

Modul 293 - Aufgabenblatt 08

von Lukas Meier

Unterricht vom 10.02.2026

Aufgabenblatt: Repetition HTML und CSS

Bearbeiten Sie die Aufgaben in der vorgegebenen Reihenfolge. Die Aufgaben werden pro Thema schrittweise anspruchsvoller.

Thema 1: HTML Grundlagen

Aufgabe 1.1 – Begriffe sicher unterscheiden

Erklären Sie in eigenen Worten:

- Was ist ein HTML-Tag?
- Was ist ein Attribut?
- Was ist der Unterschied zwischen Start-Tag, End-Tag und Inhalt?

Geben Sie pro Begriff ein kurzes Beispiel an.

Aufgabe 1.2 – Tags erkennen und einordnen

Analysieren Sie den folgenden Code:

```
<a href="https://example.com" target="_blank">Zur Website</a>  
  
<section class="intro">Willkommen</section>
```

Bearbeiten Sie dazu:

- Markieren Sie alle Tags.
- Markieren Sie alle Attribute.

- Notieren Sie, welches Element selbstschliessend ist.

Aufgabe 1.3 – Selbstschliessende Tags

Erstellen Sie eine kurze HTML-Struktur (ohne CSS und nur den body) mit:

- einer Hauptüberschrift
- einem Absatz
- einem Zeilenumbruch
- einem Bild mit sinnvollem alt-Attribut
- einer horizontalen Linie

Verwenden Sie dabei mindestens drei selbstschliessende Tags.

Aufgabe 1.4 – Semantik verbessern

Gegeben ist folgender HTML-Ausschnitt:

```
<div>
  <div>Meine Lernseite</div>
  <div>
    <div>Start</div>
    <div>Kontakt</div>
  </div>
  <div>
    <div>Heute üben wir HTML.</div>
  </div>
</div>
```

Ersetzen Sie die unspezifischen div-Elemente durch semantisch passende Tags (z. B. header, nav, main, section, footer).

Aufgabe 1.5 – Fehleranalyse

Im folgenden Code stecken mehrere konzeptionelle Fehler:

```
<section>
  <h1 class="titel">Wochenplan
  <p id="text" "wichtig">Montag: CSS wiederholen</p>
  <img source="plan.png">
</section>
```

Bearbeiten Sie:

- Finden Sie die vier Fehler.
- Korrigieren Sie den Code vollständig.
- Begründen Sie jede Korrektur in einem Satz.

Aufgabe 1.6 – Transferaufgabe HTML

Erstellen Sie eine vollständige Mini-Seite mit:

- `header` mit Titel
- `nav` mit mindestens drei Links
- `main` mit zwei `section`-Bereichen
- mindestens einem selbstschliessenden Tag
- `footer` mit Copyright-Text

Achten Sie auf korrekte Verschachtelung und sinnvolle Semantik.

Thema 2: CSS Selektoren

Aufgabe 2.1 – Grundprinzipien

Beantworten Sie schriftlich:

- Wie spricht man ein Element in CSS nur über den Tag-Namen an?
- Wie spricht man eine Klasse an?
- Wie spricht man eine ID an?

Geben Sie je ein kurzes Beispiel (HTML + CSS) an.

Aufgabe 2.2 – Selektoren zuordnen

Gegeben ist dieser HTML-Code:

```
<h1 id="title">Modul 293</h1>
<p class="info">Einführung</p>
<p class="info">Repetition</p>
<button class="info" id="start-btn">Start</button>
```

Notieren Sie für die folgenden Selektoren jeweils, welche Elemente getroffen werden:

- `p`
- `.info`
- `#title`
- `button.info`

Aufgabe 2.3 – Mehrere Elemente in einem Selektor

Erstellen Sie CSS-Regeln, sodass:

- `h1`, `h2` und `h3` dieselbe Textfarbe erhalten
- alle `p`- und `li`-Elemente dieselbe Schriftgrösse erhalten
- die Klasse `note` und die ID `warning` dieselbe Hintergrundfarbe erhalten

Nutzen Sie dafür jeweils gruppierte Selektoren mit Komma.

Aufgabe 2.4 – Kombinierte Selektoren

Ergänzen Sie passende CSS-Regeln für folgende Anforderungen:

- Nur `p`-Elemente mit Klasse `info` sollen blau sein.
- Nur das Element mit ID `menu` soll einen Rand erhalten.
- Nur `li`-Elemente innerhalb einer Klasse `nav` sollen fett sein.

Erklären Sie kurz, warum ein allgemeiner Selektor wie `.info` hier zu ungenau wäre.

Aufgabe 2.5 – Fehler finden und korrigieren

Korrigieren Sie den CSS-Code:

```
title {  
  color: red;  
}  
  
#.intro {  
  font-size: 20px;  
}
```

```
p, .hinweis #wichtig {  
  background: yellow;  
}
```

Bearbeiten Sie:

- Korrigieren Sie alle Syntax- und Logikfehler.
- Erklären Sie pro Zeile kurz, was falsch war.

Aufgabe 2.6 – Transferaufgabe Selektoren

Erstellen Sie einen HTML-Ausschnitt mit mindestens:

- 2 Überschriften
- 3 Absätzen
- 1 Liste mit 4 Einträgen
- 1 Button

Erstellen Sie danach CSS mit:

- mindestens einem Tag-Selektor
- mindestens zwei Klassen-Selektoren
- mindestens einem ID-Selektor
- mindestens zwei gruppierten Selektoren
- mindestens einem kombinierten Selektor (z. B. `p.info`)

Thema 3: CSS Box-Modell

Aufgabe 3.1 – Theorie verstehen

Beschreiben Sie den Aufbau des Box-Modells in der korrekten Reihenfolge von innen nach aussen.

Erklären Sie anschliessend in eigenen Worten:

- Was ist `padding`?
- Was ist `margin`?
- Welche CSS-Properties werden für diese verwendet?

Aufgabe 3.2 – Box aufbauen

Erstellen Sie eine Klasse `.card` mit:

- `width: 280px`
- `padding: 16px`
- `border: 2px solid black`
- `margin: 20px`

Skizzieren Sie zusätzlich, welcher Bereich Inhalt, Padding, Border und Margin ist.

Aufgabe 3.3 – Margin vs. Padding gezielt testen

Erstellen Sie zwei gleich grosse Boxen nebeneinander oder untereinander.

Bearbeiten Sie schrittweise:

- Erhöhen Sie nur das `padding` und beobachten Sie den Effekt.
- Setzen Sie das `padding` zurück und erhöhen Sie nur die `margin`.
- Beschreiben Sie den Unterschied in mindestens drei Sätzen.

Aufgabe 3.4 – Block und Inline vergleichen

Untersuchen Sie je ein `div`- und ein `span`-Element.

Bearbeiten Sie:

- Welche Standard-`display`-Werte haben die beiden Elemente?
- Welche Eigenschaften (`width`, `height`, vertikale `margin`) wirken bei `span` standardmässig nicht wie erwartet?
- Ändern Sie das `span` auf `display: inline-block`; und vergleichen Sie erneut.

Aufgabe 3.5 – Rechnen mit dem Box-Modell

Gegeben:

- `width: 300px`
- `padding-left: 20px`

- `padding-right: 20px`
- `border-left: 3px`
- `border-right: 3px`

Berechnen Sie die gesamte sichtbare Breite:

- einmal mit `box-sizing: content-box`
- einmal mit `box-sizing: border-box`

Erklären Sie den Unterschied.

Aufgabe 3.6 – Transferaufgabe Box-Modell

Erstellen Sie ein kleines Layout mit drei Karten (`article`-Elemente):

- Jede Karte enthält Titel und Text.
- Jede Karte hat sichtbare Unterschiede bei Margin und Padding.
- Mindestens eine Karte verwendet `display: inline-block`.
- Für alle Elemente soll `box-sizing: border-box`; gesetzt sein.

Dokumentieren Sie kurz, warum Ihr Layout trotz Abständen übersichtlich bleibt.

Thema 4: CSS Positionierung

Aufgabe 4.1 – Positionierungsarten beschreiben

Erklären Sie die Unterschiede zwischen:

- `position: static`
- `position: relative`
- `position: absolute`
- `position: fixed`

Ergänzen Sie pro Positionierungsart ein typisches Einsatzbeispiel.

Aufgabe 4.2 – Relative Positionierung anwenden

Erstellen Sie einen Container mit einer Box.

Setzen Sie die Box auf `position: relative;` und verschieben Sie sie mit:

- `top`
- `left`

Beschreiben Sie:

- Von welchem Ursprungsort verschoben wird.
- Ob der ursprüngliche Platz im Dokumentfluss erhalten bleibt.

Aufgabe 4.3 – Absolute Positionierung

Erstellen Sie folgenden Aufbau:

- Ein Container mit `position: relative;`
- Zwei Kind-Elemente mit `position: absolute;`

Positionieren Sie die Kinder an unterschiedliche Ecken.

Entfernen Sie danach testweise `position: relative;` beim Container und beschreiben Sie den Unterschied.

Aufgabe 4.4 – Fixed und Scrollverhalten

Erstellen Sie eine Seite mit genügend Inhalt zum Scrollen.

Fügen Sie ein fixes Element hinzu (z.B. Hinweisbox oder Top-Bar) mit `position: fixed;` und passenden Offset-Werten.

Analysieren Sie:

- Was passiert beim Scrollen?
- Welchen Einfluss hat das feste Element auf die Nutzbarkeit der Seite?

Aufgabe 4.5 – Display und Dokumentfluss

Erstellen Sie drei Boxen mit unterschiedlichem Inhalt untereinander.

Testen Sie nacheinander bei allen Boxen gleichzeitig:

- `display: block`
- `display: inline`
- `display: inline-block`

Passen Sie nun das CSS einzelner Boxen an:

- Kombination mit `position: relative` und `position: absolute`

Beschreiben Sie für jeden Fall den Einfluss auf den Dokumentfluss.

Aufgabe 4.6 – Transferaufgabe Positionierung

Erstellen Sie eine Mini-Webseite mit:

- fixer Kopfzeile
- Hauptbereich mit relativ positioniertem Container
- zwei absolut positionierten Markierungen innerhalb dieses Containers
- mindestens einem Element, das normal im Dokumentfluss bleibt

Ergänzen Sie eine kurze Reflexion (5–6 Sätze):

- Wann ist absolute Positionierung sinnvoll?
- Wann sollte man sie vermeiden?
- Welche Rolle spielt `display` im Zusammenspiel mit `position`?

Abschlussaufgabe (optional)

Erstellen Sie eine kompakte Lernseite, in der alle vier Themen sichtbar vorkommen:

- semantische HTML-Struktur
- verschiedene CSS-Selektoren
- bewusst eingesetztes Box-Modell
- mindestens zwei Positionierungsarten

Ziel: Ein sauberes, gut lesbares Mini-Projekt, das die gesamte Repetition zusammenfasst.