LU01a - Tailwind Grundlagen

Tailwind CSS ist ein modernes und hochflexibles CSS-Framework, das es Entwicklern ermöglicht, schnell und effizient ansprechende Benutzeroberflächen zu gestalten. Im Gegensatz zu traditionellen Frameworks wie Bootstrap, die vorgefertigte Komponenten und Designs bereitstellen, ist Tailwind ein Utility-First Framework. Das bedeutet, dass es sich auf kleine, atomare Klassen konzentriert, mit denen spezifische Styling-Aufgaben direkt im HTML gelöst werden können. Diese Klassen können beliebig kombiniert werden, um vollständig individualisierte Designs zu erstellen.

Was macht Tailwind besonders?

Utility-First-Ansatz

Im Mittelpunkt von Tailwind steht der Utility-First-Ansatz. Statt wie in traditionellen CSS-Stilen Klassen für spezifische Komponenten zu erstellen (z.B. .button oder .card), bietet Tailwind eine Vielzahl von vordefinierten Klassen, die direkt auf einzelne Eigenschaften wie Farbe, Grösse, Abstand oder Positionierung abzielen. Dadurch wird der Entwicklungsprozess beschleunigt, da sich Entwickler auf das Erstellen der Benutzeroberfläche konzentrieren können, ohne ständig in CSS-Dateien wechseln zu müssen.

Ein Beispiel könnte so aussehen:

```
<button class="bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-700">
    Klick mich!
</button>
```

In diesem Beispiel definiert jede Klasse eine bestimmte Eigenschaft:

- bg-blue-500 legt den Hintergrund auf eine blaue Farbe fest.
- text-white sorgt dafür, dass der Text weiss ist.
- py-2 und px-4 setzen vertikale und horizontale Innenabstände.
- rounded fügt abgerundete Ecken hinzu.
- hover:bg-blue-700 ändert die Hintergrundfarbe, wenn der Button mit der Maus überfahren wird.

Keine vordefinierten Komponenten

Tailwind liefert keine fertigen Design-Komponenten wie Buttons oder Karten. Stattdessen erlaubt es Entwicklern, alle Designentscheidungen selbst zu treffen. Dies bedeutet maximale Flexibilität und völlige Freiheit beim Styling.

Konsistenz und Wiederverwendbarkeit

Durch die Nutzung von Utility-Klassen wird eine konsistente Designsprache über das gesamte Projekt hinweg sichergestellt. Alle Entwickler im Team verwenden dieselben Klassen, was die Lesbarkeit und

Wartbarkeit des Codes verbessert.

Nachteil gegenüber komponentbasierte Frameworks wie Bootstrap

Tailwind ist ein super Werkzeug, es gibt jedoch auch Nachteile, die wir beachten müssen. Da wir mit Tailwind grundsätzlich „CSS in HTML schreiben“, werden unsere HTML-Dokumente sehr unübersichtlich. So kann eine einfache Landing-Page wie folgt aussehen:

```
<div class="bg-gray-100 min-h-screen flex items-center justify-center">
  <div class="bg-white p-8 rounded-lg shadow-lg max-w-md w-full">
    <h1 class="text-2xl font-bold mb-4 text-center">Willkommen!</h1>
    <p class="text-gray-600 mb-6 text-center">
      Dies ist ein Beispiel für eine einfache Landingpage mit Tailwind CSS.
    </p>
    <button class="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-
blue-600">
      Mehr erfahren
    </button>
  </div>
</div>
```

Dieses Problem wird jedoch entschärft, sobald man Komponentbasiert mit z.B. React arbeitet.

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/ffit/2-jahr/tailwind/learningunits/lu01/grundlagen

Last update: **2025/11/05 08:09**