Web Engineering + Design 1

CASCADING STYLE SHEETS (CSS)

Michael Gfeller

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Lernziele

Die Teilnehmer…

- **… kennen unterschiedliche Varianten wie CSS definiert werden kann und können die beste Variante auswählen**
- **… können CSS Selektoren interpretieren und anwenden**
- **… können geeignete CSS Selektoren für ein Problem definieren**
- **… können die Specificity von einer CSS-Regel berechnen**
- **… können erkennen, welche CSS Regeln auf bestimmte Elemente angewendet werden**
- **… können mit CSS eine Webseite Stylen**
- **… kennen die Eigenschaften von CSS und können diese anwenden und erklären**
- **… können mit den Chrome Dev. Tools umgehen (Teil CSS)**

Lernziele

Die Teilnehmer…

- **… kennen den Unterschied zwischen Pixel und rem/em**
- **… kennen das Problem von relativen Einheiten und Eigenschaften-Vererbung.**
- **… wissen, welche Eigenschaften vererbt werden.**
- **… können unterschiedliches CSS für Druck und Bildschirm definieren.**
- **… verstehen was der «Fluss» ist und wie Elemente sich darin verhalten**
- **… können Elemente aus dem «Fluss» nehmen und können mit «Position» umgehen**
- **... können den Unterschied zwischen den unterschiedlichen «Display» erklären und an Beispielen anwenden**
- **... kennen verschiedene Varianten wie HTML Elemente «verstecken» werden können und sind in der Lage diese korrekt anzuwenden.**

Lernziele

Die Teilnehmer…

- **… wissen was das «Box Model» ist**
- **… können «Padding» und «Margin» korrekt anwenden**
- **… können «box-sizing» erklären und korrekt anwenden**
- **… kennen den Unterschied zwischen «Border» und «Outline»**
- **… können CSS Eigenschaften selbständig nachschlagen und anwenden**

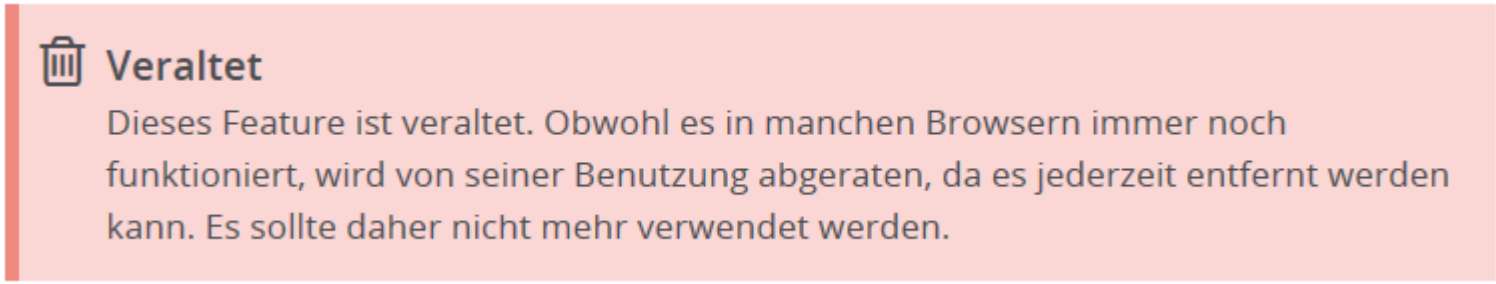
Inhaltsverzeichnis

Inhaltsverzeichnis

WARUM CSS?

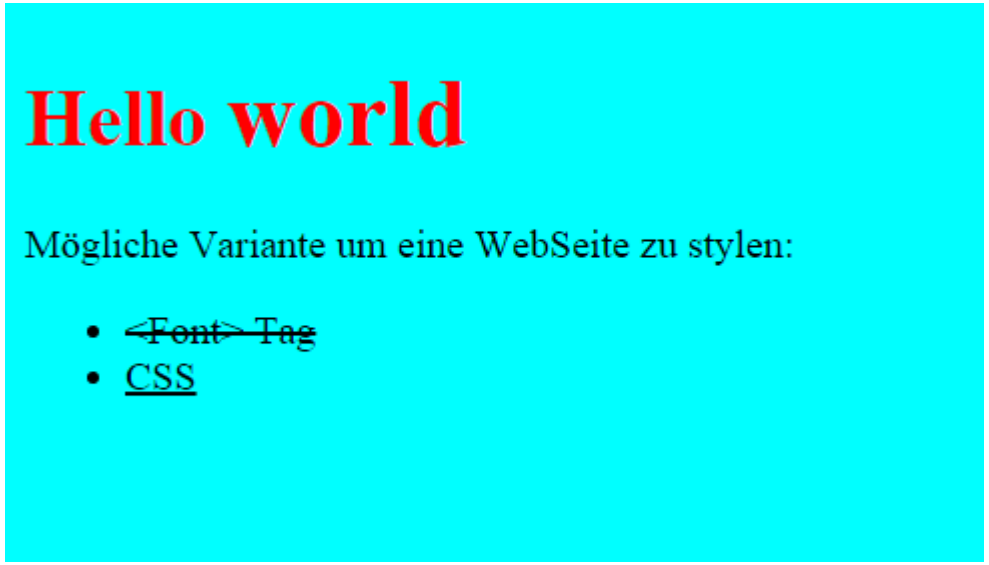
Weshalb CSS

- **HTML wurde entworfen um Information zu strukturieren**
- **Mit weiterer Popularität wurden die HTML Tags zweckentfremdet bzw. mit Style Tags ergänzt:**
  - font & blink Tag
  - u & i Tag
  - table Tag wird fürs Layouten «missbraucht»
- **CSS wurde entworfen um ein einheitliches und wiederverwendbares Styling Tool zu Verfügung zu stellen.**
  - Mit HTML5 wurden alle Style Tags als deprecated eingestuft und dürfen nicht mehr verwendet werden.
    - z.B. Font Tag: https://developer.mozilla.org/de/docs/Web/HTML/Element/font



Weshalb CSS: Beispiel (schlecht)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body bgcolor="aqua">
  <h1><font color="red">Hello <big>world</big></font></h1>
  <p>Mögliche Variante um eine WebSeite zu stylen:</p>
  <ul>
      <li><strike>&lt;Font&gt; Tag</strike></li>
      <li><u>CSS</u></li>
  </ul>
</body>
</html>
```



- **Webstorm erkennt deprecated Tags und markiert diese:**

```
<h1><font color="red">Hello <big>world</big></font></h1>
<p>Mögliche Variante um eine WebSeite zu stylen:</p>
<ul>
    <li><strike>&lt;Font&gt; Tag</strike></li>
    <li><u>CSS</u></li>
</ul>
```

Vorteil von CSS

- **Klare Trennung zwischen Struktur und Style-Informationen**
- **Ermöglicht auf unterschiedliche Ausgabegrössen zu reagieren**
  - Fluides Design => WED2
- **Es ist möglich, unterschiedliche Styles für den jeweiligen Output Type zu definieren**
  - Print
  - Screen
- **Styles können auf Gruppen von Elementen angewendet werden**
  - Kein Copy & Paste
- **Styles können in Files ausgelagert werden**

Weshalb CSS: Beispiel

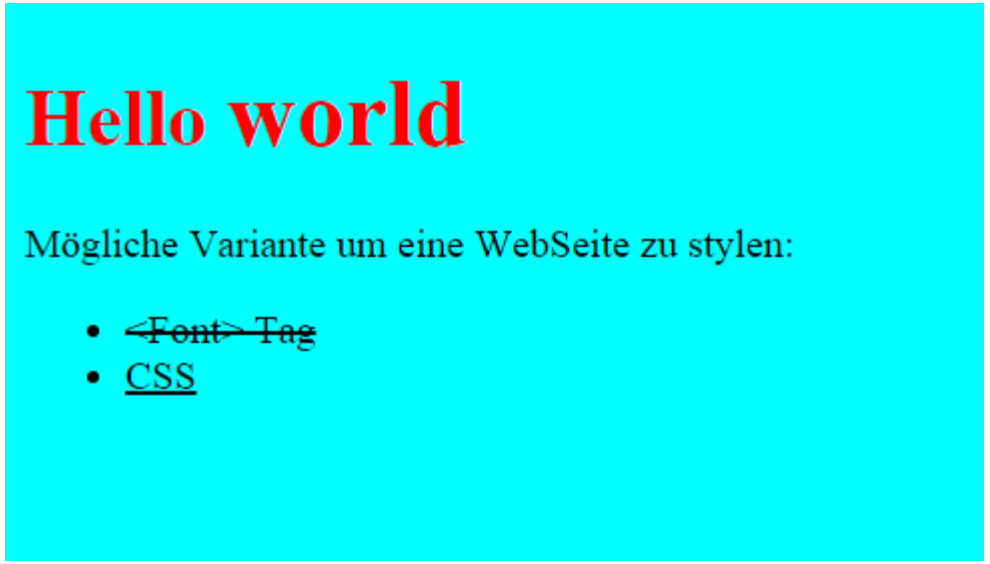
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        body{ background: aqua;}

        h1{ color: red; }

        strong{ font-size: 1.2em;}

        .deprecated{ text-decoration: line-through;}

        .important{ text-decoration: underline;}
    </style>
</head>
<body>
  <h1>Hello <strong>world</strong></h1>
  <p>Mögliche Variante um eine WebSeite zu stylen:</p>
  <ul>
      <li class="deprecated">&lt;Font&gt; Tag</li>
      <li class="important">CSS</li>
  </ul>
</body>
</html>
```



HTML ohne CSS?

- **Erste Web-Seite kam ohne Styling aus: http://info.cern.ch/hypertext/WWW/TheProject.html**
- **HTML alleine sieht «unschön» aus**
  - z.B. 20min ohne Styles http://www.20min.ch/

- de
- fr
- it

Inhalt A-Z

Suchen

20 minuten

- Schweiz
  - Zürich
  - Bern
  - Basel
  - Zentralschweiz
  - Ostschweiz
- Ausland
  - Panorama
- Wirtschaft

WAS IST CSS?

Was ist CSS

“Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents.”

Quelle: https://www.w3.org/Style/CSS/

Was ist CSS

- **Cascading Style Sheets**
  - Cascading
    - Später in der Vorlesung
  - Style Sheet - Language
    - Formale Sprachen um das Erscheinungsbild von Dokumenten bzw. Benutzeroberflächen festzulegen.
- **CSS ist ein «living standard»**
  - Wir immer weiter entwickelt
  - Aktuell ist CSS3
    - *„There will never be a CSS4“*: http://www.xanthir.com/b4Ko0
  - Seit CSS3 werden die Features mittels Modules definiert
    - Background & Borders 3: https://drafts.csswg.org/css-backgrounds-4/
    - FlexBox 1: https://www.w3.org/TR/css-flexbox-1/
    - Selector 4: https://drafts.csswg.org/selectors/
      - Oft als Synonym für CSS4 verwendet

Was ist CSS

- **Beim CSS können keine Versionen angegeben werden**
  - CSS wird vom Browser immer als neuste Variante interpretiert.
    - Selten relevant z.B. FlexBox
- **Browser Support für neue CSS Feature kann überprüft werden**
  - http://caniuse.com/#feat=css-sel3
  - http://caniuse.com/#feat=transforms3d
- **CSS besteht aus Regeln welche das Aussehen des Dokumentes verändert**
- **CSS Regeln werden auf HTML / SVG angewendet**
- **CSS ist auch ein Standard**
  - https://www.w3.org/Style/CSS/

CSS DEFINIEREN

Variante 1: Inline

- **Jedes HTML-Element hat ein Property «style»**
- **Styles können als Attributwert angegeben werden**
- **Verwendungszweck**
  - Ausprobieren
  - Folien z.B. reveal.js
  - WYSIWYG-Editoren

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
    <span style="background: green; color: pink">Hello world</span>
</body>
</html>
```

Einschub Syntax: CSS Regel

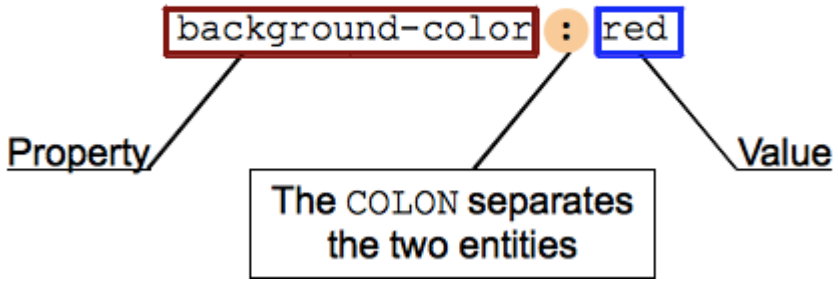
```
span, div { background: blue; }
```

CSS-Regel besteht aus

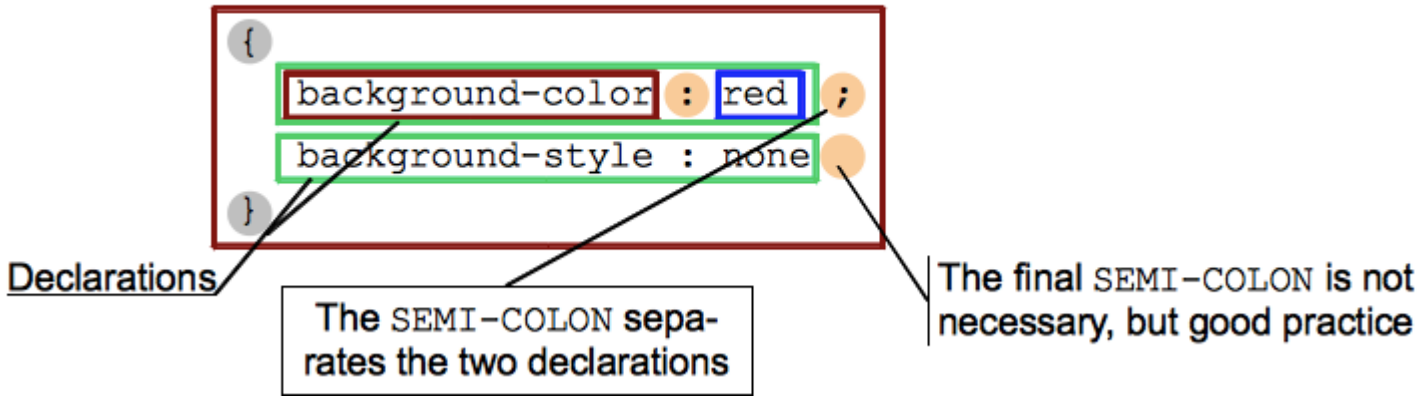
- **Selektoren**
  - z.B. für alle `span`'s und `div`'s im Dokument
- **Property : Value**
  - z.B. setzt die Hintergrundfarbe für alle selektierten Elemente auf blau

Einschub Syntax: CSS Regel

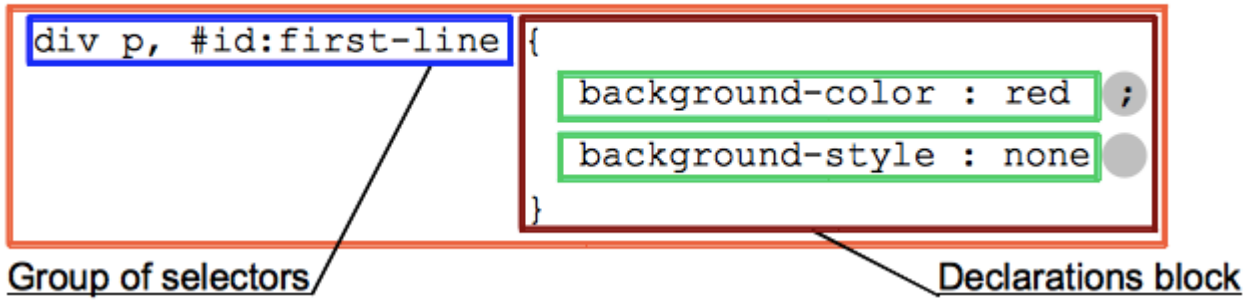
A CSS declaration :



A CSS declarations block:



A CSS ruleset (or rule):



https://developer.mozilla.org/de/docs/Web/CSS/Syntax

Einschub Syntax: CSS Kommentare

Kommentare wie folgt

```
/* multiline comment */


/*
   multiline comment
*/
```

So nicht

```
// so keine CSS kommentare!
```

Einschub Syntax: Case sensitive

- **Selektoren sind «Case-sensitive»**
- **Tipp: Immer klein schreiben**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        #Container{
            background: gray;
        }
    </style>
</head>
<body>
    <div id="container">
        <span>Hello world</span>
    </div>
</body>
</html>
```

Repetition HTML-Tags: Style Tag

Content categories	Metadata content, and if the `scoped` attribute is present: flow content.
Permitted content	Text content matching the `type` attribute, that is `text/css`.
Tag omission	Neither tag is omissible.
Permitted parent elements	If the **scoped** attribute is *not* present: where metadata content is expected or in a `<noscript>` element itself a child of `<head>` element. If the **scoped** attribute is present: where flow content is expected, but before any other flow content other than inter-element whitespace and `<style>` elements, and not as the child of an element whose content model is transparent.
DOM interface	`HTMLStyleElement`

https://developer.mozilla.org/de/docs/Web/HTML/Element/style
https://html.spec.whatwg.org/multipage/semantics.html#the-style-element

Variante 3: File

- **Mit dem <link> Tag kann ein externes CSS File geladen werden**
  - rel-attribute
- **Im <head>**
- **Verwendungszweck**
  - Immer! (ausser man hat einen sehr guten Grund)

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="file.css">
</head>
<body>
    <span>Hello world</span>
</body>
</html>
```

```css
/*file.css:*/
span{
    background: blue;
    color: pink;
}
```

Variante 3a: File mit Screen

- **Beim «link» Tag kann mit media=«print» bzw. «screen» angegeben werden, ob das Stylesheet für das jeweilige Medium angewendet werden soll.**
- **Beispiele**
  - Werbungen beim Drucken ausblenden
  - Einblenden von einem speziellen Druck-Footer

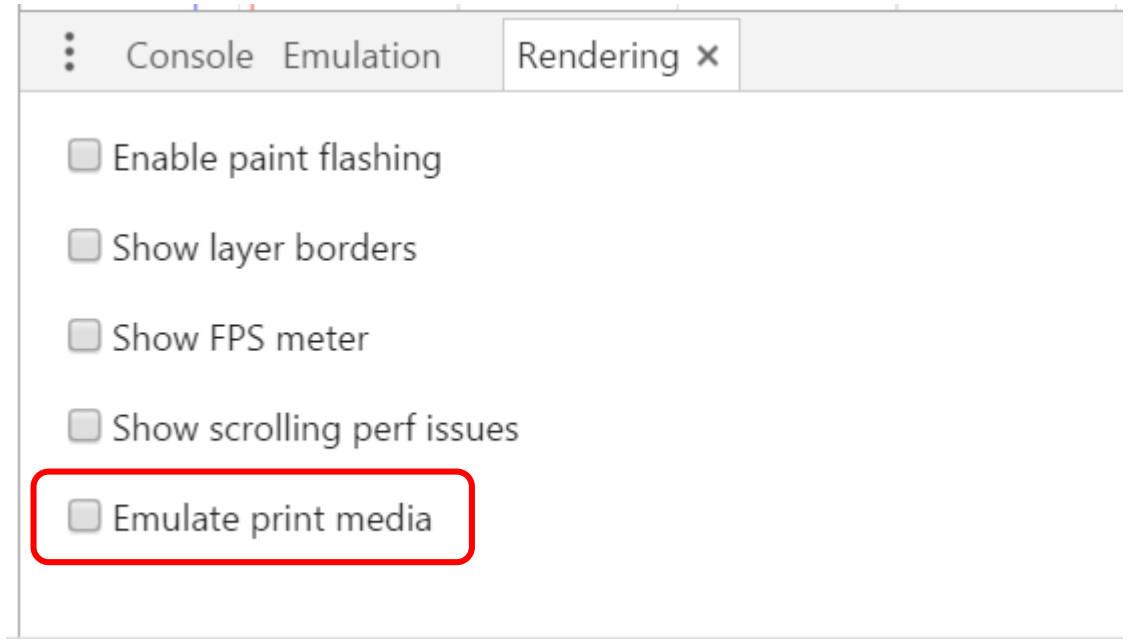
```html
<html lang="en">
<head>
    <link rel="stylesheet"  href="print.css" media="print"/>
    <link rel="stylesheet"  href="screen.css" media="screen"/>
</head>
<body>
    <p>Hello World</p>
</body>
</html>
```

```css
/*screen.css*/
p{
    color: blue;
}

/*print.css*/
p{
    color: red;
}
```

Variante 3a: File mit Screen

Die Chrome Dev. Tools können den «Print» Simulieren



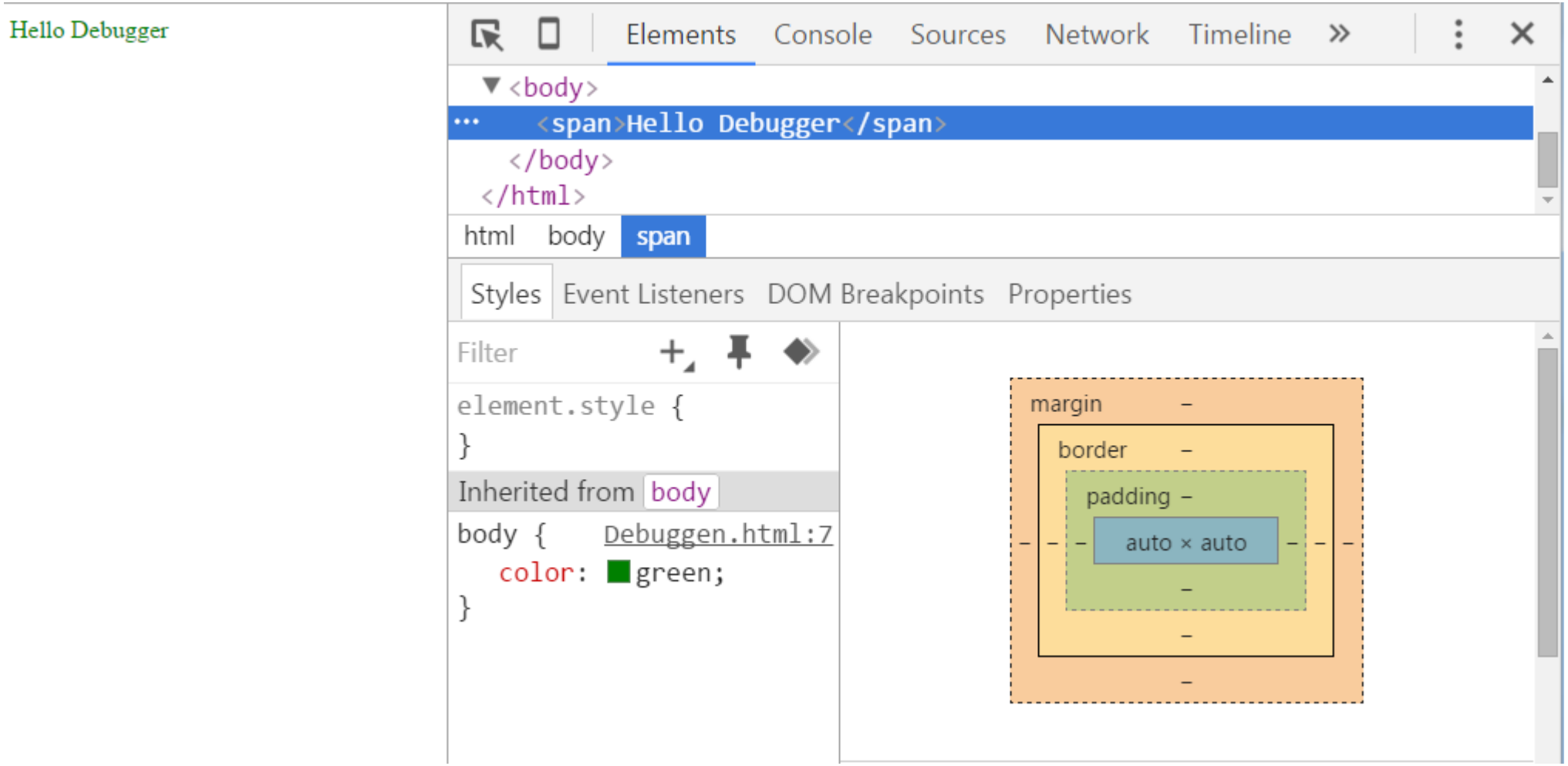
Variante 2: Style

- **Inhalt innerhalb von <style> … </style> wird als Style interpretiert**
- **Im <head>**
- **Verwendungszweck**
  - Ausprobieren Advanced
  - Vorlesung / Prüfungsaufgaben ☺
  - Bessere Performance
- **Ausblick: Diese Variante erlaubt Scoped Styles**
  - Sehr minimaler Support => aktuell nicht sinnvoll

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        span{
            background: red;
        }
    </style>
</head>
<body>
    <span>Hello world</span>
</body>
</html>
```

CSS «DEBUGGEN»

Chrome


Hello Debugger
Elements
Console
Sources
Network
Timeline
<body>
Hello Debugger
</body>
</html>
html
body
span
Styles
Event Listeners
DOM Breakpoints
Properties
Filter
element.style {
}
Inherited from body
body {
Debuggen.html:7
color: green;
}
margin
border
padding
auto × auto

SELEKTOREN BASICS

Selektor

Der Selektor ist für die Selektion von Elementen verantwortlich. Auf diese Selektion werden die Styles angewandt.

Es gibt verschiedene Typen von Selektoren

- **Typ**
- **ID**
- **Universal**
- **Klassen**
- **Attribut**
- **Pseudo-Elemente**
- **Pseudo**

Selektoren können kombiniert werden

Type-Selektor

- **Selektiert alle HTML-Elemente vom angegeben Type**
  - z.B. alle Links / alle Inputs / alle …

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        a{
            color: red;
        }
    </style>
</head>
<body>
    <a href="#" target="_blank">Link1</a>
    <a href="#" target="_blank">Link2</a>
    <a href="#" target="_blank">Link3</a>
    <a href="#" target="_blank">Link4</a>
</body>
</html>
```

ID Selektor

- **ID Selektor wird mit #id erstellt**
- **Der performanteste Selektor**
- **Hinweis:**
  - ID Selektoren sollten in modularem CSS nicht verwendet werden
    - Mehr dazu in WED2/3

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        #container {
            background: gray;
        }
    </style>
</head>
<body>
    <div id="container">
        <span>Hello world</span>
    </div>
</body>
</html>
```

KOMBINATOREN

Kombinatoren

- **Es gibt 4 verschiedene Varianten von Kombinatoren**
  - Oft benutzt
    - Kindselektor: e > f
    - Nachfahrenselektor: e f
  - Für Spezialfälle
    - Nachbarselektor: e + f
      - The lobotomized owl selector: * + *
        - Ausführlich: http://alistapart.com/article/axiomatic-css-and-lobotomized-owls
    - Geschwisterselektor: e ~ f

- **Zum nachlesen: https://wiki.selfhtml.org/wiki/CSS/Selektoren/Kombinator**

Kindselektor / Nachfahrenselektor

Kindselektor (Child Selector):

Werden zwei einfache Selektoren durch die schliessende spitze Klammer („>“) miteinander verbunden, z.B. „E > F“, so wird das Element F nur dann angesprochen, wenn es Kindelement eines E-Elements ist.

Nachfahrenselektor (Descendant Selector):

Werden zwei einfache Selektoren durch ein Leerzeichen („ “) miteinander verbunden, z.B. „E F“, so wird das Element F nur dann angesprochen, wenn es Nachfahre eines E-Elements ist.

```
<html lang="en">
<head>
    <style>
        body > ul {                    1
            list-style-type: decimal;
        }
        body ul {                      2
            opacity: 0.5;
        }
    </style>
</head>
<body>
    <ul>
        <li>
            <a href="#" target="_blank">AAAAAAAAA</a>
        </li>
        <ul>
            <li>
                <a href="#" target="_blank">C1</a>
            </li>
        </ul>
    </ul>
</body>
</html>
```

Kindselektor / Nachfahrenselektor

```
<html lang="en">
<head>
    <style>
        ol > li{ 1
            text-decoration: line-through;
        }
        ol li{ 2
            background: gray;
        }
    </style>
</head>
<body>
    <ol>
        <li>
            <a href="#" target="_blank">AAAAAAAAA</a>
        </li>
        <ul>
            <li>
                <a href="#" target="_blank">A1</a>
            </li>
        </ul>
        <li>
            <a href="#" target="_blank">BBBBBBBBBB</a>
        </li>
    </ol>
</body>
</html>
```

Nachbarselektor / Geschwisterselektor

Nachbarselektor (Adjacent sibling Selector):

Werden zwei einfache Selektoren durch das Pluszeichen („+“) miteinander verbunden, z.B. „E + F“, so wird das Element F nur dann angesprochen, wenn es im Elementbaum direkt auf ein E-Element folgt, also der direkte Nachbar ist.

Geschwisterselektor (General sibling Selector):

Werden zwei einfache Selektoren durch die Tilde („~“) miteinander verbunden, z.B. „E ~ F“, so werden alle F-Elemente angesprochen, die im Elementbaum in derselben Ebene auf ein E-Element folgen - unabhängig davon, ob sich zwischen den Elementen weitere, im Selektor nicht genannte, Elemente befinden.

```
<html lang="en">
<head>
<style>
        h1 + p{ 1
            font-weight: bold;
        }
        p + p{ 2
            color: green;
        }
        h1 ~ p{ 3
            text-decoration: underline;
        }
    </style>
</head>
<body>
<aside>
    <h1>Wichtig</h1>
    <p>Erster Paragraph</p>
    <span>Blablalblablalbllalbnllalaaa</span>
    <span>Blablalblablalbllalbnllalaaa</span>

    <p>Blablalblablalbllalbnllalaaa</p>

    <p>Blablalblablalbllalbnllalaaa</p>

    <p>Blablalblablalbllalbnllalaaa</p>
</aside>
</body></html>
```

UNIVERSAL SELEKTOR

Universal

- **Ein Universal Selektor besteht aus einem Sternzeichen „*“ (der Asterisk)**
- **Selektiert alle Elemente**
- **Wird oft für globale CSS-Resets gebraucht.**
  - Später mehr dazu

```
* {
    margin :0px;
    padding: 0px;
    box-sizing: border-box;
}
```

- **Kann bei Kombinatoren verwendet werden um in der HTML Strukturen weiter in die Tiefe zu navigieren z.B. Enkel selektieren**

Universal

```
<html lang="en">
<head>
    <style>
        li > * {                    ①
            text-decoration: underline;
        }
        body * span {               ②
            opacity: 0.5;
        }
    </style>
</head>
<body>
    <ul>
        <li>
            <span>A</span>
        </li>
        <li>
            <i>B</i>
        </li>
        <li>
            <strong>C</strong>
        </li>
    </ul>
    <span>D</span>
</body>
</html>
```

KLASSEN SELEKTOR

CSS Klasse

- **CSS Klasse hat nichts mit einer JavaScript bzw. Java Klasse zu tun!**
- **CSS definiert einen Klassen Selektor Type**
- **Dieser Type ist für Gruppierung von «gleichgesinnten» HTML-Elementen**
- **Ein Klassen Selektor wird mit einem . (Punkt) definiert z.B:**
  - .info .alert .deprecated
- **Jedes HTML-Element hat ein Property «class»**
  - Mehrere Klassen mit « » (Leerzeichen) getrennt
- **Kombinationen mit andern Selektoren sind möglich z.B:**
  - *div.box* selektiert alle «div»s mit der Klasse «box»
  - *.box.danger* selektiert nur Elemente, welche beide Klassen besitzen
  - Best Practices: Neue CSS Klasse für diese Konstellation erstellen

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .box {
            border : 1px black solid;
        }
        .box-danger{
            background : red;
        }
        .box-info{
            background : yellow;
        }
    </style>
</head>
<body>
    <div class="box box-danger">Warnung!</div>
    <div class="box box-info">Info!</div>
</body>
</html>
```

CSS Klasse

- **Die Namen von CSS-Klassen sollten generisch gewählt sein.**
- **Die Klassen sollten so granular wie möglich sein.**
  - z.B. Aufsplitten von box und box-error
  - Mehr in WED2/3
- **In den meisten Fällen sind CSS Klassen ID-Selektoren / Typen vorzuziehen**
  - Wiederverwendbarkeit
- **Namen von Klassen sollten «generisch» sein und die den Sinn von der Klasse beschreiben und nicht das Aussehen z.B.:**

```
/*schlecht*/
.red{
    background: red;
}

/*oke*/
.alert{
    background: red;
}
```

ATTRIBUTE SELEKTOR

Attribute Selektor

- **Seit CSS2 kann anhand von Attributen und Attributwerten selektiert werden.**
- **Überprüfen ob Attribute vorhanden ist:**
  - *[Attribute]
    - * ist optional
  - h1[Attribute]
  - .class[Attribute]
  - #id[Attribute]
- **Beispiel**

```
a[href]{
    color: green;
}
```

Attribute Selektor

- **Es können auch Werte überprüft werden**
  - [a=v]
    - Überprüft ob der Wert von Attribute gleich dem Wert v ist. **Komplett**
  - [a~=v]
    - Überprüft ob der Wert von Attribute das Wort v beinhaltet. **Alleinstehend**
  - [a|=v]
    - Überprüft ob der Wert von Attribute mit dem Wort v startet. **Alleinstehend**
  - [a^=v]
    - Überprüft ob der Wert von Attribute mit dem Wert v startet.
  - [a$=v]
    - Überprüft ob der Wert von Attribute mit dem Wert v endet.
  - [a*=v]
    - Überprüft ob der Wert von Attribute das Wert v beinhaltet.

Attribute Selektor

```
<html>
<head>
<style>
        [type=submit] {font-weight: bold;}
        [title~=Berlin] {color: green;}
        [hreflang|=de] {font-style: italic;}
    </style>
</head>
<body>

<p>
     <input type="text" value="Beispiel">
     <input type="submit" value="Absenden">
</p>

<ul>
    <li title="Berlin: eine schöne Stadt">Viele Informationen.</li>
    <li title="Die Stadt Berlin besteht aus mehreren Stadteilen">Mehr Informationen.</li>
    <li title="Das Brandenburger Tor steht in Berlin">Noch mehr Informationen.</li>
</ul>

<ul>
    <li><a href="http://www.example.org/" hreflang="de">Beispielverweis</a></li>
    <li><a href="http://www.example.com/" hreflang="de-at">Beispielverweis</a></li>
    <li><a href="http://www.example.net/" hreflang="en-de">Beispielverweis</a></li>
</ul>
</body>
</html>
```

Attribute Selektor

```
<html>
<head>
<title>CSS-Beispiel: Attributselektor</title>
    <style>
        a[href^="http://"] { font-weight: bold }
        a[href$=".pdf"]    { color: green }
        a[href*="wiki"]    { font-style: italic }
    </style>
</head>
<body>
<h1>Teilübereinstimmungen</h1>
<ul>
    <li>
        <a href="http://example.com/">Beispiellink</a>
    </li>
    <li>
        <a href="https://example.com/example.pdf">Ein PDF-Dokument</a>
    </li>
    <li>
        <a href="http://wiki.selfhtml.org/">Das selfhtml wiki</a>
    </li>
    <li>
        <a href="https://example.org/WIKI/example.pdf/">Ein PDF-Dokument</a>
    </li>
</ul>
</body>
</html>
```

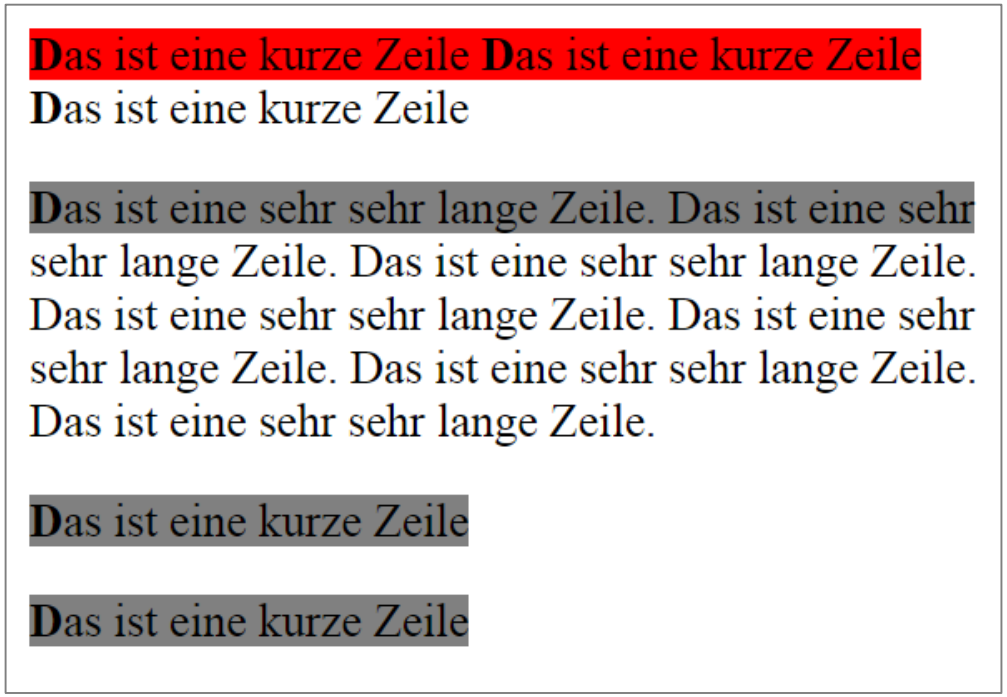
1 2 3

PSEUDO ELEMENT

Pseudoelement

- **Frage: Wie ist es möglich, mit CSS folgende Darstellung zu erzeugen?**

```
<div class="container">
    <span>Das ist eine kurze Zeile</span>
    <span>Das ist eine kurze Zeile</span>
    <span>Das ist eine kurze Zeile</span>
    <p>Das ist eine sehr sehr lange Zeile….</p>
    <p>Das ist eine kurze Zeile</p>
    <p>Das ist eine kurze Zeile</p>
</div>
```



Pseudoelement

- **Pseudoelemente erlauben es bestimmte Teile eines Elementes zu formatieren.**
- **::first-line**
  - Selektiert die erste Textzeile eines Elementes
- **::first-letter**
  - Selektiert den ersten Buchstabe eines Elementes
- **Zeigt nur Wirkung bei Block- und blockähnlichen Elementen**
  - Inline-block
  - Listenpunkte
  - Tabellenüberschriften und –zellen

```
<style>

    p::first-line{
        background: gray;
    }
    .container::first-line{
        background: red;
    }
    .container ::first-letter{
        font-weight: bold;
    }
    span{
        display: inline-block;
    }
</style>
```

::before ::after

- **::before und ::after erzeugen ein neues Element, welches vor bzw. nach dem selektiertem Element eingefügt wird.**

```
<html lang="en">
<head>
<meta charset="UTF-8">
<style>
        .question::before{
            font-weight: bold;
            content: '¿';
        }

        .question::after{
            font-weight: bold;
            content: '?';
        }
    </style>
</head>
<body>
    <p class="question">Te gusta el verano</p>
</body>
</html>
```

Resultat

¿Te gusta el verano?

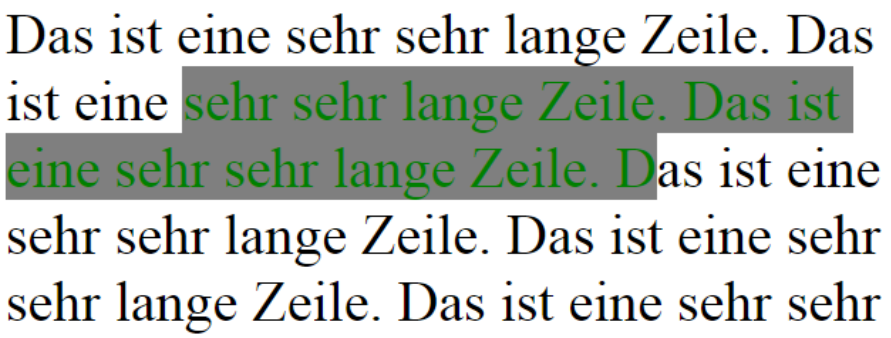
Debugger

```
<p class="question">
  ::before
  "Te gusta el verano"
  ::after
</p>
```

::selection

- **::selection ermöglicht es die Vorder- und Hintergrundfarbe der Selektion anpassen**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        p::selection {
            background: gray;
            color : green
        }
    </style>
</head>
<body>
    <p>Das ist eine sehr sehr lange Zeile… </p>
</body>
</html>
```



- Wichtig: Das Verhalten ist Browser spezifisch.

PSEUDO KLASSEN

Pseudoklassen

- **Pseudoklassen sind Selektoren, welche Eigenschaften / Zustände von Elemente Berücksichtigen**
- **Liste aller Pseudoklassen: https://wiki.selfhtml.org/wiki/CSS/Selektoren/Pseudoklasse**
- **Es wird unterschieden**
  - strukturelle Pseudoklassen
    - :first-child, :nth-child(), :empty,…
  - dynamische Pseudoklassen
    - :hover, :active, :focus, :visited,…
  - diverse
    - :lang(), :not(), :matches()
- **Die Pseudoklassen sind Teil der Übungen / Selbststudium.**

n-Variable

- **Einige Pseudoklassen können mit einem «n» umgehen z.B. :nth-child( )**
  - :nth-child(1)
    - Selektiert das erste Element (index 1)
  - :nth-child(2)
    - Selektiert das zweite Element (index 2)
  - :nth-child(2n)
    - Selektiert alle Elemente mit index % 2 == 0
  - :nth-child(2n+1)
    - Selektiert alle Elemente mit index + 1 % 2 == 0
  - :nth-child(3n)
    - Selektiert alle Elemente mit index % 3 == 0
  - :nth-child(odd) bzw. :nth-child(even)
    - Selektiert alle geraden bzw. ungeraden Elemente
- **Index startet bei 1!**

:nth-child() Tabelle Formatieren

- **Abwechselnde Hintergrundfarbe verbessert die Lesbarkeit einer Tabelle**

```
table tr:nth-child(2n){
    background: gray;
}
```

A	B	X
A	1	X
C	2	X
D	3	X
E	4	X
F	5	X
G	6	X

SELEKTOR HINTERGRUND INFOS

CSS Interpretation

- **Browser interpretiert ein CSS-Selektor von rechts nach links**

```
#id ul li a { color: red; }
```

- **Alle «a» Elemente werden selektiert**
  - Für jedes Element wird nach einem Parent-«li»-Tag gesucht.
    - Für jedes Element wird nach einem Parent-«ul»-Tag gesucht.
      - Für jedes Element wird nach einem Parent mit der #id gesucht.
        - Style wird angewendet
  - Falls die Suche fehlschlägt – wird die Regel nicht angewandt

- **Wie wird folgender Ausdruck interpretiert?**

```
[href]{
    color: red;
}
```

CSS Performance

https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Writing_efficient_CSS

Wichtigste Punkte

- **Avoid the descendant selector**
  - A > B wenn möglich A B vorziehen
- **Don’t qualify ID rules with tag names or classes**
  - A > #ID ist unnötige da #ID eindeutig ist
- **Rely on inheritance**

```
/*bad*/
.navigation li a { font-family: Georgia, Serif; }
/*besser*/
.navigation { font-family: Georgia, Serif; }
```

- **Avoid universal rules**
  - Auch folgendes ist eine Universal Regel:

```
[href]{ }
```

QUELLEN / ZUM NACHSCHLAGEN

Informationsquellen / Bildquellen

- https://wiki.selfhtml.org/
- https://developer.mozilla.org/de/