# CSS

## Was ist CSS?

CSS (Cascading Style Sheets) ist eine weitere Sprache, die wir beim Erstellen von Webseiten verwenden.

HTML: Organisation der Inhalte; mit CSS passen wir das **Aussehen** an.

Wie HTML ist CSS keine Programmiersprache. Es ist eine Gestaltungssprache, deren Syntax beschreibt, wie bestimmte Attribute von HTML-Elementen verändert werden sollen

## CSS-Regeln

```
body
{
    background-color: blue;
}
```

## Syntax von CSS-Regeln

Eine CSS-Regel wird erstellt, indem man einen **Selektor** definiert (im vorigen Beispiel body) und dann eine geschweifte Klammer öffnet, um den Beginn der Deklarationen für diesen Selektor anzuzeigen.

Zwischen den geschweiften Klammern platziert man eine Liste von Schlüssel-Wert-Paaren der Stileigenschaften und ihrer Werte, wobei jede Deklaration mit einem Semikolon endet.

Eine schließende geschweifte Klammer beendet die CSS-Regel.

```
body
{
    background-color: blue;
}
```

## Häufig verwendete CSS-Eigenschaften

`border: style color width`

Fügt einen Rahmen mit definierter Farbe, Breite und Stil hinzu (z.B. dotted, dashed, solid, ridge...).

`background-color: [keyword | #Hexcode]`

Legt die Hintergrundfarbe fest. Einige Farben sind in CSS vordefiniert (z.B, black, white, red, usw.).

`color: [keyword | #Hexcode]`

Bestimmt die Vordergrundfarbe (meist für Text).

```
body
{
    background-color: blue;
}
```

## Weitere häufige CSS-Eigenschaften

`font-size: [absolute Größe | relative Größe]`

oft feste Pixelgrößen (10px, 12px, ...)
oder Prozentangaben (80%, 120%)

`font-family: [Schriftname | generischer Name]`

Bestimmte „websichere“ Schriftarten
sind in CSS vordefiniert, z.B. `sans-serif`.

`text-align: [left | right | center | justify]`

Für die Textausrichtung

## CSS-Selektoren: Einführung

CSS-Selektoren müssen sich nicht nur auf HTML-Tag-Kategorien beziehen.

Es gibt auch ID-Selektoren und Klassen-Selektoren.

Ein **Tag-Selektor** gilt für alle Elemente mit diesem HTML-Tag.

Beispiel:

```
h2
{
    font-family: times;
    color: #fefefe;
}
```

## ID-Selektoren

Ein ID-Selektor gilt nur für das HTML-Element mit dieser eindeutigen ID.

HTML:

```
<p id="unique">
  Dieser Absatz hat eine eindeutige ID.
</p>
```

CSS:

```
#unique
{
    border: 4px dotted blue;
    text-align: right;
}
```

## Klassen-Selektoren

Ein Klassen-Selektor gilt für alle
HTML-Elemente mit dieser Klasse.

HTML:

```
<p class="students">Erster Absatz.</p>
<div class="students">Zweites Element.</div>
<p class="students">Dritter Absatz.</p>
```

CSS:

```
.students
{
    opacity: 0.7;
    background-color: yellow;
}
```

## Cascading: Mehrere Regeln kombinieren

Beide Regeln werden angewendet. Die Schriftgröße kommt aus der ersten Regel, die Farbe aus der zweiten (Klassenselektoren sind *spezifischer* als Tag-Selektoren)

```
<p class="special">Dieser Text ist rot und 16px groß!</p>
```

```
p {
    color: blue;
    font-size: 16px;
}

.special {
    color: red;
}
```

**Das Box-Modell**

Jedes HTML-Element ist als Box aufgebaut:

- *content:* der eigentliche Inhalt des Elements
- *padding:* der innere Abstand zum Rahmen
- *border:* der Rahmen des Elements
- *margin:* der äußere Abstand zu anderen Elementen

Das Verständnis dieses Modells hilft beim Layouten

## Praxisbeispiel (HTML)

```
<article class="preview">
    <h2 id="article-title">CSS in der Praxis</h2>
    <p class="meta">Geschrieben von: Max</p>
    <div class="content">
        Ein Beispiel für die Verwendung von CSS-Selektoren
        und dem Box-Modell.
    </div>
</article>
```

## Praxisbeispiel (CSS)

```css
.preview {
    background-color: #f5f5f5;
    margin: 20px auto;
    padding: 15px;
    border: 1px solid #ddd;
    width: 80%;
}

#article-title {
    color: #333;
    margin-bottom: 10px;
}

.meta {
    color: #666;
    font-size: smaller;
}

.content {
    padding: 10px;
    background-color: white;
}
```

**Implementierung von Stylesheets**

Stylesheets können **direkt im HTML** eingebettet werden

- Platzierung innerhalb von `<style>` Tags im head-Bereich
- Geeignet für kleine, einfache Seiten

Alternative: **separate CSS-Dateien** (empfohlen)

- Einbindung über `<link>` Tags im head-Bereich
- Trennung von Inhalt (HTML) und Gestaltung (CSS)
- Ermöglicht Browser-Caching für bessere Performance

Debugging von CSS

- Browser Developer Tools (F12 oder Rechtsklick → Inspect)
- Direkte Überprüfung der angewandten Stile

## CSS einbetten vs. externe Datei

Einbettung im HTML:

```
<head>
    <style>
        body { background-color: white; }
    </style>
</head>
```

Externe CSS-Datei (empfohlen):

```
<head>
    <link rel="stylesheet" href="styles.css">
</head>
```

## Beispiel: Multiplikationstabelle

```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="UTF-8">
    <title>Multiplikationstabelle</title>
</head>
<body>
    <table>
        <tr> <td>1</td><td>2</td><td>3</td> </tr>
        <tr> <td>2</td><td>4</td><td>6</td> </tr>
        <tr> <td>3</td><td>6</td><td>9</td> </tr>
    </table>
</body>
</html>
```

## Beispiel: Multiplikationstabelle

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>Multiplikationstabelle</title>
  <style>
    table {
      border: 1px solid blue;
    }
    td {
      border: 1px solid gray;
      padding: 5px;
    }
    </style>
</head>
<body>
   <table>
   <tr>
     <td>1</td><td>2</td><td>3</td>
   </tr>
   <tr>
     <td>2</td><td>4</td><td>6</td>
   </tr>
   <tr>
     <td>3</td><td>6</td><td>9</td>
    </tr>
    </table>
</body>
</html>
```

## Erste Verbesserung

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>Multiplikationstabelle</title>
  <style>
    table {
      border: 1px solid blue;
    }
    td {
      border: 1px solid gray;
      padding: 5px;
    }
    </style>
</head>
<body>
   <table>
   <tr>
     <td>1</td><td>2</td><td>3</td>
   </tr>
   <tr>
     <td>2</td><td>4</td><td>6</td>
   </tr>
   <tr>
     <td>3</td><td>6</td><td>9</td>
    </tr>
    </table>
</body>
</html>
```

## Finale Version (HTML)

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>Multiplikationstabelle</title>
  <link rel="stylesheet" href="table.css">
</head>
<body>
  <table>
    <tr><td>1</td><td>2</td><td>3</td></tr>
    <tr><td>2</td><td>4</td><td>6</td></tr>
    <tr><td>3</td><td>6</td><td>9</td></tr>
    </table>
</body>
</html>
```

## Finale Version (CSS)

```css
table {
    border: 5px solid #cc0000;
    margin: 20px;
}

tr {
    height: 50px;
}

td {
    background-color: black;
    color: white;
    font-family: Georgia;
    font-size: 22px;
    text-align: center;
    width: 50px;
}
```

**Block-Elemente**

- Beginnen in einer neuen Zeile, nehmen die volle verfügbare Breite ein
- Beispiele: `<div>`, `<p>`, `<h1>`, `<ul>`

**Inline-Elemente**

- Fließen im Text
- Nehmen nur die benötigte Breite ein
- Beispiele: `<span>`, `<strong>`, `<a>`

Änderung des Verhaltens mit der *display*-Eigenschaft:

- display: inline → Block-Elemente fließen wie Text
- display: block → Inline-Elemente werden zu Containern
- display: inline-block → Kombination beider Verhaltensweisen

## Praktisches Beispiel: Horizontales Menü

```html
<ul class="nav">
    <li>Start</li>
    <li>Produkte</li>
    <li>Kontakt</li>
</ul>
```

```css
.nav {
    list-style: none; /* entfernt Aufzählungszeichen */
}
.nav li {
    display: inline;  /* Listenpunkte horizontal */
    padding: 0 20px;  /* Abstand zwischen Einträgen */
}
```

## Ausblick

### Responsive Design

- Webseiten für alle Bildschirmgrößen optimieren
- Verwendet Eigenschaften wie *max-width*
- Nutzt Media Queries für verschiedene Geräte

CSS-Frameworks

- Fertige Werkzeuge wie **Bootstrap** und **Tailwind**
- Erleichtern die Webentwicklung

# EXTRAS IN 3 MINUTEN

FRAGEN – ANTWORTEN – RÄTSEL
UND KURZE ZUSAMMENFASSUNG

## Profilkarte in modernem HTML/CSS

```
<div class="profile-card">
    <img id="profile-img" src="/api/placeholder/120/120" alt="Profilbild">
    <h1>Jane Doe</h1>
    <p class="bio">Informatik-Studentin an der Uni Bamberg.
       Begeistert von Webentwicklung und KI.</p>

    <ul class="details">
        <li>🎓 BSc Informatik</li>
        <li>📍 Bamberg</li>
        <li>💻 Entwicklerin</li>
    </ul>

    <div class="links">
        <a href="#">Portfolio</a>
        <a href="#">GitHub</a>
        <a href="#">LinkedIn</a>
    </div>
</div>
```

## Profilkarte: Grundlegendes CSS

```css
/* Grundlegende Kartenstruktur */
.profile-card {
    background: white;
    border-radius: 8px; /* Neu: Abgerundete Ecken */
    padding: 20px;
    width: 300px;
    text-align: center;
}

/* Bildstil */
#profile-img {
    width: 120px;
    height: 120px;
    border-radius: 50%; /* Neu: Macht das Bild rund */
    margin-bottom: 15px;
}
```

```css
h1 {
    color: #1a1a1a;
    font-size: 24px;
    margin-bottom: 10px;
}
```

## Ausblick: Code für Reset

Diese Zeilen entfernen die in jedem Browser eingebauten Standardabstände.

Das nennt man **Reset**.

```
* {
    box-sizing: border-box;
    margin: 0;
}

img {
    max-width: 100%;
    display: block;
}

button, input, textarea, select {
    font: inherit;
}
```

## Ausblick: Code für Container mit Flexbox

Mit diesen wenigen Zeilen ordnen sich die Karten automatisch nebeneinander an

Wenn der Bildschirm zu schmal wird, brechen die Karten automatisch um

Das nennt man **responsive Design**.

```
.card-container {
    display: flex;
    flex-wrap: wrap;
    gap: 20px;
    justify-content: center;
    padding: 20px;
}
```

## Ausblick: Grid-Layouts

Erzeugt ein noch präziseres Raster

Die Karten haben alle die gleiche Breite

Moderne CSS-Technik für komplexe Layouts

```
.grid-container {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
    gap: 20px;
    padding: 20px;
}
```

## Sie wollen tiefer einsteigen?

- Lernen durch Anpassen der Beispiele:
- Farben der Karte ändern
- Abstände anpassen
- Details hinzufügen
- Border-Radius-Werte testen
- Schatten-Effekte variieren
- Tiefer einsteigen mit folgenden Themen:
- Flexbox und Grid
- Responsive Design
- CSS-Animationen

CSS ist eine Gestaltungssprache (keine Programmiersprache) und beschreibt das Aussehen von Webseiten

Das Box-Modell: content, padding, border, margin

CSS-Regeln bestehen aus Selektor und Deklarationsblock mit Eigenschaft-Wert-Paaren

Block- vs. Inline-Elemente bestimmen das Layout-Verhalten

Drei Selektortypen: Tag-Selektoren, ID-Selektoren (#), Klassen-Selektoren (.)

"Cascading": Mehrere Regeln können kombiniert werden, spezifischere gewinnen