

CSS SELEKTOREN, VERERBUNG & SPEZIFITÄT

Themen:

- CSS Selektoren im Detail
- CSS Inheritance (Vererbung)
- CSS Spezifität
- Der Cascading Algorithmus

CSS Selektoren - Übersicht

Was sind CSS Selektoren?

CSS Selektoren bestimmen, welche HTML-Elemente von CSS-Regeln betroffen sind.

Kategorien:

- **Element-Selektoren:** p, div, h1
- **Klassen-Selektoren:** .klasse
- **ID-Selektoren:** #id
- **Attribut-Selektoren:** [attribute]
- **Pseudo-Selektoren:** :hover, ::before
- **Kombinatoren:** , >, +, ~ (Leerzeichen: Nachfahre, >: Direkter Nachfahre, +: Nächstes Element, ~: Geschwister)

Beispiel:

```
/* Element */          p { color: blue; }
/* Klasse */          .text { color: red; }
/* ID */              #header { color: green; }
```

Element- und Klassen-Selektoren

Element-Selektoren - Wählen alle Elemente eines bestimmten Typs aus

Klassen-Selektoren - Wiederverwendbare Styles für mehrere Elemente - Mit . Punkt eingeleitet

Beispiel:

```
<p>Normaler Text</p>
<p class="wichtig">Wichtiger Text</p>
<p class="wichtig hervorgehoben">Sehr wichtig</p>

p { color: black; }                      /* Alle Paragraphen */
.wichtig { color: red; }                  /* Klasse wichtig */
.hervorgehoben { font-weight: bold; }    /* Klasse hervorgehoben */
```

ID- und Attribut-Selektoren

ID-Selektoren

- Eindeutige Identifikation eines Elements
- Mit # Hash eingeleitet

Attribut-Selektoren

- Wählen Elemente basierend auf Attributen aus
- Vielseitige Matching-Möglichkeiten

Beispiele:

```
<div id="header">Kopfbereich</div>
<input type="email" required>
<a href="mailto:test@example.com">Email</a>

#header { background: blue; }          /* ID-Selektor */
[required] { border: 2px solid red; }   /* Hat required Attribut */
[href^="mailto:"] { color: green; }    /* href beginnt mit mailto: */
```

Pseudo-Selektoren und Kombinatoren

Pseudo-Klassen (mit :)

- Wählen Elemente in bestimmten Zuständen
- :hover, :focus, :first-child, :nth-child()

Pseudo-Elemente (mit ::)

- Wählen Teile von Elementen
- ::before, ::after, ::first-line

Kombinatoren

```
/* Nachfahre */          .menu a { color: blue; }
/* Direktes Kind */     .menu > a { color: red; }
/* Nachfolger */        h1 + p { margin-top: 0; }
/* Geschwister */       h1 ~ p { color: gray; }
```

Beispiel:

```
a:hover { color: red; }           /* Hover-Zustand */
li:first-child { font-weight: bold; } /* Erstes li-Element */
p::first-line { text-transform: uppercase; } /* Erste Zeile */
```

CSS Inheritance (Vererbung)

Was ist CSS Inheritance?

Bestimmte CSS-Eigenschaften werden automatisch von Eltern- an Kindelemente weitergegeben.

Vererbbare Eigenschaften:

- Text: color, font-family, font-size, line-height
- Listen: list-style-type - Tabellen: border-collapse

Nicht vererbbar:

- Layout: width, height, margin, padding
- Hintergrund: background-color, border

Steuerung:

- inherit: Explizit erben
- initial: Auf Standardwert zurücksetzen
- unset: Auf vererbbar/initial zurücksetzen

HTML-Struktur:

```
<body>
  <div class="container">
    <p>Dieser Text ist blau (geerbt)</p>
    <p class="special">Spezieller Text</p>
    <span>Span-Text</span>
  </div>
</body>
```

CSS:

```
body {
  color: blue;          /* Vererbt sich */
  margin: 20px;         /* Vererbt sich NICHT */
}

.container {
  border: 1px solid black; /* Vererbt sich NICHT */
}

.special {
  color: inherit;      /* Explizit erben: wird blau */
  margin: inherit;     /* Explizit erben: wird 20px */
}
```

Ergebnis: Alle Texte sind blau, aber nur .special hat margin von 20px.

Inheritance Beispiele

CSS Spezifität

Was ist Spezifität?

Spezifität bestimmt, welche CSS-Regel angewendet wird, wenn mehrere Regeln auf dasselbe Element zutreffen.

Spezifitäts-Berechnung:

1. **Inline-Styles**: 1000 Punkte (`style="..."`)
2. **IDs**: 100 Punkte (`#id`)
3. **Klassen, Attribute, Pseudo-Klassen**: 10 Punkte (`.class`, `[attr]`, `:hover`)
4. **Elemente, Pseudo-Elemente**: 1 Punkt (`div`, `::before`)

Faustregel: Inline > IDs > Klassen > Elemente

Bei gleicher Spezifität: Die später definierte Regel gewinnt!

Spezifität Beispiele

HTML:

```
<p id="wichtig" class="text highlight">Welche Farbe hat dieser Text?</p>
```

CSS Regeln und ihre Spezifität:

p { color: black; }	/* 0-0-0-1 = 1 */
.text { color: blue; }	/* 0-0-1-0 = 10 */
.highlight { color: yellow; }	/* 0-0-1-0 = 10 */
#wichtig { color: red; }	/* 0-1-0-0 = 100 */
p.text.highlight { color: green; }	/* 0-0-2-1 = 21 */
#wichtig.highlight { color: purple; }	/* 0-1-1-0 = 110 */

Ergebnis: Der Text ist **purple** (höchste Spezifität: 110)

Bei Gleichstand:

```
.text { color: blue; }      /* Spezifität: 10 */
.highlight { color: yellow; } /* Spezifität: 10, aber später → gewinnt! */
```

Der CSS Cascading Algorithmus

Der Cascading Algorithmus bestimmt in folgender Reihenfolge, welcher Wert angewendet wird:

- 1. Origin und Importance** - Browser-Standard < Benutzer < Autor - !important kehrt Reihenfolge um
- 2. Spezifität** - Inline (1000) > ID (100) > Klasse (10) > Element (1)
- 3. Source Order (Reihenfolge)** - Bei gleicher Spezifität: später definiert gewinnt
- 4. Inheritance** - Vererbbare Eigenschaften von Elternelementen

Beispiel des kompletten Prozesses:

```
/* 1. Browser Standard: color: black */
/* 2. Autor CSS: */ p { color: blue; }          /* Spez: 1 */
/* 3. Autor CSS: */ .text { color: red; }        /* Spez: 10 → gewinnt */
/* 4. Autor CSS: */ .other { color: green; }     /* Spez: 10, aber früher */
/* 5. HTML: */      style="color: yellow"        /* Spez: 1000 → gewinnt */
```

Praktische Tipps

Best Practices für CSS Selektoren:



DO:

- Verwende Klassen für wiederverwendbare Styles
- Halte Selektoren so einfach wie möglich
- Nutze semantische Klassennamen
- Strukturiere CSS logisch (generell → spezifisch)



DON'T:

- Vermeide zu spezifische Selektoren: .menu ul li a
- Verwende IDs nicht für Styling (nur für JavaScript) - Vermeide !important (außer in Ausnahmefällen)
- Nutze keine Inline-Styles für Design

Debugging-Tipp: Browser-Entwicklertools zeigen angewendete Regeln und deren Spezifität!

```
/* Gut */
.navigation-link { color: blue; }
.navigation-link:hover { color: red; }
```

```
/* Schlecht */
#header nav ul li a { color: blue !important; }
```

Ressourcen:

[MDN CSS Selectors](#)

[MDN CSS Specificity](#)

[MDN CSS Cascade](#)