei.
`<video src = "movie.ogg" > Fehlermeldung </video>`	Bindet ein Video ein.

6. Formulare

Die Erstellung von Formularen dient zur Dialogerzeugung (Interaktivität) einer WEB-Seite mit dem User. Wichtig ist, dass in HTML keine Möglichkeit existiert, Formularinhalte selbst zu verarbeiten (nur einfache clientseitige Validierung), sondern nur Inhalte zu sammeln und an andere Instanzen weiterzuleiten.

6.1 Formulardeklarationen

Das form-Tag ① leitet eine Formulardeklaration, wird mit dem End-form-Tag ② abgeschlossen und muss alle Elemente umschließen. Des Weiteren können Ziel-URL, Übertragungsart und Codierung festgelegt werden. :

```
① < form action="Ziel URL" [method="post | get"] [enctype="multipart/form-data"] >
②     . . . Formularkörper mit allen Eingabe-Elementen . . . </form>
```

6.2 semantische Beschriftungen von Formularelementen

Mit dem **<label>**-Element werden Eingabefelder beschriftet. Die Zuordnung erfolgt entweder als umschließendes Elternelement ① oder durch die Verknüpfung des **for**-Attributs mit der **id** des Eingabeelementes ②.

```
① <label for="input-id" > Beschriftung </label>
        <input type="text" name="name1" id="input_id" />
② <label> Hinweistext <input type="text" name="name2" /> </label>
```

6.3 Eingabe-Elemente

Bei allen Elementen ist das **name** -Attribut für die Datenübertragung zwingend notwendig. Die optische Darstellung und der Umfang der Implementierung sind vom Browser und der Browserversion abhängig und können variieren.

HTML – Tag und Attribute	Darstellung und Beschreibung

`<input type="text" name="E1"`
`placeholder="Hinweis" required`
`pattern="[A-Z,a-z]{3}[0-9]{2}"`
`title="Bitte 3 Buchstaben und 2 Ziffern eingeben" />`
	einfache Textzeile (mit optionalen Attr.)
	placeholder zeigt einen Eingabehinweis
	required erzwingt eine Anwendereingabe
	pattern validiert nach regulärem Ausdruck
	Hinweis

`type="url|email|date|number|tel|time ..."`	**Textzeile(n)** mit spezieller clientseitiger Validierung
`<input type="password" name="E2" />`	**Passwordbox** Eingabezeichen sind maskiert •••••••

`<input type="checkbox" name="E3" />`
`<input type="checkbox" name="E4" />`
`<input type="checkbox" name="E5" />`
	Checkbox
	Auswahl : m aus n
	(value ="..." ⇒ übertragener Wert)
	Wert 1 ☐
	Wert 2 ☑
	Wert 3 ☑

`<input type="radio" name="E6" value="1" />`
`<input type="radio" name="E6" value="2" />`
`<input type="radio" name="E6" value="3" />`
	Optionbutton
	Auswahl : 1 aus n
	(Namen müssen gleich sein)
	Wert A ○
	Wert B ◉
	Wert C ○

`<input type="submit" />`	**Button (Submit)** Datenübertragung (notwendig) Daten absenden
`<input type="reset" />`	**Button (Reset)** löscht Eingaben (optional) Zurücksetzen
`<input type="button" value="mach was" />`	**Button (Allgemein)** (z.B.: mit JS) frei verfügbar mach was
`<input type="hidden" value="x" name="E6" />`	**Hidden Feld** dient (nicht sichtbar) zur Wertübergabe.
`<input type="file" name="E7" />`	**File- Dialog** mit enctype ! Durchsuchen... Keine Datei ausgewählt.

`<textarea name="E8" cols="15" rows="4">`
`Mehrzeilige Eingabe mit Zeilenumbruch`
`</textarea>`
	Textfeld
	Größe und Scrollbalken
	sind wahlweise einstellbar.
	Mehrzeilige Eingabe mit Zeilenumbruch

`<select name="E9" [multiple] [size="1"] >`
`<option value="1">Wert 1</option> ...`
`</select>`
	ListBox oder ComboBox
	(Auswahl 1 aus n)
	size bestimmt die Darstellung
	multiple ermöglicht Auswahl m aus n
	Wert 1
	Wert 1 / Wert 2 / Wert 3 / Wert 4

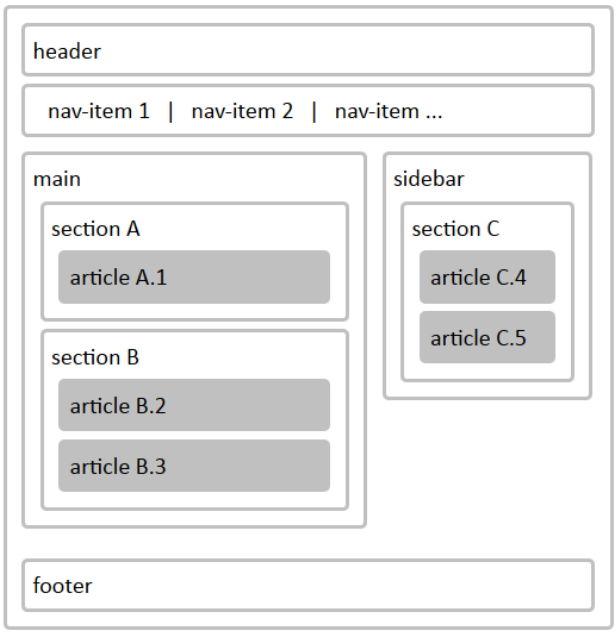
7. Semantik-Elemente zur Seitenstrukturierung

Mit HTML 5 wurde HTML um die Möglichkeit erweitert, mit semantischen Elementen die Struktur einer Seite besser auszuzeichnen. Diese Tags ersetzen zum Teil die bisherigen ** -**Konstruktionen und können wie alle bisherigen HTML-Tags wie auch ohne Zuweisung einer ID oder Klasse mit CSS formatiert werden. Die Tags erlauben es den Suchmaschinen, den Inhalt und die Struktur der Webseite besser zu erfassen.

HTML – Tag	Beschreibung
<body>	Steht für den sichtbaren Bereich eines HTML-Dokumentes (kann nur einmal enthalten sein).
<main>	Definiert den Hauptinhalt der Seite. Es ist nur ein <main> Element pro Seite zulässig.
<nav>	Beschreibt einen Abschnitt der ausschließlich Navigationslinks enthält.
<section>	Beschreibt einen Abschnitt eines Dokumentes.
<article>	Beschreibt eigenständigen Inhalt, der unabhängig von den übrigen Inhalten sein kann.
<address>	Definiert einen Abschnitt mit Kontaktinformationen.
<header>	Definiert den Kopfteil ("header") einer Seite oder eines Abschnittes. Er enthält oft ein Logo, den Titel der Website und die Seitennavigation.
<footer>	Definiert den Fußteil ("footer") einer Seite oder eines Abschnittes. Er enthält oft Copyright-Hinweise, einen Link auf das Impressum oder Kontaktadressen.
<aside>	Steht für eine Randbemerkung. Der übrige Inhalt sollte auch verständlich sein, wenn dieses Element entfernt wird.

```
   :
①  <header>header</header>
②  <nav>
     <li> nav-item 1 </li>
     <li> nav-item 2 </li>
     <li> nav-item ... </li>
   </nav>
③  <main> main
④    <section> section A
⑤      <article> article A.1 </article>
     </section>
     <section> section B
       <article> article B.2 </article>
       <article> article B.3 </article>
     </section>
⑥  </main>
   <aside> sidebar
     <section> section C
       <article> article C.4 </article>
       <article> article C.5 </article>
     </section>
   </aside>
⑦  <footer> footer </footer>
   :
```

HTML-Code (Auszug) eines typischen Layouts



Hinweis : Mit den HTML-Semantik-Elementen wird keine Layout realisiert. Dieses wird weiterhin über entsprechende CSS- Anweisungen gestaltet.

Hier kurz:

```
li      { display : inline;}
main    { width : 60% ; float : left ; }
aside   { width : 32% ; float : left ; }
footer  { clear : both ; }
```

sowie diverse `margin`-Angaben für den Abstand.

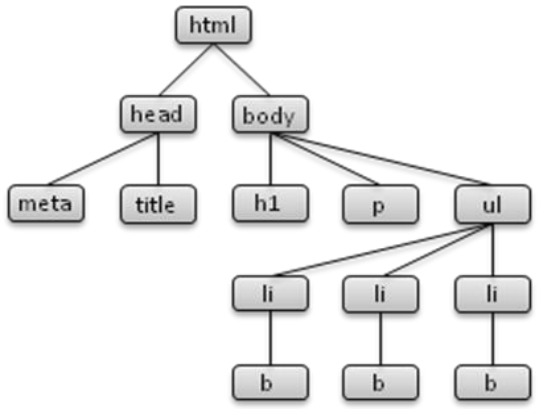
8. Das DOM (Document Object Model)

Im DOM wird das gesamte Markup eines HTML-Dokuments als Hierarchie aus nodes (zu Deutsch: Knoten) betrachtet. Dabei wird eine vereinheitlichte Zugriffssyntax (W3-DOM) spezifiziert (mittlerweile von allen aktuellen Browsern unterstützt), die einen einfachen Zugriff (z.B.: JavaScript) auf das Dokument ermöglicht.

8.1 Darstellung

Eine gute Darstellung ist beispielsweise mit dem DOM-Inspector des Browsers Firefox möglich:

HTML - Document	DOM Hierarchie (Element-Knoten)

<pre><html>
  <head><meta charset="utf-8" />
  <title>DOM Tutorial</title>
  </head>
<body><h1 id="Ueb" >DOM Lektion 1</h1>
  <p>einige Knoten-Arten</p>
 	
   - nodeType = 1 Element
   - nodeType = 2 Attribut
   - nodeType = 3 Text
 	
</body>
</html></pre>
	

Knoten werden u. a. unterschieden in Element-, Attribut- und Textknoten die miteinander in Beziehungen (Eltern-Kinderelemente, Vorfahren-Nachkommen oder Geschwisterelemente) stehen.

8.2 Zugriff auf Element-, Attribut- und Textknoten

Einfacher Zugriff (z.b.: mit JavaScript) kann über einfache Methoden realisiert werden:

`document.getElementById('id-wert')`	selektiert einen Element-Knoten (über die ID)
`document.getElementByName('name-wert')`	selektiert einen Element-Knoten (über den Namen)
`document.getElementByClass('class-wert')`	selektiert einen Element-Knoten (über die Klasse)
`document.getElementById('id-wert').attribut = 'wert'`	verändert einen Attributswert
`document.getElementById('id-wert').innerHTML = 'text'`	ändert den Textinhalt

Weiterhin werden auch Methoden zum Einfügen und Löschen von Elementen bereitgestellt, die eine dynamische Änderung des WEB-Dokuments ermöglichen.

9. Sonderzeichen

Der Zeichenvorrat von HTML basiert auf dem Unicode-System. Das bedeutet, dass alle Sonderzeichen (= Zeichen die nicht im ASCII-Zeichensatz vorhanden sind) durch HTML-Entities dargestellt werden müssen.

Symbol	Entities	Symbol	Entities	Symbol	Entities	Symbol	Entities
Ä	Ä	ö	ö	¼	¼	√	√
Ö	Ö	ü	ü	½	½	∞	∞
Ü	Ü	ß	ß			©	©
ä	ä	€	€	¶	¶	®	®