Winter School 2022

Bergische Universität Wuppertal

21.03 – 25.03.2022

Einführung in **HTML & CSS**

Nadine Sutor und Andreas Mertgens

#wsde22

HTML und CSS - Einführung

- HTML-Dokumentenaufbau
- Konzept block vs. inline
- Grundlegende Elemente
- Seitenspezifische Elemente für Aufbau und Struktur einer Digitalen Edition am Beispiel der *Tagebücher von Karl Wiesinger (1961-1973)*
- Physisches, d.h darstellungsorientiertes HTML-Mark-up (Überleitung zu CSS)

Aufbau eines HTML-Dokuments

```
<!DOCTYPE html>
```
Ein HTML-Dokument beginnt mit der Kennzeichnung des **Dokumententyps** (kein Element!)

```
<html>
```
Dem folgt das "Wurzelelement". Damit identifizieren Browser Anfang und Ende eines HTML-Files

```
    <head>

        <title></title>
```
Darin kommt zunächst der Dokumenten-"Kopf". Enthält Informationen, die für das ganze Dokument relevant sind (u.a. den Titel)

```
    </head>

    <body></body>
```
Anschließend folgt der Dokumenten-"Körper". Hier befindet sich der eigentliche Inhalt des Dokuments. Teil, der vom Webbrowser dargestellt wird

```
</html>
```

Hypertext Markup Language

- HTML-Dokumente bilden die Grundlage des *World Wide Webs* und werden von Webbrowsern dargestellt
- HTML *strukturiert* elektronische Dokumente (man kann Texte mit Hyperlinks versehen, man kann Bilder einbetten)
- HTML-Dateien können zusätzliche Angaben in Form von Metadaten enthalten (z.B. über die verwendeten Sprachen, über den Inhalt des Textes, eingebettete Schriften)
- Weiterentwickelt und gepflegt vom *W3C* (World Wide Web Consortium) und der *WHATWG* (Web Hypertext Application Technology Working Group)
- vollständige Spezifikation: https://html.spec.whatwg.org/multipage/

Wichtiges Konzept: block vs. inline
Hat Einfluss auf die Darstellung und den Umgang mit Elementen

"Every HTML element has a default display value, depending on what type of element it is. There are two display values: **block** and **inline**" *(w3schools)*

- Block-Elemente: container für weitere Elemente, beginnen immer in einer neuen Zeile, produzieren in der Ausgabe einen Absatz und bilden die Struktur eines HTML-Dokuments

  `<div>,<p>`

- Inline-Elemente: Beginnen nicht in einer neuen Zeile, sondern stehen mitten im "Text", produzieren im Output keinen Weißraum (whitespace)

  `<span>`

Wichtige Blockelemente

- `<p>` (*paragraph*) = **Absätze**
  Browser fügen i.d.R. vor und nach einem Absatz eine Leerzeile ein. Will man diese vermeiden, verwendet man stattdessen
- `<div>` = (*division*) gruppierendes Element (**container**)
- `<h1>` bis `<h6>` (*heading*) = **Überschriften**
- `<address>` = führt zu einer Blockdarstellung für **Adressen**
- `<blockquote>` = hebt **Zitate** in einem eigenen, anders formatierten (meist eingerückten) Absatz hervor
- `<pre>` = leitet einen Textabschnitt mit **präformatiertem** Text ein.
  Alles, was zwischen `<pre></pre>` steht wird so angezeigt, wie es eingegeben wurde (Leerzeichen und Zeilenumbrüche werden übernommen)

Zutaten:

- 150g Bulgur
- 300 ml.
- Gemüsebrühe
- 1 Paprikaschote

Blockelemente: **Listen**

geordnet und nummeriert. Eine nummerierte Liste wird durch die Tags

`<ol>` und **`<li>`** erzeugt

vs. ungeordnet. Eine ungeordnete Liste wird durch die Tags

`<ul>` und **`<li>`** erzeugt

In beiden Listentypen muss mindestens ein Listenelement vorhanden sein

```
<h3>Zutaten:</h3>
    <ul>
      <li>150g Bulgur</li>
      <li>300 ml.</li>
      <li>Gemüsebrühe</li>
      <li>1 Paprikaschote</li>
     ...
</ul>
```

Blockelemente: **Tabellen**

Nährwerte pro Portion

kcal	Eiweiß	Fett	Kohlenhydrate
477	19,65 g	15,58 g	63,16 g

Werden durch die Tags

<table>	Anfang der Tabelle
<caption>	Überschrift der Tabelle
<th>	Kopf der Tabelle
<tr>	Zeile der Tabelle
<td>	Eintrag/Zelle der Tabelle

erzeugt

Geht noch komplexer!

```
<table>
<caption>Nährwerte pro Portion</caption>
     <tr>
         <th>kcal</th>
         <th>Eiweiß</th>
         <th>Fett</th>
<th>Kohlenhydrate</th>
      </tr>
      <tr>
          <td>477</td>
          <td>19,65 g</td>
          <td>15,58 g</td>
          <td>63,16 g</td>
        </tr>
  </table>
```

Inline-Elemente: Bisher befindet sich in den Blockelementen nur Text. Zur weiteren Auszeichnung kommen nun inline-Elemente hinzu.

- `<strong>` erzeugt eine **fette** Schrift
- **`<span>`** **gruppierendes Element**, ordnet inline-Elementen Eigenschaften zu. Hat selbst keine Formateigenschaft.
- `<br/>` erzeugt einen **Zeilenumbruch** (eigene Syntax: hat keinen eingebetteten Inhalt. Ist ein leeres Element)

“logische” Auszeichnungselemente

- `<em>` betont die Textausgabe (i.d.R. mit einer *Kursivierung*)
- `<code>` erzeugt eine `Teletypeschrift`
- `<samp>` “dies ist ein Beispiel”
- `<kbd>` “dies ist eine Tastatureingabe”

Weitere “logische” Auszeichnungselemente

- `<var>` “dies ist eine **Variable**”
- `<cite>` “dies ist ein **Querverweis** zu einer anderen Quelle”
- `<dfn>` “dies ist eine **Begriffsdefinition**”
- `<abbr>` “dies ist eine **Abkürzung**”. Erzeugt in Kombination mit dem Attribut *@title* in der Ausgabe ein Pop-up

  (`<abbr title=”Pop-up-Text”>...</abbr>`)
- `<q>` “dies ist ein **Zitat**”. Setzt den Text in „Anführungszeichen“ und *kursiv.* (vs. blockquote! = Block-Element)

Einbau von Hyperlinks

Verweise sind ein entscheidender Bestandteil jedes Hypertextprojekts und ein wesentlicher Gewinn des WWW. Für einen Verweis wird das sog. Ankerelement **<a>** mit einem **href**-Attribut verwendet.

<a> ist ein Inline-Element. Der Wert des Attributs ist eine URI:

```
<a href="URL">Verweistext</a>
```

Das Attribut `target= _blank | blank | _self`

1) öffnet jedes mal ein neues Fenster 2) öffnet einmalig ein neues Fenster, Tab aktualisiert sich, oder 3) öffnet den Link im gleichen Fenster

Einbau von Bildern

Um Bilder und Grafiken in HTML-Dokumente einzubetten, referenzieren Sie die Dateien an den gewünschten Stellen im HTML-Dokument:

```
<img src=″Hier steht ein Link”/>   (Ressource aus dem WWW)
<img src=″../../datei.jpg”/>       (ein relativer Pfad in Ihrem Dateisystem)
```

Hinter dem “=” steht der Pfad zur Grafikdatei, auf die verwiesen wird.

Der obige Dateiname als Wert des Attributes `src` verweist auf dieselbe Ressource wie der Link. Es ist üblich Bilddateien in einer eigenen Directory zu sammeln, z.B. *images* oder *resources*.

Einbau von Bildern

Für eine eingebundene Grafik sollten **Breite und Höhe** immer mit angegeben werden:

```
<img src="datei.jpg"
width="142" height="148"/>
```

`width="100%"` stellt das Bild in seiner Originalgröße dar

Weiteres Attribut:

`alt` = Alternativtext (Screenreader)

Übliche Dateiformate für WWW-gerechte Grafiken:

- **GIF** (Graphics Interchange Format)
- **PNG** (Portable Network Graphics) Gutes Einzelbildformat, Logos und Icons
- **JPEG** (Joint Photographic Experts Group) für Fotos (verlustbehaftet)
- **SVG** (Scalable Vector Graphics) Vektorgrafikformat, XML-basiert, Logos, Icons

Seitenspezifische Elemente für Aufbau und Struktur einer Website am Beispiel der *Tagebücher von Karl Wiesinger (1961-1973)*

Digitale Edition - ZIM Graz

Karl Wiesinger

Digitale Edition der Tagebücher (1961–1973)

StifterHaus
ADALBERT-STIFTER-INSTITUT DES LANDES OBERÖSTERREICH

UNI GRAZ

Home | About | Tagebücher | Kommentar | Register | Dokumentation | Literatur

Karl Wiesinger (1923–1991): Eine Skizze zu Leben und Werk

Kindheit und Jugend

Karl Wiesinger wird am 13.3.1923 als Sohn eines Dentisten und einer Hausfrau in Linz geboren. Sein Aufwachsen beschreibt er als behütet, wiewohl die Beziehung zwischen seinen Eltern konfliktreich gewesen sein dürfte.

Die wachsenden politischen Spannungen der Zwischenkriegszeit erlebt Wiesinger im Zuge der Februarkämpfe, die am Morgen des 12. Februar 1934 in Linz ihren Ausgang nehmen und an deren Ende mehrere hundert Tote zu beklagen sind. Wiesinger beschreibt in einem Interview zum Erscheinen des Romans *Der rosarote Straßenterror* im Jahr 1974, wie er die Ereignisse als Kind wahrnahm: „Die Schüsse damals und die Belagerung des Zentralkinos durch das Bundesheer und die Pferdekadaver nachher auf den Straßen und die Munitionshülsen, die herumgelegen sind, einige Tote, die ich gesehen hab als Kind, das hat mich stark beeindruckt (...)“.[1]

Kinderfoto: Sylvia Gresak, Privatbesitz

Erster Eindruck:

- optisch schlicht, „alles in einem Guss“
- Integration von Bild und Text
- schönes, farbiges Hervorheben von Themen im Text

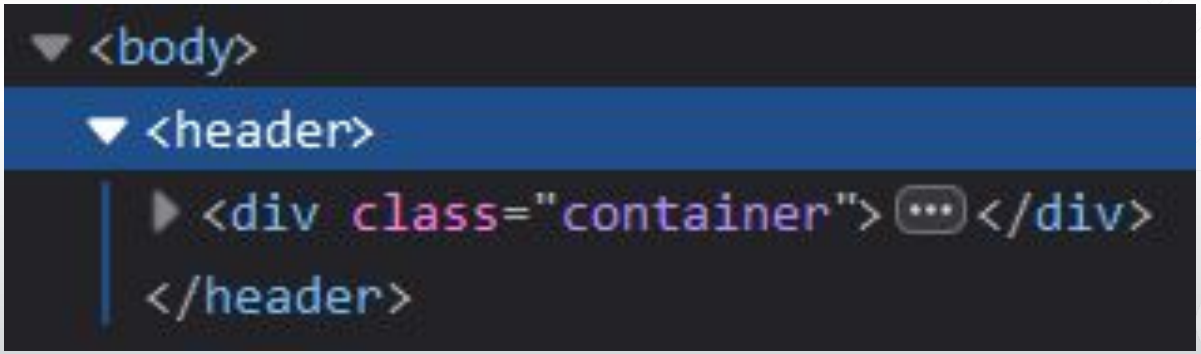
Welche Elemente wurden hier verwendet?

- header
- nav
- main
- section
- article
- div
- img
- a
- footer
- …

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

`header` (nicht zu verwechseln mit `<head>` !)

- repräsentiert eine Gruppe von Einführungs- oder Navigationshilfen
- auch andere Elemente wie zum Beispiel ein Logo oder ein Suchfeld
- umfasst i.d.R. `div's` mit einer vergebenen Klasse für weitere (Text)-Container



`<nav>` - (navigation)

- zeichnet einen Seitenbereich aus, der dazu bestimmt ist Navigationslinks bereitzustellen, die entweder zum aktuellen Dokument oder zu anderen Dokumenten zeigen
- häufige Beispiele für Navigationsbereiche sind (*drop-down)*-Menüs, sog. *sidebars* oder Inhaltsverzeichnisse

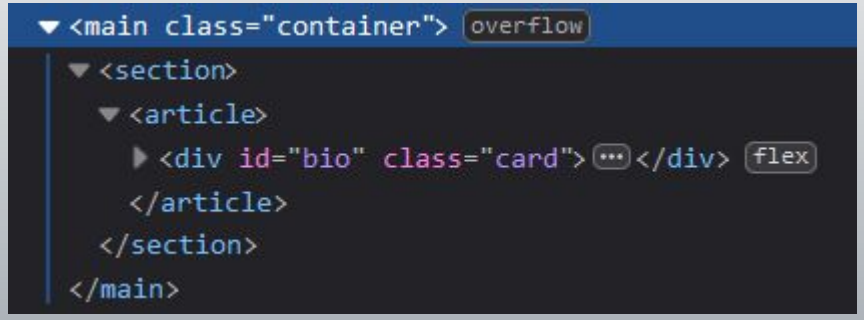
```
<nav class="menu">
  <ul>
    <li><a href="#">Startseite</a></li>
    <li><a href="#">Über</a></li>
    <li><a href="#">Kontakt</a></li>
  </ul>
</nav>
```

In diesem Beispiel wird das `<nav>` Element zusammen mit einer ungeordneten Liste (`<ul>`) verwendet, die wiederum Links enthalten. Mit CSS kann diese Liste als Seitenleiste, als oben mittig zentrierte Navigationsleiste oder Drop-Down-Menü verwendet werden.
https://developer.mozilla.org/de/docs/Web/HTML/Element/nav

`main`

- dient als Container für den HAUPT-Inhalt einer Website
- Hauptteil besteht aus Inhalt, der auf das zentrale Thema oder der zentralen Funktionalität einer Applikation aufbaut
- Inhalt sollte “einzigartig” sein. Ausgenommen von Inhalten, der in dem Projekt wiederholt dargestellt wird, wie z.B. Seitenleisten, Navigationlinks, Logos, und Sucheingaben

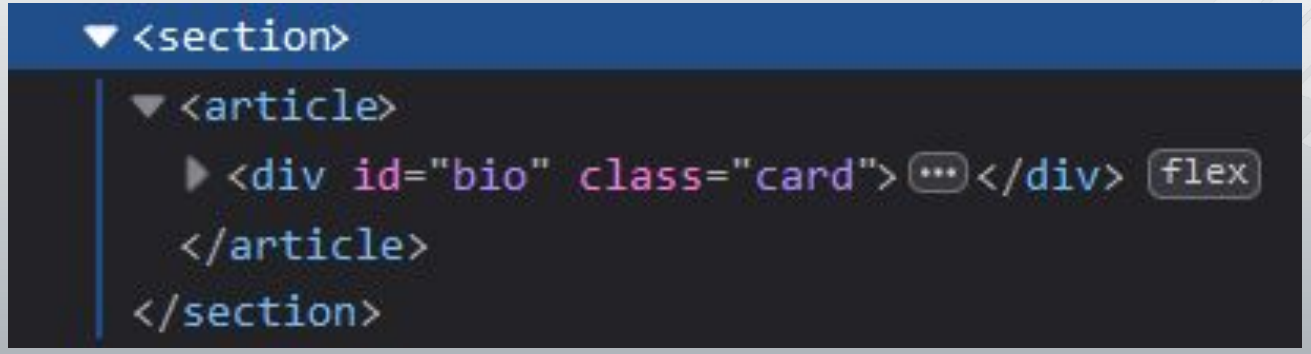
- `<main>` **darf kein** Nachfolger eines `<article>, <aside>, <footer>, <header>` oder `<nav>` Elements sein!
- **Nur ein** `<main>` Element kann pro Dokument verwendet werden



`section` – Abschnitte

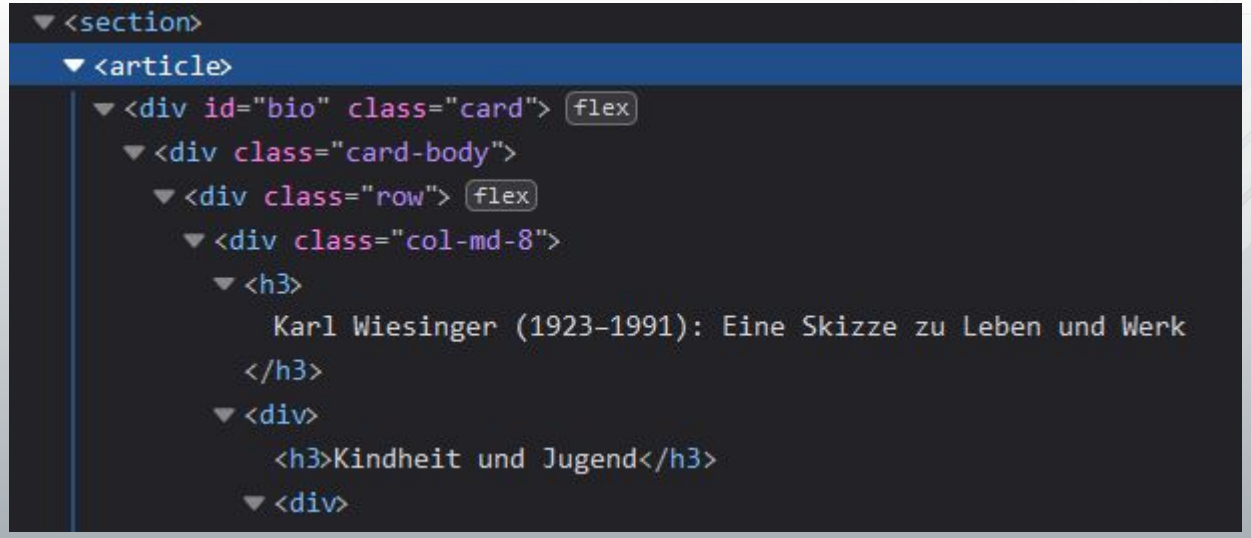
- stellt eine allgemeine Unterteilung des Dokumentes dar (z.B. eine thematische Gruppierung des Inhaltes, typischerweise mit einer Überschrift)
- jeder `<section>` Abschnitt sollte identifizierbar sein, üblicherweise durch Einfügen einer Überschrift (`h1-h6`) als Kindelement vom `<section>` Element oder durch Vergabe einer Klasse
- Wenn der Inhalt des `<section>` Elementes “sinnvoll” vom übrigen Inhalt “getrennt” werden soll, nutzt man stattdessen das `<article>` Element (nächste Folie)

- `<section>` soll nicht als allg. Containerelement benutzt werden! → `<div>`



`article`

- stellt eine in sich geschlossene Komposition in einem Dokument, einer Seite, Anwendung oder Website dar, die unabhängig verteilt oder wiederverwendbar sein soll
- z. B. ein Forumsbeitrag, ein Zeitschriften- oder Zeitungsartikel oder ein Blogeintrag
- ein Dokument kann mehrere Artikel enthalten, z. B. in einem Blog, in dem der Text jedes Artikels nacheinander angezeigt wird. Während die Leser*innen blättern, wäre jeder Beitrag in einem `<article>`-Element kodiert, welche wiederum innerhalb von `<section>` stehen



`footer`

- repräsentiert eine Fußzeile für den Sektionsinhalt, bzw. `<article>`, `<aside>` und `<nav>`
- Eine Fußzeile enthält typischerweise Informationen über die Institutionen, die Copyright-Daten, das Impressum, die Lizenzen etc.

```
<footer class="footer">
  <div class="container"> overflow
    <div class="row"> flex
      <div class="col-md-6">
        <h6>Weitere Informationen</h6>
        <p> ••• </p>
        <div class="icons">
          <a href="/" target="_blank"> ••• </a>
          <a href="https://informationsmodellierung.uni-graz.at/" target="_blank"> ••• </a>
          <a href="http://creativecommons.org/licenses/by-nc/4.0/" target="_blank"> ••• </a>
        </div>
```

F R A G E N ?

Die Frage nach dem guten (Präsentations-)Stil !

HTML ist zunächst eine Sprache für die **visuelle Darstellung** für uns als menschliche “Normalbeobachter”. Das sieht man besonders deutlich im “**physischen**” (Darstellungs-) Markup.

- `<tt>` `teletype` = nicht-proportionale Schrift
  (The “tt” element is obsolete. Use CSS instead.)
- `<i>` *italic* = kursiv
- `<b>` **bold** = fett

Physisches, d.h. darstellungsorientiertes Markup.

- `<sup>` hochgestellt
- `<sub>` tiefgestellt
- `<big>` groß
  (The “big” element is obsolete. Use CSS instead.)
- `<small>` klein
- `<hr/>` horizontal rule (leeres Blockelement)

Sollte man **so nicht verwenden**!
Stilanweisungen stattdessen mittels **CSS** vergeben = Guter Stil!

Common Attributes

`id` = Unique element identifier (Eindeutige ID für Elemente)
`class` = class(es) of an element (vergibt Klassen an Elemente)
`style` = Stylesheet-Regeln (für “inline”-Stile)

Wichtig für CSS!

`<div>` = Mit zusätzlichen Attributen werden div’s Eigenschaften zugeordnet. `div` hat häufig das `id` oder `class`-Attribut.
`<span>` = Hauptaufgabe von `span` ist es inline-Textabschnitten Eigenschaften zuzuordnen. `span` hat häufig das `style`-Attribut.

F R A G E N ?

Überleitung zu CSS

Vielen Dank für Ihre Aufmerksamkeit!

BERGISCHE
UNIVERSITÄT
WUPPERTAL

BERGISCHE
UNIVERSITÄT
WUPPERTAL