

CSS

Css steht für **Cascading Style Sheet** und entscheidet wie die verschiedensten Html Elemente auszusehen haben. Durch ihnen lassen sich Form, Farbe, Grösse, Platzierung und vieles mehr abändern bzw. definieren.

Der Begriff „Cascading“ in CSS kommt von den Änderungsstufen welche höhere und niedrigere Prioritäten haben.

Das folgende Dokument zeigt euch sehr einfache Arten wie man welche Manipulationen durch CSS machen kann. Jedoch dem Styling mit CSS setzt ihr die Grenzen, je ausgefallener eure Ideen, desto spannender wird die ganze Sache.

Selektoren

Es gibt 3 Arten von Selektoren, welche oft genutzt werden.

1. Typselektoren
2. Klassenselektoren
3. ID-Selektoren

Typselektoren

Bei den Typselektoren werden die direkten Html Elemente angesprochen. Als eine kleinere Erläuterung hier ein Beispiel:

```
<head>
  <style>
    p {color: blue;}
  </style>
</head>
<body>
  <p>Hallo Welt</p>
</body>
```

Wie Ihr hier sehen könnt, werden hier **allen** p-Tags die Schriftfarbe blau verliehen. Um die Elemente ansprechen zu können müsst ihr das Element erwähnen und dem einen Style geben.

Klassenselektoren

Klassenselektoren geben euch die Möglichkeit nur bestimmten Elementen diesen Style zu geben und nicht allen. Das bringt Vorteile, wenn man beispielsweise nur die Abschnitte rot geschrieben haben möchte. Erneut ein Beispiel:

```
<head>
  <style>
    .Abschnitt {color: blue;}
  </style>
</head>
<body>
  <p>Hallo Welt</p>
  <p class="Abschnitt">Hallo Mond</p>
  <span class="Abschnitt">
    Hallo Sonne
  </span>
</body>
```

Wie ihr seht werden die p-Tags mit dem Class-Attribut: „Abschnitt“ blau gefärbt, wogegen der normale p-Tag weiterhin schwarz geschrieben wird. Was hier auch schön ersichtlich ist, das verschiedene Elemente mit demselben Klassenattribut versehen werden kann. Wichtig zu wissen ist, dass der Klassenselektor mit einem Punkt und dem Namen der Klasse angesprochen wird.

ID-Selektoren

ID Kurzform von Identity steht für die einmalige Bestimmung von egal was. In CSS verwendet man diese Selektoren um nur einen einzigen Elementen abändern zu können und auch nur diesen.

```
<head>
  <style>
    .Abschnitt {color: blue;}
    #Wichtig {color: red;}
  </style>
</head>
<body>
  <p id="Wichtig">Hallo Welt</p>
  <p class="Abschnitt">Hallo Mond</p>
  <span class="Abschnitt">
    Hallo Sonne
  </span>
</body>
```

Eine ID wird über einen # angesprochen, auch hier mit der Bezeichnung in ihm. Wichtig zu wissen einen ID mit derselben Bezeichnung, dürft ihr nur **einmal** auf der Webseite vergeben.

Natürlich lassen sich diese auch kombinieren

```
<p class="Abschnitt" id="Wichtig">
  Hallo Welt
</p>
```

Aber da könnt ihr selbst herumexperimentieren und euch austoben ☺.

3 Arten um CSS zu implementieren

1. Inline CSS
2. Internal CSS
3. External CSS

Inline CSS

Beginnen wir zu Erst mit der Inline Methode. Erstellt dafür einen Paragraphen (<p></p>) und ergänzt diesen wie folgt:

```
<p style="color: blue;">Hallo Welt</p>
```

In dieser Variante erreichen wir die grösste Priorität um die Elemente anpassen zu können, da hier direkt in den Quellcode des Elementes geschrieben wird.

Der grosse Nachteil bei Inline CSS ist, dass bei jedem einzelnen Element das gleiche Styling ergänzt werden muss. Beispiel:

```
<p style="color: blue;">Hallo Welt</p>
<p style="color: blue;">Hallo Mond</p>
```

Internal CSS

Als nächstes folgt die Internal CSS Version. In dieser ändern wir das Element ausserhalb des eigentlichen Elementes.

```
<head>
  <style>
    p {color: blue;}
  </style>
</head>
<body>
  <p>Hallo Welt</p>
</body>
```

In dieser Variante schreiben wir den CSS Inhalt in das gleiche Dokument, aber nicht mehr in dasselbe Element. Diese Variante hat eine niedrigere Priorität als die obere, jedoch verhindert man dadurch redundante Daten (sich immer wiederholende Codezeilen).

Im folgenden Beispiel seht ihr genau wie sich die wiederholenden Codezeilen verabschieden:

```
<head>
  <style>
    p {color: blue;}
  </style>
</head>
<body>
  <p>Hallo Welt</p>
  <p>Hallo Mond</p>
</body>
```

Nun werden beide Paragraphen blau geschrieben, mit dem Unterschied, dass die blaue Schrift nur einmal definiert wurde.

External CSS

Als External CSS bezeichnet man eine Datei, welche in die eigentliche HTML-Datei hineingeladen wird und so genutzt wird. Hier ein kleineres Beispiel:

style.css:

```
<style>
  p {color: blue;}
</style>
```

xy.html:

```
<head>
  <link rel="stylesheet" href="style.css"/>
</head>
<body>
  <p>Hallo Welt</p>
  <p>Hallo Mond</p>
</body>
```

yz.html:

```
<head>
  <link rel="stylesheet" href="style.css"/>
</head>
<body>
  <p>Hallo Welt</p>
  <p>Hallo Mond</p>
</body>
```

Hier haben wir denselben Effekt wie bei der Internal CSS, jedoch mit dem Unterschied, dass diese Variante nun für weitere .html Files genutzt werden kann. Wie ihr raten könnt hat diese Variante die niedrigste Priorität.

Welcher dieser Varianten ihr verwenden möchtet ist euch überlassen, aber nach meiner Meinung nach ist die External CSS Variante die sinnvollste, da die niedrige Priorität über den Kaskaden-Effekt und !important-Ansatz ausgeglichen werden kann.

Kaskaden-Effekt und !important

Sowohl mit dem Kaskaden-Effekt, als auch mit dem !important Attribut, lassen sich CSS Styles priorisieren. Das bedeutet somit hat man die Möglichkeit bereits vorgegebene Styles, durch die Webseite selbst als auch durch Browser, entgegenzuwirken.

Hier ein kleines Beispiel für den Kaskaden-Effekt:

```
<head>
  <style>
    p {color: blue;}
    div p{color: red;}
  </style>
</head>
<body>
  <p>Hallo Welt</p>
  <div>
    <p>Hallo Mond</p>
  </div>
</body>
```

Da durch den zweiten Befehl genauer definiert wird, welchen p-Tag wir meinen, entsteht ein Kaskaden-Effekt wodurch die beiden p-Tag differenziertere Farben erhalten. Obwohl laut erster p-CSS Definition alle p-Tags blau sein müssten.

Durch das !important Attribut wird einem bestimmten Style die höchste Prioritätsstufe verliehen. Ein kleines Beispiel zum Einleuchten:

```
<head>
  <style>
    p {color: blue !important;}
  </style>
</head>
<body>
  <p style="color: red;">Hallo Welt</p>
  <p style="color: red;">Hallo Mond</p>
</body>
```

Wir haben nun durch Inline-CSS bestimmt, dass die Schriftfarbe Rot zu sein hat. Jedoch durch das !important Attribut oben, wird die Schriftfarbe der p-Tags nun blau. Man gibt durch das !important Attribut, der blauen Schriftfarbe die höchste Priorität.

Weitere sehr nützliche CSS-Styles findet ihr auf folgender Seite:

<https://www.w3schools.com/cssref>