Liste der wichtigsten CSS units

An sich gibt es unzählige mögliche CSS units – hier sollen nur die wichtigsten vorgestellt werden.

1.1 Keine CSS unit, nur die Zahl (z.B. beim CSS-Befehl opacity)

Viele CSS-Befehle arbeiten ganz ohne Einheiten und brauchen nur eine Zahl.

Der CSS-Befehl opacity z.B. beschreibt die Durchsichtigkeit eines HTML-Elements mit einer Zahl zwischen 1 und 0, ohne CSS Einheit dazu.

1.2 Die CSS unit px: Pixel (=Bildpunkte)

Sehr genaue Einheit, reagiert nicht auf verschiedene Fenstergrößen oder Veränderungen in der Größe von Eltern-Elementen.

Nützlich, wenn z.B. der Text-Bereich einer Website unabhängig von der Fenstergröße immer eine gewisse Breite haben soll, um die Lesbarkeit zu erhöhen.

1.3 Die CSS unit %: Prozent vom Eltern-Element

Die CSS unit % versucht, den jeweiligen Wert basierend auf dem gleichen Wert des Eltern-Elements zu berechnen.

Nehmen wir an, es befindet sich ein DIV1 in einem anderen DIV2, wobei das DIV2 eine width von 300px aufweist.

Das DIV1 bekommt eine width von 80% zugewiesen.

Der Browser berechnet nun die tatsächliche Breite des DIV1, indem er die 300px vom DIV2 mit 80% multipliziert, was 240px ergibt.

Ändert sich die width vom DIV2, ändert sich die Breite vom DIV1 automatisch entsprechend mit.

% – Prozent vom Eltern-Element

- **Was heißt das?**
  Prozentwerte richten sich nach der Größe von dem Element, das "darüber" steht, also dem sogenannten Eltern-Element.
  Wenn dein Eltern-Element z. B. 500px breit ist, dann ist ein Kind-Element mit width: 50%; automatisch 250px breit – weil 50% von 500px eben 250px sind.

Einfaches Beispiel:

Stell dir eine Tafel Schokolade vor. Die Tafel ist dein Eltern-Element und hat 100 Stückchen (also 100%). Jetzt machst du ein kleineres Stück draus – z. B. die Hälfte (50%). Dein kleineres Stück wird dann 50 Schoko-Stückchen groß.

Noch ein Beispiel mit einem Bild:

1. **Eltern-Element**: Ein großer Kasten, z. B. ein div, ist 400px breit.
2. **Kind-Element**: Ein kleinerer Kasten, der sich IN dem großen Kasten befindet, bekommt width: 50%;.
3. **Ergebnis**:
   Der kleine Kasten wird 50% von 400px breit, also 200px.

Divi Spezifisch:

width und max-width in Divi – einfach erklärt

width: 80%;

- **Was macht das?**
  Die Breite der Reihe wird auf **80% der Breite des übergeordneten Elements** (also des Eltern-Elements) gesetzt.
  Beispiel: Wenn das übergeordnete Element (z. B. der Container) 1000px breit ist, dann ist die Reihe 80% davon, also 800px breit.

max-width: 1920px;

- **Was macht das?**
  Die maximale Breite der Reihe wird auf **1920px begrenzt**. Selbst wenn die width: 80%; theoretisch mehr Platz einnehmen würde (z. B. auf einem sehr großen Bildschirm), wird die Breite der Reihe niemals größer als 1920px.

Zusammen spielen width und max-width so:

1. **Bildschirm kleiner als 1920px**:
   Die Reihe nimmt 80% der Breite des übergeordneten Elements ein.
   Beispiel: Bei einem Bildschirm mit 1000px Breite hat die Reihe eine Breite von 800px (80%).
2. **Bildschirm größer als 1920px**:
   Die Reihe bleibt bei **maximal 1920px**, auch wenn 80% mehr Platz ergeben würde.
   Beispiel: Bei einem Bildschirm mit 4000px Breite würde die Reihe statt 3200px (80%) nur 1920px breit sein.

Warum macht man das?

- **width sorgt für Flexibilität:** Die Reihe passt sich an die Breite des Eltern-Elements oder des Bildschirms an.
- **max-width sorgt für Kontrolle:** Verhindert, dass die Reihe auf sehr großen Bildschirmen zu breit wird und Inhalte unlesbar oder unschön verteilt wirken.
-

Vielleicht hilft ein Vergleich mit einer Tüte Gummibärchen:

- Die Tüte füllt sich flexibel zu 80% (width), aber es gibt eine Obergrenze, wie viele Gummibärchen reinpassen können (max-width).

1.4 Die CSS units vw und vh: Prozent von der Fenster-Breite bzw. Fenster-Höhe

Die CSS units vw und vh funktionieren ähnlich wie die CSS unit %. Aber sie basieren nicht auf dem Eltern-Element, sondern auf der Fenster-Höhe bzw. Fenester-Breite.

100vw bedeutet volle Breite des Browser-Fensters, und 100vh bedeutet volle Höhe des Browser-Fensters.

Die beiden Einheiten können sehr nützlich sein, wenn es um responsive Design geht.

Was am besten benutzen?

- vw und vh, wenn etwas auf der Fenstergröße basieren soll (z.B. wenn ein Element die ganze Breite ausfüllen soll)
- px, wenn etwas eine genau bestimmte Größe braucht

- %, wenn etwas aufs Eltern-Element reagieren soll
- keine Einheit dort wo man keine braucht (z.B. opacity oder z-index)