

Prof. Dr.-Ing. Georg J. Schneider

Multimedia und Medieninformatik
Fachbereich Informatik
Hochschule Trier

Literatur

CSS Spezifikation

<https://www.w3.org/Style/CSS/>

Unter „CSS Online Informationen“

[https://www.w3.org/Style/CSS/
learning#translations](https://www.w3.org/Style/CSS/learning#translations)

html-handbuch

<http://webkompetenz.wikidot.com/>

Michael Kipp

<http://michaelkipp.de/web/index.html>

W3Schools

<https://www.w3schools.com>

HTML5 & CSS3
BMU Verlag



CSS

Cascading Style Sheets

- Formatierungssprache für Dokumente, z.B. für (X)HTML-Dokumente

Lernziele

- Konzepte von CSS kennen und anwenden können
- Einfache CSS-Stilvorlagen erstellen können

Was ist CSS?

- Ermöglicht es, Formatierungen (z.B. Schriftarten, Farbe, Abstände) in strukturierten Dokumenten durchzuführen
- Wird insbesondere zusammen mit HTML/XHTML und XML eingesetzt
- Gestaltung speziell an das Ausgabemedium anpassbar
- Vom W3C (World Wide Web Consortium) genormt
- Mittel, um die Darstellung vom Inhalt zu trennen: Inhalte in (X)HTML und deren Gestaltung mit CSS realisieren

CSS-Versionen

1996: CSS 1

1998: CSS 2

2011: CSS 2.1: verabschiedet 7 Juni 2011, geändert 12. April 2016

zurzeit: CSS 3 (<https://www.w3.org/Style/CSS/Overview.de.html>)

CSS

Kommentare

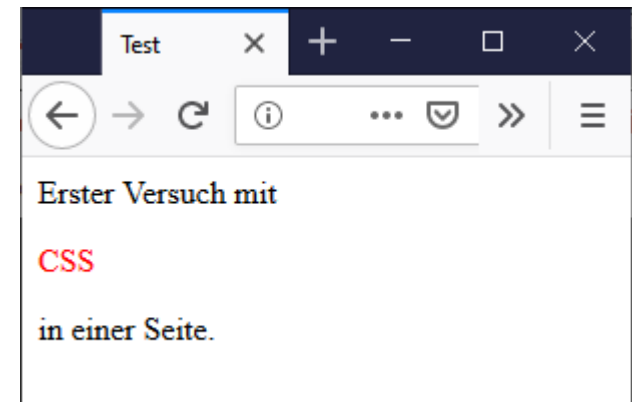
```
/*  
Dies ist ein Kommentar  
*/
```

```
/* Basis-Style */  
h1 { color:white;  
      background-color:black; /*silver*/  
}
```

CSS

Beispiel

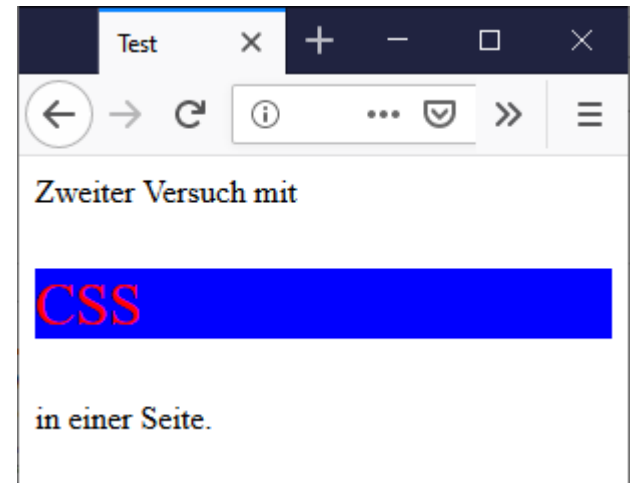
```
<!DOCTYPE html>
<html lang="de">
  <head>
    <title>Test</title>
  </head>
  <body>
    Erster Versuch mit
    <p style="color:red">CSS </p>
    in einer Seite.
  </body>
</html>
```



CSS

Beispiel

```
<!DOCTYPE html>
<html lang="de">
<head>
<title>Test</title>
</head>
<body>
Zweiter Versuch mit
<p style="color:red; font-size:30px; background-color:blue">CSS
</p>
in einer Seite.
</body>
</html>
```



CSS

Nachteil der Vorgehensweise

- Zwar Trennung von Struktur und Layout, aber
- Beide Konzepte in einer Datei vermischt.
- Jede Änderung muss einzeln in das Element geschrieben werden
- Keine allgemeinen Vorgaben möglich, wie:
Alle Überschriften 30 Pixel und in roter Farbe

CSS

Terminologie

- CSS-Deklaration
- CSS-Eigenschaft:Wert;
- CSS-Stilvorlage (Stilregel, Formatdefinition)
- CSS-Eigenschaft an (X)HTML-Element koppeln
- Mehrere CSS-Eigenschaften durch Semikolon trennen
- Alle Stilvorlagen zusammen bilden ein CSS-Stylesheet

Einfache CSS-Stilvorlagen

```
Selektor1,  
Selektor2 { CSS-Eigenschaft1:Wert1;  
            CSS-Eigenschaft2:Wert2;  
            }
```

```
body { color:red; background-color:white; }
```

The diagram illustrates the components of a CSS declaration block. A dashed blue box encloses the entire line of code. A label 'Selektor' is positioned to the left of the 'body' selector, with a vertical dashed line connecting it to 'body'. A label 'Deklaration' is placed below the opening curly brace, with a horizontal dashed line connecting it to the brace. A label 'CSS-Eigenschaft Wert' is placed below the first declaration, with vertical dashed lines connecting it to 'color:red;'. A label 'Deklarationsblock' is centered below the entire dashed box, with a horizontal dashed line connecting it to the box's boundaries.

Universalselektor

```
* { color:red; }
```

Einfache CSS Stilvorlagen

In HTML Datei im **head**

```
<!DOCTYPE html>
<html>
<head>
<title>simpleCSS</title>
<style>
  body { background-color:silver;
        color:navy; }
  h1 { color:red; }
</style>
</head>

<body>
<h1>Herzlich willkommen</h1>
<p>Hier entsteht eine neue Website</p>
</body>
```

Herzlich willkommen

Hier entsteht eine neue Website

CSS Stilvorlagen

Externe Datei

Stylesheet-Datei mit Endung .css

Referenz im HTML-Dokument:

```
<link rel="stylesheet" type="text/css" href="stylesheet.css">
```

- Attribut `type="text/css"`: Stylesheet im CSS-Format
- Attribut `href`: beliebige URL

CSS validieren

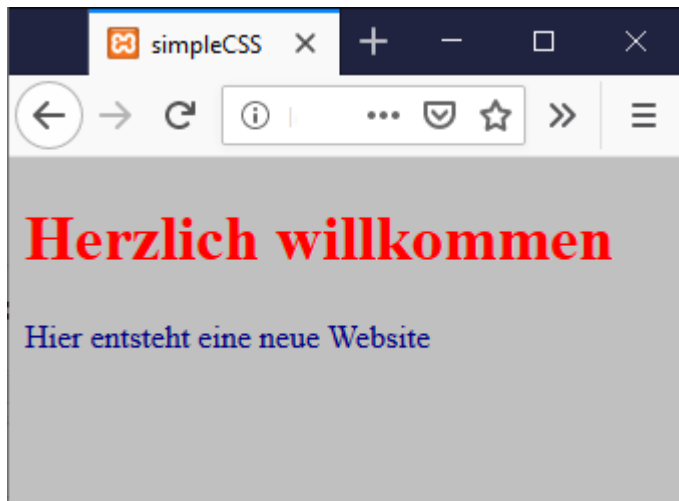
Validierungsservice: <http://jigsaw.w3.org/css-validator>

CSS Stilvorlagen

Beispiel

Datei: stylesheet.css

```
body {background-color:silver;  
      color:navy;}  
h1 {color:red;}
```



HTML-Datei mit Referenz auf Stylesheet

```
<!DOCTYPE html>  
<html>  
<head>  
<title>simpleCSS</title>  
<link rel="stylesheet"  
      type="text/css"  
      href="stylesheet.css">  
</head>  
<body>  
<h1>Herzlich willkommen</h1>  
<p>Hier entsteht eine neue  
Website</p>  
</body>
```

CSS Stilvorlagen

Beispiel

Mehrere Elemente mit demselben Stil

```
h1, h2 { color:red; }
```

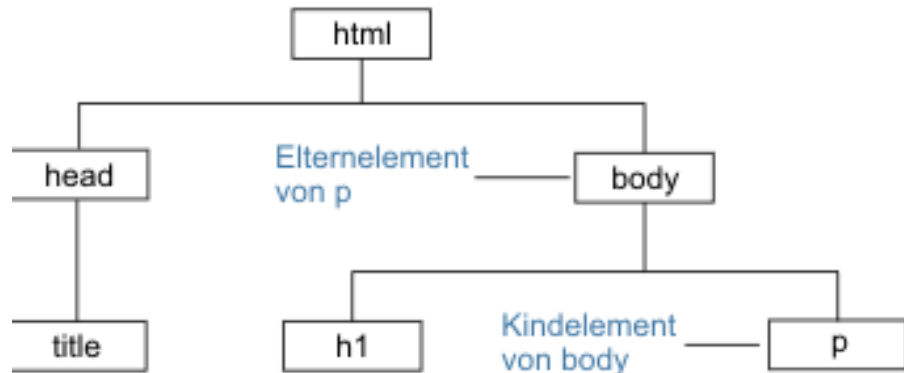


```
<!DOCTYPE html>
<html>
<head>
<title>simpleCSS</title>
<link rel="stylesheet"
type="text/css"
      href="stylesheet.css">
</head>
<body>
<h1>Herzlich willkommen</h1>
<h2>Hier entsteht eine neue
Website</h2>
<p> zum Thema CSS </p>
</body>
```

CSS

Vererbung

- Elternelemente vererben ihre Stilvorlagen im Allgemeinen an ihre Kindelemente
- Beispiel: Gilt `body {color: navy;}`, dann sind auch alle darin enthaltenen Absätze in der Schriftfarbe `navy`
- Wichtig für wartbare Stylesheets



CSS

Vererbung Konflikte 1

Kaskade

- Stilvorlagen können zueinander in Konflikt stehen
- Konflikte werden durch Kaskade aufgelöst
- Kaskade weist jeder Stilregel ein Gewicht zu
- Diejenige mit dem höchsten Gewicht »gewinnt«

CSS

Vererbung Konflikte 2

Aufsteigende Hierarchie von Style Sheets mit Overriding

- `! important` zeichnet wichtige Regeln aus und erhöht die Priorität
- Herkunft der Regeln: (Autor <- Leser <- User Agent)
- Selektoren: Berechnung der Genauigkeit
- Reihenfolge der Spezifikation: die zuletzt spezifizierte Angabe gewinnt – und: Inline Styleangabe (`<p style="...">`) ist wichtiger als ein verlinktes Stylesheet

CSS

Selektoren

Elemente

- Element-Selektoren verwenden den Namen des Elementes
z.B.: `p` oder `h1`
- Sollen mehrere Elemente gleich dargestellt werden, werden die Selektoren mit Kommas abgetrennt
z.B.: `p, h1`

CSS

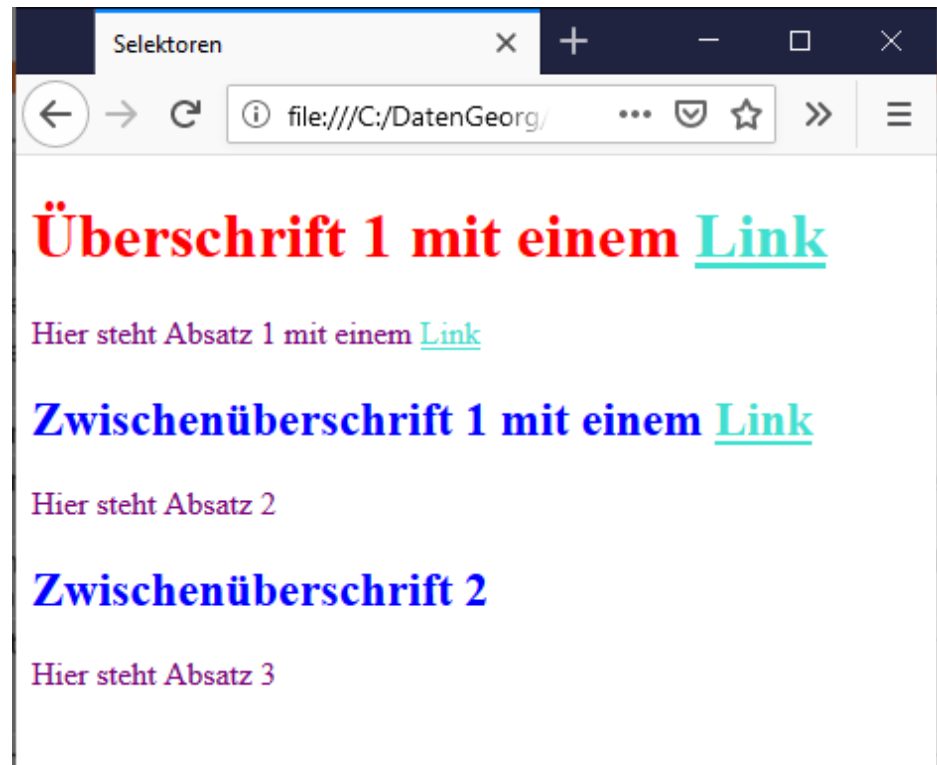
Selektoren

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Selektoren
  </title>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="styleSelektoren1.css">
</head>
<body>
  <h1>Überschrift 1 mit einem <a href="selektoren1.html">Link</a></h1>
  <p>Hier steht Absatz 1 mit einem <a href="selektoren1.html">Link</a></p>
  <h2>Zwischenüberschrift 1 mit einem <a href="selektoren1.html">
    Link</a></h2>
  <p>Hier steht Absatz 2</p>
  <h2>Zwischenüberschrift 2</h2>
  <p>Hier steht Absatz 3</p>
</body>
</html>
```

CSS

Selektoren

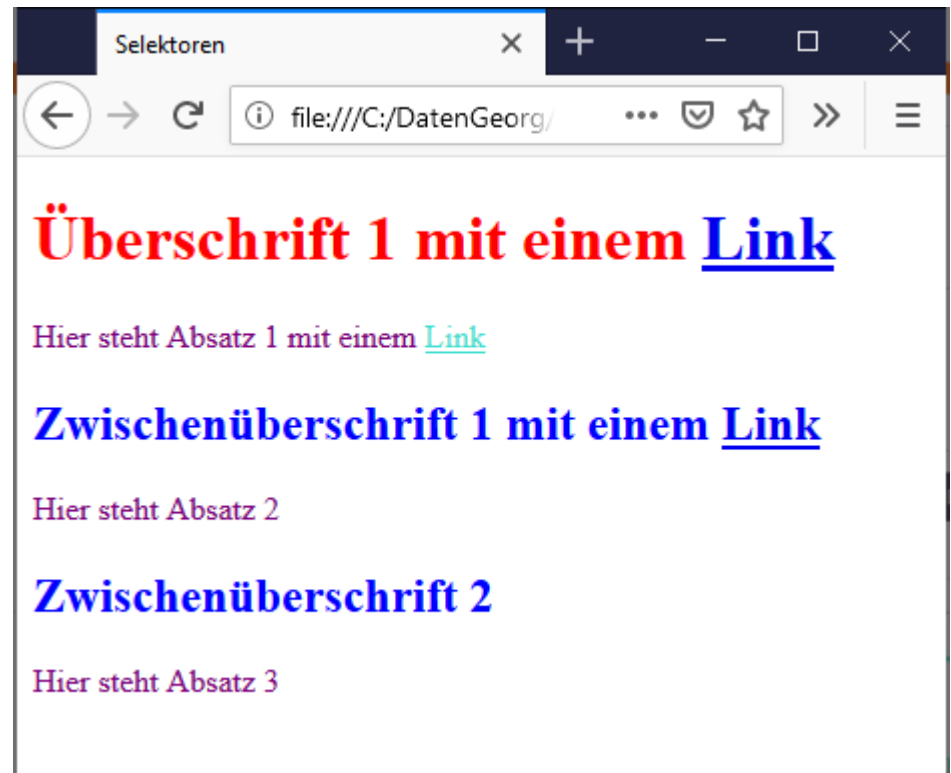
```
h1 {  
  color: red;  
}  
h2 {  
  color: blue;  
}  
p, div {  
  color: purple;  
}  
a {  
  color: turquoise;  
}
```



CSS

Nachfahren Selektor

```
h1 {  
  color: red;  
}  
h2 {  
  color: blue;  
}  
p, div {  
  color: purple;  
}  
p a {  
  color: turquoise;  
}
```



Alle Elemente `a`, die innerhalb eines Paragraphen (`p`) liegen, werden angesprochen.

CSS

Selektoren

Klassen

- Klassen werden in der HTML Seite mit dem Attribut `class` definiert
z.B.: `<p class="meineKlasse">`
- In einem Stylesheet werden sie mit einem der Klasse vorangestellten Punkt angesprochen
z.B.: `.meineKlasse`

CSS

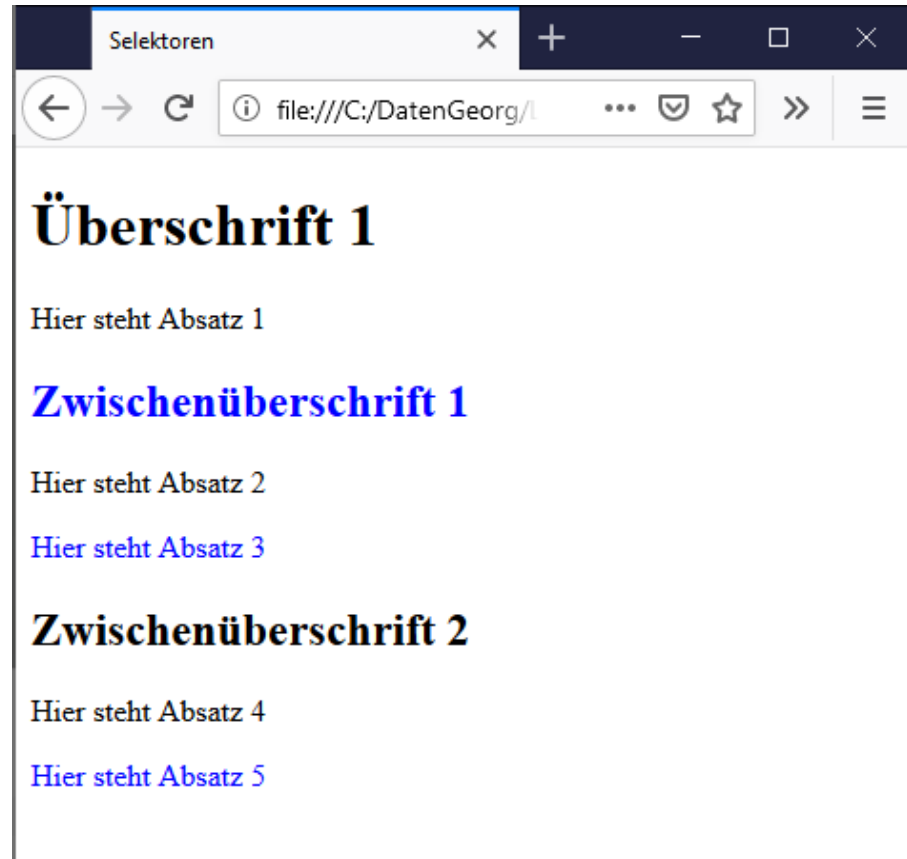
Selektoren

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Selektoren
  </title>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="styleSelektoren3.css">
</head>
<body>
  <h1>Überschrift 1</h1>
  <p>Hier steht Absatz 1</p>
  <h2 class="blau">Zwischenüberschrift 1</h2>
  <p>Hier steht Absatz 2</p>
  <p class="blau">Hier steht Absatz 3</p>
  <h2>Zwischenüberschrift 2</h2>
  <p>Hier steht Absatz 4</p>
  <p class="blau">Hier steht Absatz 5</p>
</body>
</html>
```

CSS

Selektoren

```
.blau {  
color: blue;  
}
```



CSS

Selektoren

Klassen

- In einem Stylesheet können mehrere Klassen definiert werden
z.B.: `.meineKlasse {color:blue;}`
`.deineKlasse {font-size:20px;}`
- Ein Element kann mehreren Klassen zugeordnet werden. Diese Klassen werden im Attribut `class` hintereinander geschrieben
z.B.: `<p class="meineKlasse deineKlasse">`

CSS

Selektoren

IDs

- IDs werden in der HTML Seite mit dem Attribut `id` definiert
z.B.: `<p id="absatz1">`
- IDs zeichnen ein einzelnes Element aus und müssen daher innerhalb der Seite eindeutig sein.
- In einem Stylesheet werden sie mit einer der ID vorangestellten Raute angesprochen
z.B.: `#absatz1`

CSS

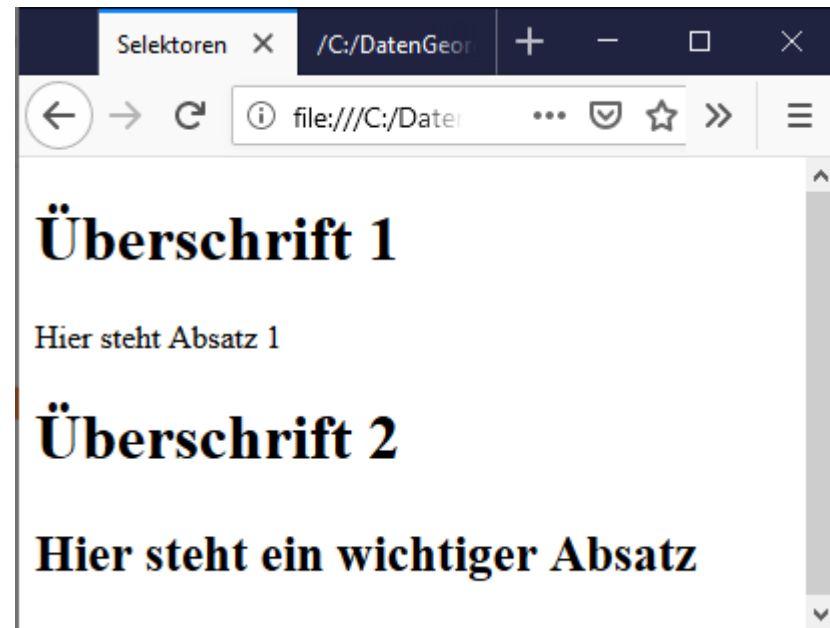
Selektoren

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Selektoren
  </title>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="styleSelektoren4.css">
</head>
<body>
  <h1>Überschrift 1</h1>
  <p>Hier steht Absatz 1</p>
  <h1>Überschrift 2</h1>
  <p id="absatz1">Hier steht ein wichtiger Absatz </p>
</body>
</html>
```

CSS

Selektoren

```
#absatz1 {  
  font-size: 25px;  
  font-weight: bold;  
}
```



CSS

Selektoren

Klassen und Elemente miteinander verbinden

- Möchte man, dass nur für bestimmte Elemente einer Klasse, die Eigenschaften einer Klasse abgeändert werden, so kombiniert man Element und Klassen-Selektor

z.B.: `<p class="blau">`
`<h1 class="blau">`

- Es sollen jetzt nur die `p`-Elemente zusätzlich mit größerer Schrift versehen werden, die zur Klasse `blau` gehören.
- In einem Stylesheet werden sie dann mit `Elementname.Klassenname` **selektiert**
z.B.: `p.blau`

CSS

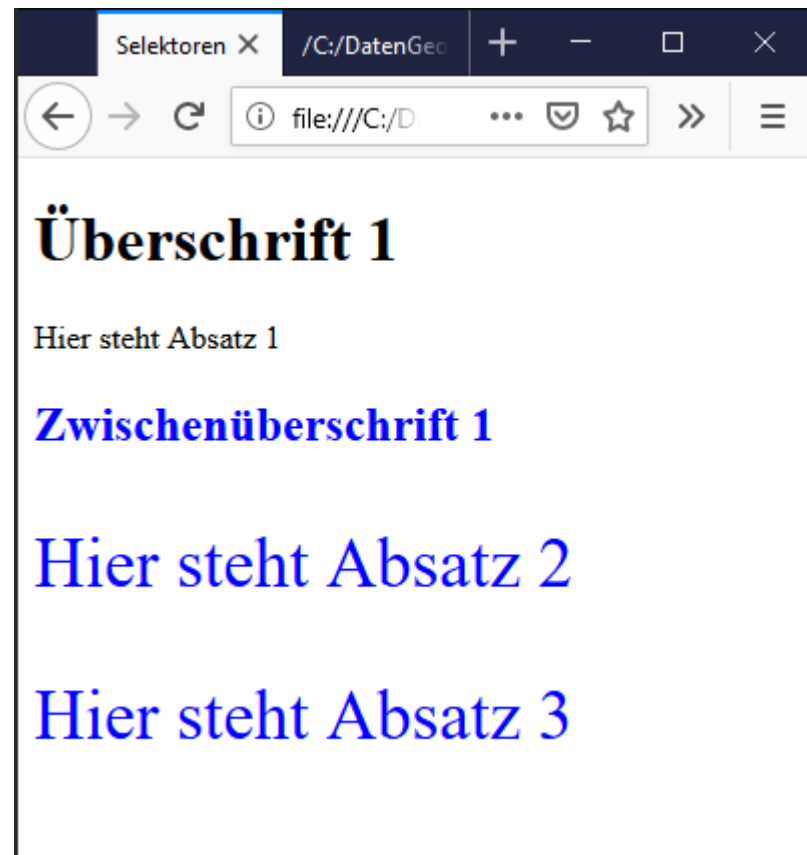
Selektoren

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Selektoren
  </title>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="styleSelektoren5.css">
</head>
<body>
  <h1>Überschrift 1</h1>
  <p>Hier steht Absatz 1</p>
  <h2 class="blau">Zwischenüberschrift 1</h2>
  <p class="blau">Hier steht Absatz 2</p>
  <p class="blau">Hier steht Absatz 3</p>
</body>
</html>
```

CSS

Selektoren

```
.blau {  
  color:blue;  
}  
p.blau {  
  font-size:35px;  
}
```



CSS

Selektoren

Pseudoklassen

- Pseudoklassen sind keine Klassen im eigentlichen Sinne, sondern beschreiben zusätzliche Eigenschaften von Elementen, wie das strukturelle oder dynamische Verhalten von Elementen oder Besonderheiten, wie z.B. die Sprache
- In einem Stylesheet wird z.B. die Eigenschaft zur Darstellung eines bereits besuchten Links wie folgt dargestellt: `a:visited`
Elemente, über denen sich der Mauszeiger befindet mit
z.B.: `p:hover`

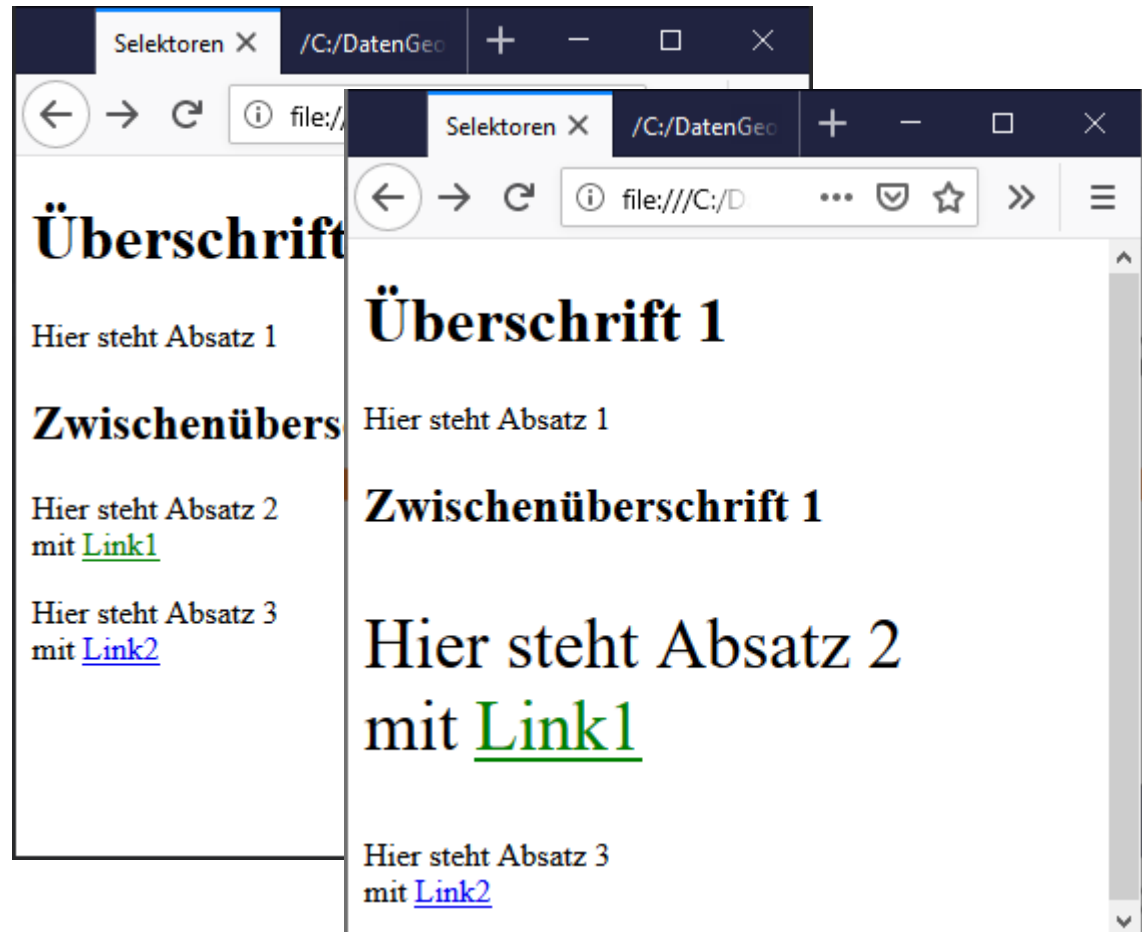
CSS

Selektoren

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Selektoren
  </title>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="styleSelektoren6.css">
</head>
<body>
  <h1>Überschrift 1</h1>
  <p>Hier steht Absatz 1</p>
  <h2>Zwischenüberschrift 1</h2>
  <p>Hier steht Absatz 2<br>mit <a href="ziel1.html">Link1</a></p>
  <p>Hier steht Absatz 3<br>mit <a href="ziel2.html">Link2</a></p>
</body>
</html>
```

CSS Selektoren

```
a:visited {  
    color:green;  
}  
p:hover {  
    font-size:35px;  
}
```



CSS

Selektoren

Einfügen von neuem Inhalt:

::before
::after

Beispiel:

```
<head>
<style type="text/css">
span.ironie::before { content:" ;-) "}
span.ironie::after  { content:" ;-) "}
</style>
</head><body>
<div> <span class="ironie">Ab heute wird alles besser!</span>
</div>
```



CSS Selektoren

| Muster | Beschreibung | Beschrieben in Abschnitt |
|--------------------------------|--|--------------------------------|
| * | Stimmt mit jedem Element überein. | Universeller Selektor |
| E | Stimmt mit jedem E-Element überein (d.h. ein Element des Typs E). | Typselektoren |
| E F | Stimmt mit jedem F-Element überein, das ein Nachfahre eines E-Elements ist. | Selektoren für Nachfahren |
| E > F | Stimmt mit allen F-Elementen überein, die Kindelemente eines Elements E sind. | Kindselektoren |
| E:first-child | Stimmt mit Element E überein, wenn E das <i>erste</i> Kindelement des übergeordneten Elements ist. | Die Pseudoklasse :first-child |
| E:link E:visited | Stimmt mit dem Element E überein, wenn E der Quellanker eines Hyperlinks ist, dessen Ziel noch nicht besucht wurde (:link), oder dessen Ziel bereits besucht wurde (:visited). | Die Link-Pseudoklassen |
| E:active E:hover E:focus | Stimmt während bestimmter Benutzeraktionen mit E überein. | Die Dynamic-Pseudoklassen |
| E:lang(c) | Stimmt mit einem Element des Typs E überein, wenn dieses sich in der (menschlichen) Sprache c befindet (die Dokumentsprache gibt an, wie die Sprache ermittelt wird). | Die Sprach-Pseudo-Klasse :lang |
| E + F | Stimmt mit jedem F-Element überein, dem unmittelbar ein Element E vorausgeht. | Benachbarte Selektoren |
| E[foo] | Stimmt mit jedem E-Element überein, dessen Attribut „foo“ gesetzt ist (ganz gleich, welchen Wert es hat). | Attribut-Selektoren |
| E[foo="warning"] | Stimmt mit jedem E-Element überein, dessen Attribut „foo“ genau den Wert von „warning“ hat. | Attribut-Selektoren |
| E[foo~="warning"] | Stimmt mit jedem E-Element überein, dessen Attribut „foo“ eine Liste von durch Leerzeichen voneinander getrennten Werten enthält, und einer dieser Werte gleich „warning“ ist. | Attribut-Selektoren |
| E[lang]="en"] | Stimmt mit jedem E-Element überein, dessen Attribut „lang“ eine Liste mit durch Trennstriche voneinander getrennten Werten enthält, die (von links) mit „en“ beginnen. | Attribut-Selektoren |
| DIV.warning | Nur HTML. Dasselbe wie DIV[class~="warning"]. | Klassen-Selektoren |
| E#myid | Stimmt mit jedem E-Element überein, dessen ID gleich „myid“ ist. | ID-Selektoren |

CSS Selektoren

- Komplette Liste der Selektoren unter:
 - <https://www.w3.org/TR/selectors-3/>
und
 - <https://www.w3.org/TR/selectors-4/>
- Browserunterstützung: <https://caniuse.com/#search=selectors>
- Test: <http://tools.css3.info/selectors-test/test.html>

CSS Selektoren

Weitere Beispiele

```
div > p {color:blue;}
```

direkter Nachfolger

```
div * b {color:violet;}
```

alle Nachfolger ab dem Übernächsten

```
div p *[href]
```

Element, bei dem das Attribut href gesetzt ist, und das sich in einem p befindet, das sich wiederum in einem div befindet

CSS Selektoren

Weitere Beispiele

```
p { font-weight:bold; font-family:Tahoma,sans-serif;
    font-size:14pt }
p[title] { color:red }
p[title="center"] { color:blue; text-align:left }
div[title~="Berlin"] { background-color:#FFFF00 }
*[lang|"en"] { background-color:#FF0000;
               color:#FFFFFF }
```

```
<p title="right">Das ist ein Textabsatz, rechts ausgerichtet.</p>
<p title="center">Das ist ein Textabsatz, zentriert? Wirklich zentriert?</p>
...
<div title="Es folgen Infos zu Berlin">Eine Menge Inhalt zu Berlin</div>
...
Text <span lang="en">about</span> einen
<span lang="en-US">English man in New York</span>
```

CSS Selektoren

Weitere Beispiele

Das ist ein Textabsatz, rechts ausgerichtet.

Das ist ein Textabsatz, zentriert? Wirklich zentriert?

Eine Menge Inhalt zu Berlin

Text **about** einen **English man in New York**

```
<p title="right">Das ist ein Textabsatz, rechts ausgerichtet.</p>
<p title="center">Das ist ein Textabsatz, zentriert? Wirklich zentriert?</p>
...
<div title="Es folgen Infos zu Berlin">Eine Menge Inhalt zu Berlin</div>
...
Text <span lang="en">about</span> einen
<span lang="en-US">English man in New York</span>
```


CSS

Formatierung

Schriftgröße

Beispiel: `p.blau {font-size:35px;}`

- Verschiedene Einheiten für Größen im Allgemeinen:
 - cm (Zentimeter)
 - mm (Millimeter)
 - in (Zoll)
 - pt (Punkt)
 - em (relative Schriftgröße, in etwa Größe des Buchstaben „m“)
 - rem (relative Schriftgröße, aber in Bezug auf das Root-Element)
 - pc (Pica)
 - % (bezieht sich auf die Größe des übergeordneten Elementes)

CSS

Formatierung

Schriftgröße

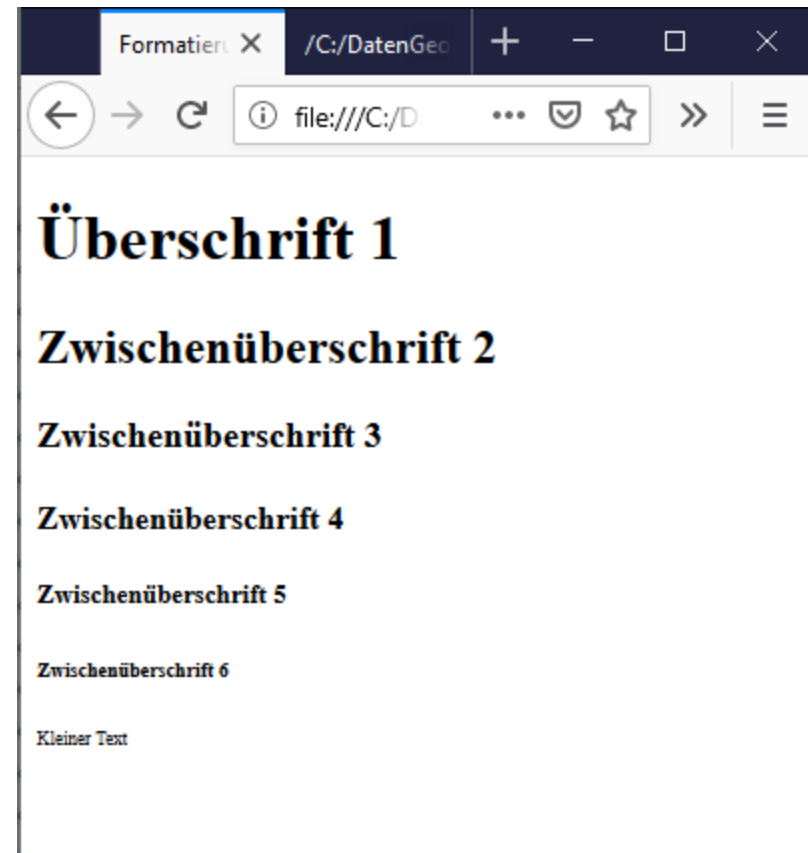
Symbolische Bezeichnung der Größe

- `xx-small`
- `x-small`
- `small`
- `medium`
- `large`
- `x-large`
- `xx-large`

CSS

Schriftgrößen

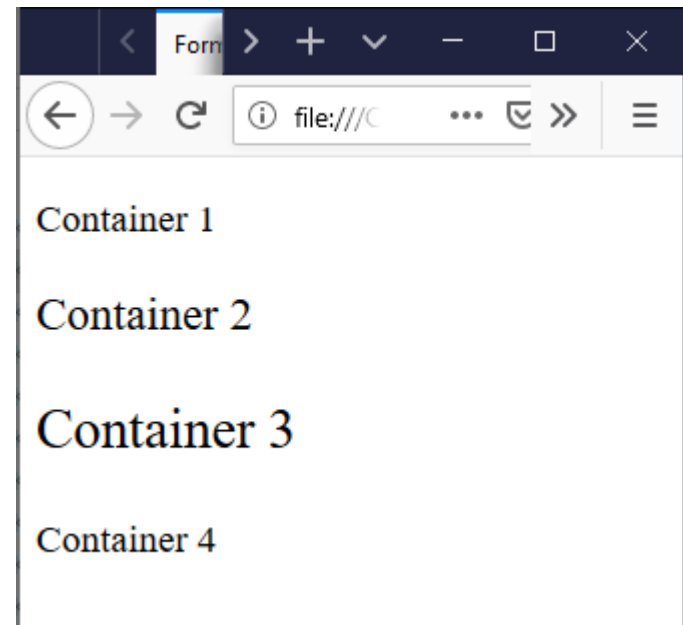
```
...  
<style>  
  h1 { font-size:xx-large;}  
  h2 { font-size:x-large;}  
  h3 { font-size:large;}  
  h4 { font-size:medium;}  
  h5 { font-size:small;}  
  h6 { font-size:x-small;}  
  p { font-size:xx-small;}  
</style>  
</head>  
<body>  
  <h1>Überschrift 1</h1>  
  <h2>Zwischenüberschrift 2</h2>  
  <h3>Zwischenüberschrift 3</h3>  
  <h4>Zwischenüberschrift 4</h4>  
  <h5>Zwischenüberschrift 5</h5>  
  <h6>Zwischenüberschrift 6</h6>  
  <p>Kleiner Text</p>  
</body>  
...
```



CSS

Schriftgrößen

```
...  
<style>  
  .groesser { font-size:120%;}  
</style>  
</head>  
<body>  
  <div class="groesser">  
    <p>Container 1</p>  
    <div class="groesser">  
      <p>Container 2</p>  
      <div class="groesser">  
        <p>Container 3</p>  
      </div>  
    </div>  
  </div>  
  <div class="groesser">  
    <p>Container 4</p>  
  </div>  
</body>  
...
```



CSS

Schriftarten

Serifenschriften

- "Times New Roman"
- Times
- Georgia

Serifenlose Schriften

- Verdana
- Arial
- Helvetica
- Geneva

Monospace-Schriften

- "Courier New"
- Courier

CSS

Mit Schriften gestalten

Schriftarten

- Es können mehrere Schriftarten mit der Eigenschaft `font-family` zugewiesen werden.
- Beginnend beim ersten Eintrag wird die Schriftart gewählt, die auf dem Rechner installiert ist.
- Sollte keine passende Schriftart installiert sein, sollte als letzter Eintrag eine Schriftartenfamilie folgen.

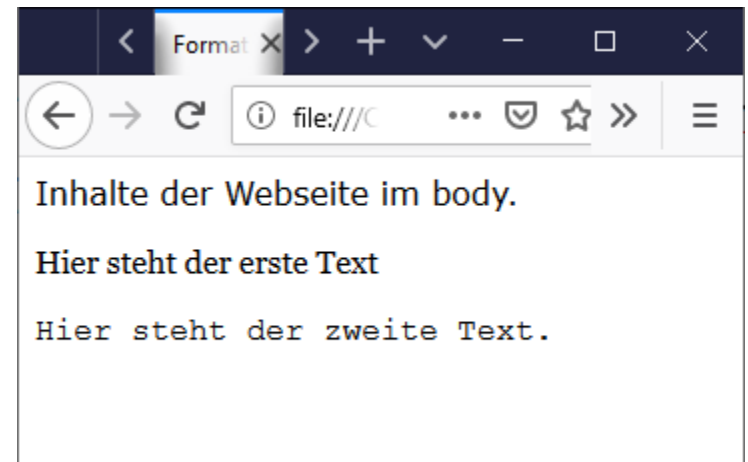
Beispiele

```
body { font-family:Verdana, Arial, Helvetica, sans-serif; }  
p.text { font-family:Georgia, serif; }  
p.program { font-family:"Courier New", Courier, mono; }
```

CSS

Schriftarten

```
...
<head>
  <title>
    Formatierung
  </title>
  <meta charset="UTF-8">
  <style>
    body {      font-family:Verdana, Arial, Helvetica, sans-serif; }
    p.text {    font-family:Georgia, serif; }
    p.program {font-family:"Courier New", Courier, mono; }
  </style>
</head>
<body>
  Inhalte der Webseite im body.
  <p class="text">
    Hier steht der erste Text
  <p class="program">
    Hier steht der zweite Text.
</body>
...
```



CSS

Eigenschaften von Schriften

Schriftschnitt

- Ähnlich wie bei den Elementen `strong` oder `bold` können Schriften mit Hilfe von Style Angaben stärker hervorgehoben werden. Es existieren die Angaben `lighter`, `normal`, `bold`, `bolder`
- Ebenso existiert das Gegenstück zum Element `i` für kursive Darstellung. Es existieren die Angaben `italic`, `normal` und `oblique`

Beispiele

`p { font-weight:bold; }` definiert Fettschrift für einen Absatz

`p { font-style:italic; }` definiert Kursivschrift für einen Absatz

CSS

Weitere Eigenschaften von Schriften

CSS-Kurzform

```
p { font-family:Verdana, sans-serif;  
    font-size:medium;  
    font-weight:bold;  
    font-style:italic; }
```

identisch mit

```
p { font:italic bold medium Verdana, sans-serif; }
```

Werte für `font-size` und `font-family` müssen genau in dieser Reihenfolge und als letzte Werte im Deklarationsblock angegeben werden.

CSS

Mit Schriften gestalten

Schriftarten aus dem Web

- Es können Schriftarten aus dem Web verwendet werden. Diese Schriftarten werden dann beim Laden der Webseite dazu geladen und dienen zur Darstellung des Textes.

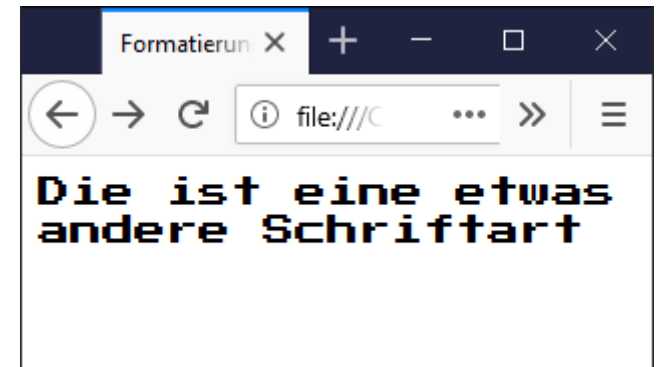
Beispiele

- <https://fonts.google.com>
- ```
<link
href="https://fonts.googleapis.com/css?family=Press+Start
+2P&display=swap" rel="stylesheet">
```
- ```
font-family: 'Press Start 2P', cursive;
```

CSS

Externe Schriftarten

```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Formatierung
  </title>
  <meta charset="UTF-8">
  <link
href="https://fonts.googleapis.com/css?family=Press+Start+2P&display=swap" rel="stylesheet">
  <style>
    body {font-family: 'Press Start 2P', cursive;}
  </style>
</head>
<body>
  Die ist eine etwas andere Schriftart
</body>
</html>
```



CSS

Eigenschaften von Schriften

Schriftfarbe

- Die Schriftfarbe kann mit der Eigenschaft `color` verändert werden.
- Es können dabei benannte Farben verwendet werden, wie `blue`, `green`, `red`, usw.
- Farben können aber auch im `rgb`- oder `hsl`-Modell spezifiziert werden
- Nähere Infos unter <https://developer.mozilla.org/de/docs/Web/CSS/Farben>)

Beispiele

`p { color:blue; }` definiert blaue Schriftfarbe für einen Absatz

CSS

Farben

Mit Farben gestalten

- Farbwerte: `#rrggbb`
- Farbnamen: `white`, `black` etc.

Schriftfarbe, Hintergrundfarbe

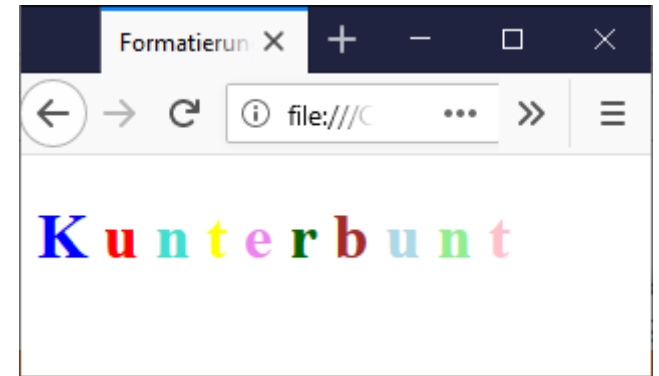
```
h3 { color:#c33b00; }  
body { background-color:silver; }
```

CSS

Schriftfarben

```
<style>
.blau      {color: blue;}
.rot       {color: red;}
.türkis    {color: turquoise;}
.gelb      {color: yellow;}
.violett   {color: violet;}
.dunkelgrün {color: darkgreen;}
.braun     {color: brown;}
.hellblau  {color: lightblue;}
.hellgrün  {color: lightgreen;}
.pink      {color: pink;}
</style>
```

```
<h1>
<span class = "blau">K</span>
<span class = "rot">u</span>
<span class = "türkis">n</span>
<span class = "gelb">t</span>
<span class = "violett">e</span>
<span class = "dunkelgrün">r</span>
<span class = "braun">b</span>
<span class = "hellblau">u</span>
<span class = "hellgrün">n</span>
<span class = "pink">t</span>
</h1>
```



CSS

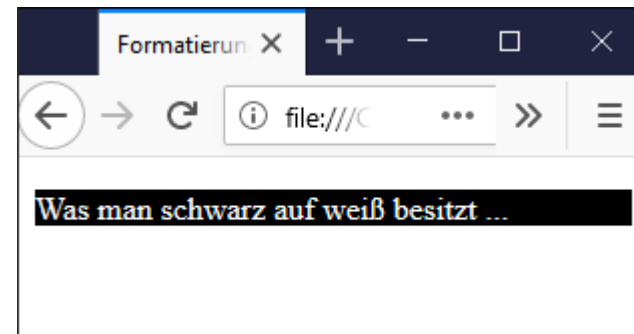
Mit Schriften gestalten

Schrifthintergrund

- Der Hintergrund einer Schrift kann durch die Eigenschaft `background-color` verändert werden

Beispiele

- Inverse Darstellung
- `color: white;`
- `background-color: black;`



CSS

Eigenschaften von Schriften

Verzierungen

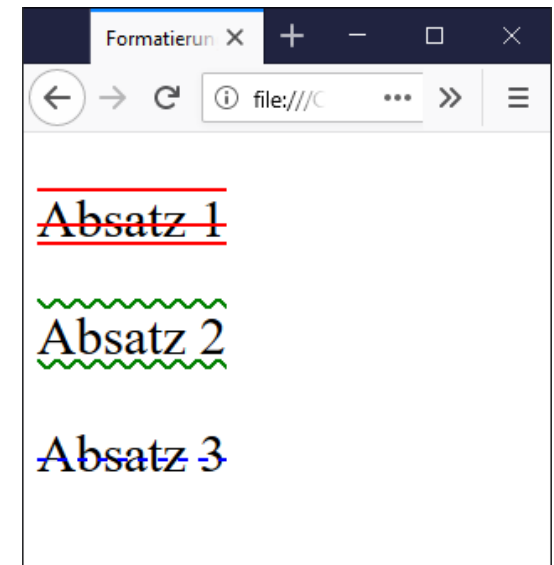
- Verzierungen können mit der Eigenschaft `text-decoration` hinzugefügt werden.
- Es können Werte, wie `overline`, `underline` oder `line-through` verwendet werden.
- Zusätzlich können weitere Eigenschaften angegeben werden, wie Farben oder Linieneigenschaften, z.B. `dashed`, `wavy`

CSS

Verzierungen

...

```
<style>
  p {font-size: 2em;}
.stil1 {text-decoration: underline line-through overline red;}
.stil2 {text-decoration: underline overline wavy green;}
.stil3 {text-decoration: line-through blue dashed;}
</style>
</head>
<body>
  <p class="stil1">Absatz 1</p>
  <p class="stil2">Absatz 2</p>
  <p class="stil3">Absatz 3</p>
</body>
```



CSS

Textausrichtung

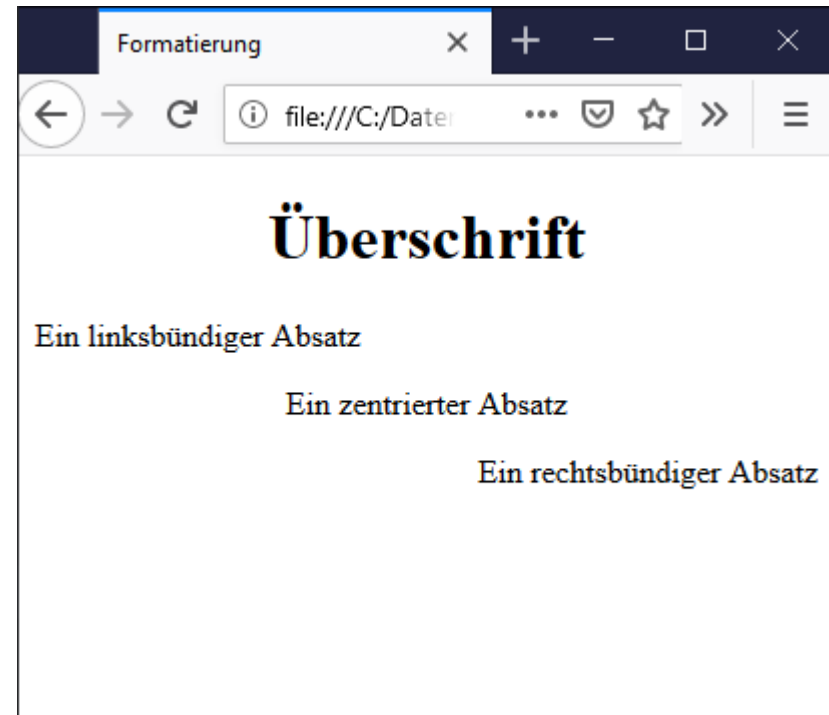
Bündigkeit

- Sätze können rechtebündig, linksbündig oder mittig positioniert werden.
- Die Eigenschaft `text-align` oder `text-indent` werden dazu verwendet.
- Für `text-align` können die Werte `left`, `right` oder `center` verwendet werden. Für `text-indent` können Größenangaben verwendet werden.

CSS

Textausrichtung

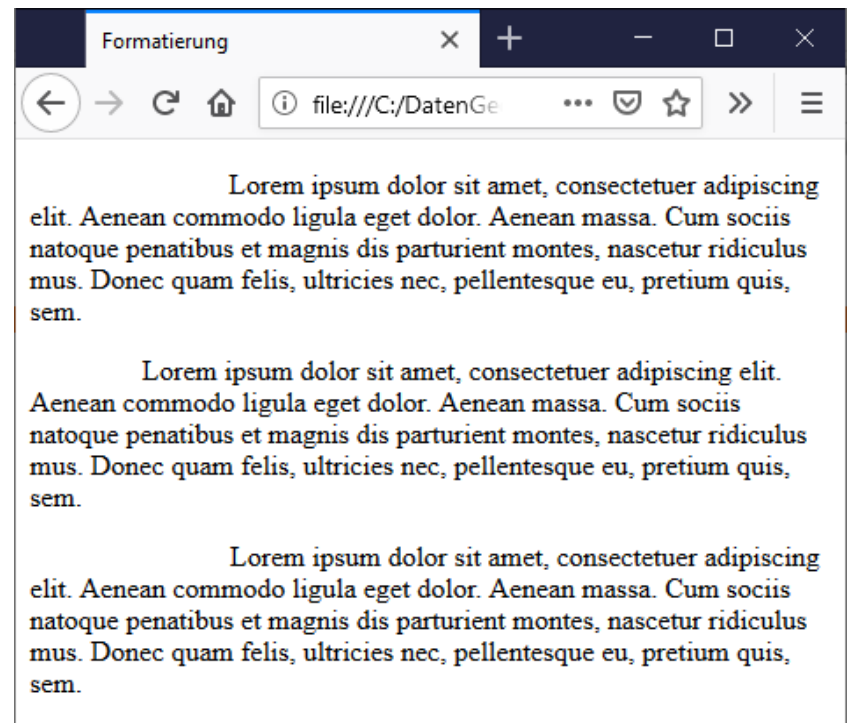
```
<!DOCTYPE html>
<html lang="de">
<head>
  <title>
    Formatierung
  </title>
  <meta charset="UTF-8">
  <style>
    .links      {text-align: left; }
    .zentriert {text-align: center }
    .rechts     {text-align: right; }
  </style>
</head>
<body>
  <h1 class="zentriert">Überschrift</h1>
  <p class="links">Ein linksbündiger Absatz</p>
  <p class="zentriert">Ein zentrierter Absatz</p>
  <p class="rechts">Ein rechtsbündiger Absatz</p>
</body>
</html>
```



CSS

Textausrichtung

```
#absatz1{  
text-indent: 3cm;  
}  
#absatz2{  
text-indent: 4em;  
}  
#absatz3{  
text-indent: 25%;  
}
```



CSS

Hintergründe

Farben und Bilder als Hintergründe

- Der Hintergrund kann durch die Eigenschaft `background-color` verändert werden.
- Durch die Eigenschaft `background-image` können Bilder als Hintergrund eingebunden werden.
Es sollte aber immer auch zusätzlich eine Farbe angegeben werden, die gewählt wird, falls das Bild nicht angezeigt werden kann.

CSS

Textausrichtung

```
...
<style>
  body {background-image: url("katze.jpg");
        /*background-repeat: no-repeat;*/
        background-color: azure;}
  h1 {background-color: darkturquoise;}
  p {background-color: darkslateblue;
     color: beige;}
</style>
</head>
<body>
<h1>Überschrift 1</h1>
<p>
Hier steht Absatz 1<br>
Dieser besteht aus mehreren Zeilen
</p>
</body>
```



CSS

Hintergründe

Bilder als Hintergründe

- Bilder werden sowohl in der Breite, als auch in der Länge wiederholt. Hiermit lassen sich Strukturen als Hintergrund erzeugen.
- Zur Eigenschaft `background-image` können weitere Informationen angegeben werden:
 - `no-repeat`
 - `repeat-x`
 - `repeat-y`
- Die Eigenschaft `background-position` dient zur Positionierung der Grafik. Beispiele sind
 - `background-position: center top;`
 - `background-position: 30px 60px;`

CSS

Textausrichtung

```
...  
body {  
background-image: url("katze.jpg");  
background-repeat: no-repeat;  
background-color: azure;  
background-position: center top;  
}  
...
```



CSS

Hintergründe

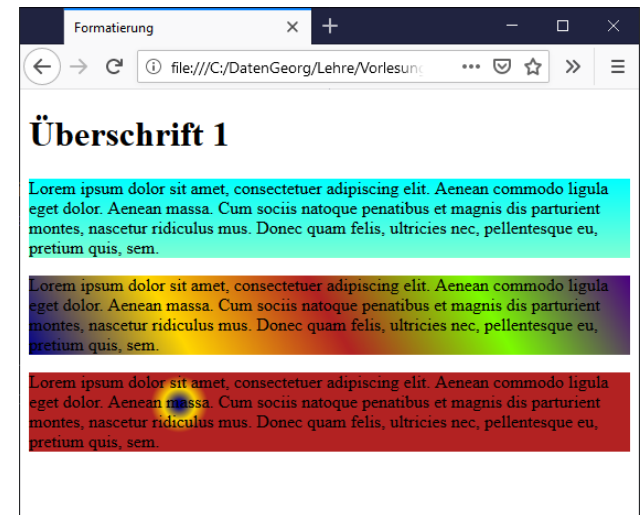
Farbverläufe

- Die Eigenschaft `background-image` kann zum Erzeugen von Farbverläufen benutzt werden:
 - `linear-gradient`
 - `radial-gradient`
 - `conic-gradient`
- Es können Übergänge zwischen zwei und auch mehreren Farben erzeugt werden. Beispiele sind:
 - `background-image: linear-gradient(aqua, aquamarine);`
 - `background-image: linear-gradient(60deg, darkblue, gold, firebrick, lawngreen, indigo);`
 - `background-image: radial-gradient(circle closest-side at 25% 40%, darkblue 10%, gold 50%, firebrick 90%);`

CSS

Textausrichtung

```
#p1{  
    background-image: linear-gradient(aqua, aquamarine);  
#p2{  
    background-image: linear-gradient(60deg, darkblue, gold,  
    firebrick, lawngreen, indigo);  
#p3{  
    background-image: radial-gradient(circle closest-side at 25% 40%,  
    darkblue 10%, gold 50%, firebrick 90%);}
```



CSS

Größe, Abstand, Positionen

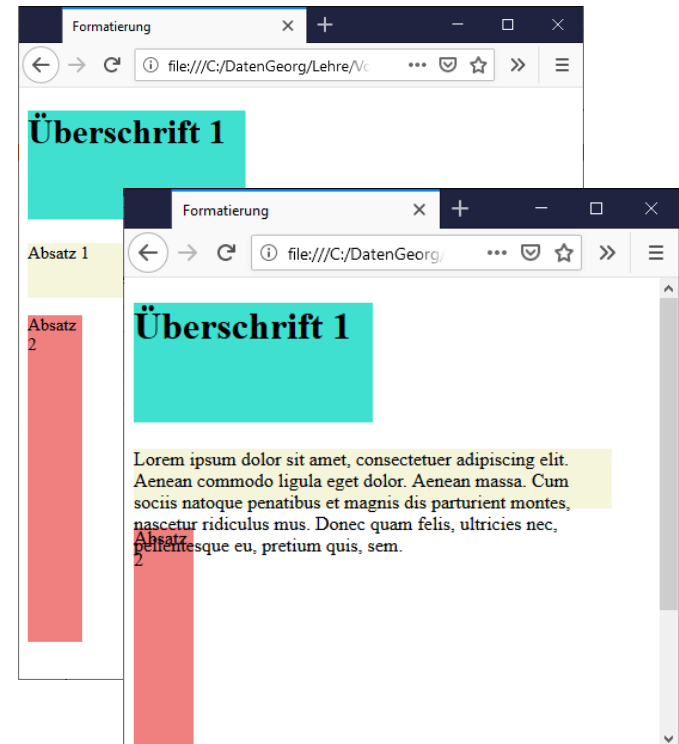
Größe von Elementen

- Die Eigenschaften `width` und `height` dienen dazu, die Höhe und Breite von Elementen zu spezifizieren.
- Es können absolute oder relative Einheiten zum Einsatz kommen
- Weiterhin kann eine Mindestgröße oder Maximalgröße angegeben werden:
 - `min-width`, `min-height`
 - `max-width`, `max-height`

CSS

Größe, Position, Ausrichtung

```
#überschrift {width: 200px; height: 100px; background-color: turquoise;}
#absatz1 {width: 400px; height: 50px; background-color: beige;}
#absatz2 {width: 50px; height: 300px; background-color: lightcoral;}
</style>
</head>
<body>
  <h1 id="überschrift">Überschrift 1</h1>
  <p id="absatz1">
    Absatz 1
  </p>
  <p id="absatz2">
    Absatz 2
  </p>
</body>
```



CSS

Größe, Position, Ausrichtung

```
<style>
  #überschrift {
    width: 200px;
    min-height: 100px;
    background-color: turquoise;
  }
  #absatz1 {
    width: 400px;
    min-height: 50px;
    background-color: beige;
  }
  #absatz2 {
    width: 100px;
    min-height: 300px;
    background-color: lightcoral;
  }
</style>
```



CSS

Größe, Abstand, Positionen

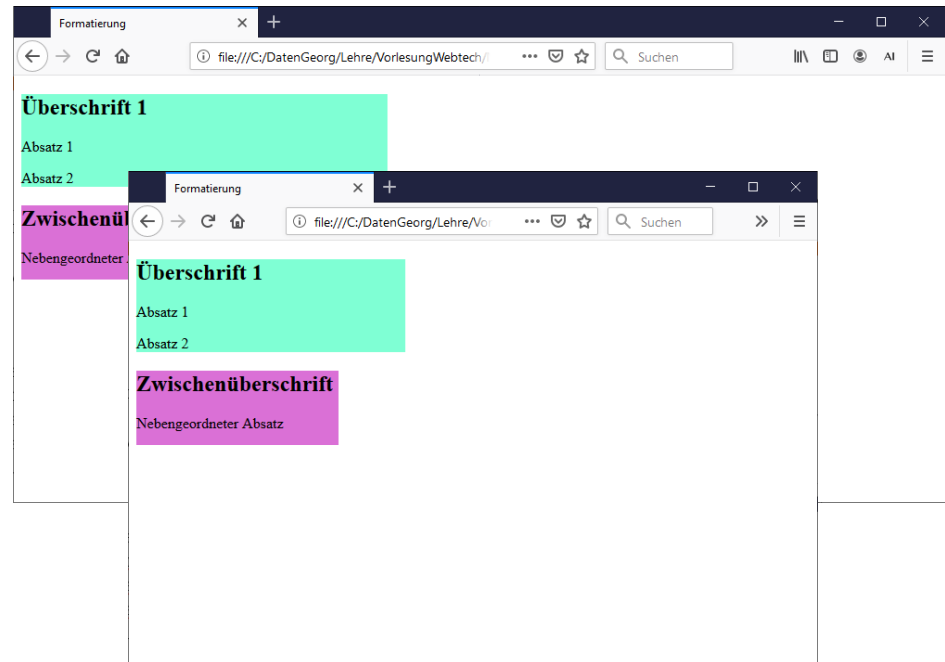
Größe von Elementen

- Die Verwendung prozentualer Werte führt dazu, dass sich die Bereiche an die Größe des Fensters anpassen.
- Die prozentualen Angaben beziehen sich jeweils auf die Größe des umgebenden Containers.
- Eine Größenangabe für die Seite ist daher oft sinnvoll:
 - `html { height: 100%; }`

CSS

Größe, Position, Ausrichtung

```
<style>
  article {width: 40%; min-height: 100px; background-color: aquamarine;}
  aside {width: 30%; min-height: 80px; background-color: orchid;}
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p>Absatz 1</p>
    <p>Absatz 2</p>
  </article>
  <aside>
    <h2>Zwischenüberschrift</h2>
    <p>Nebengeordneter Absatz</p>
  </aside>
</body>
```



CSS

Größe, Abstand, Positionen

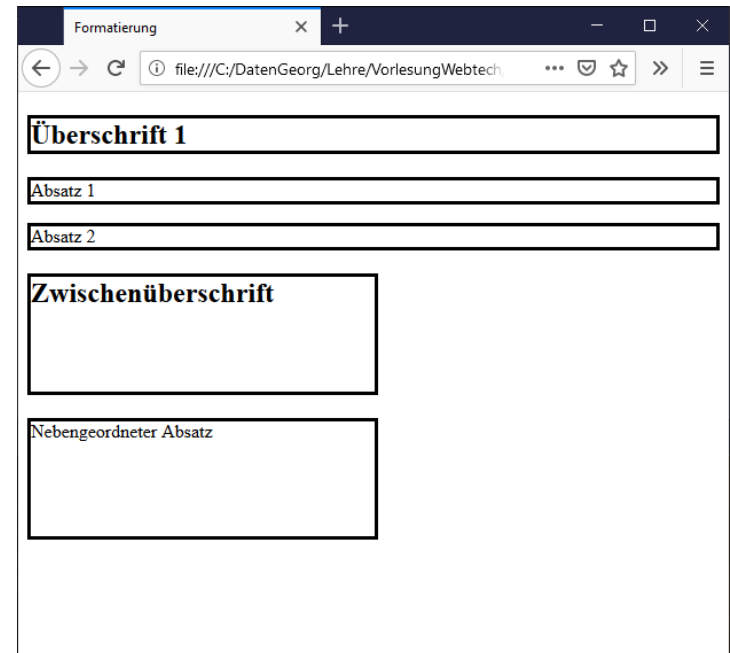
Rahmen um Elemente

- Um Elemente können Rahmen angebracht werden.
- Alle 4 Seiten können unabhängig bearbeitet werden.
- Es stehen viele verschieden Stile zur Verfügung.

CSS

Größe, Position, Ausrichtung

```
...
<style>
  article * { border-style: solid; }
  aside * { border-style: solid; width: 300px; min-height: 100px; }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p>Absatz 1</p>
    <p>Absatz 2</p>
  </article>
  <aside>
    <h2>Zwischenüberschrift</h2>
    <p>Nebengeordneter Absatz</p>
  </aside>
</body>
</html>
```



CSS

Größe, Position, Ausrichtung

```
...
<style>
  p { border-top-style: dashed; border-right-style: double;
      border-bottom-style: dotted; border-left-style: hidden;
      width: 200px; }
  #absatz2 { border-style: inset; border-width: 30px;
            border-color: turquoise; width: 200px; }
  #aside1 { border-style: ridge; border-width: 30px;
            border-color: blue; border-radius: 30px; width: 300px; }
</style>
</head>
    ...
    <article>
      <h1>Überschrift 1</h1>
      <p>Absatz 1</p>
      <p id="absatz2">Absatz 2</p>
    </article>
    <aside id="aside1">
      <h2>Zwischenüberschrift</h2>
      <div>Nebengeordneter Absatz</div>
    </aside>
    ...
```

CSS

Größe, Position, Ausrichtung

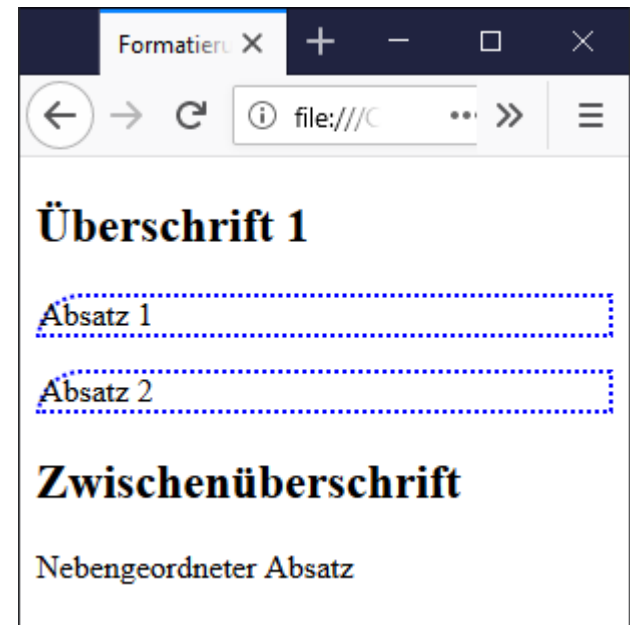
```
...
<style>
  p { border-top-style: dashed; border-right-style: double;
      border-bottom-style: dotted; border-left-style: hidden;
      width: 200px; }
  #absatz2 { border-style: inset; border-width: 30px;
            border-color: turquoise; width: 200px; }
  #aside1 { border-style: ridge; border-width: 30px;
            border-color: blue; border-radius: 30px; }
</style>
</head>
...
<article>
  <h1>Überschrift 1</h1>
  <p>Absatz 1</p>
  <p id="absatz2">Absatz 2</p>
</article>
<aside id="aside1">
  <h2>Zwischenüberschrift</h2>
  <div>Nebengeordneter Absatz</div>
</aside>
...
```



CSS

Größe, Position, Ausrichtung

```
...  
<style>  
  p { border:2px dotted blue;  
        border-top-left-radius: 25px;}  
</style>  
</head>  
<body>  
  <article>  
    <h1>Überschrift 1</h1>  
    <p>Absatz 1</p>  
    <p id="absatz2">Absatz 2</p>  
  </article>  
  <aside id="aside1">  
    <h2>Zwischenüberschrift</h2>  
    <div>Nebengeordneter Absatz</div>  
  </aside>  
</body>  
</html>
```



CSS

Größe, Abstand, Positionen

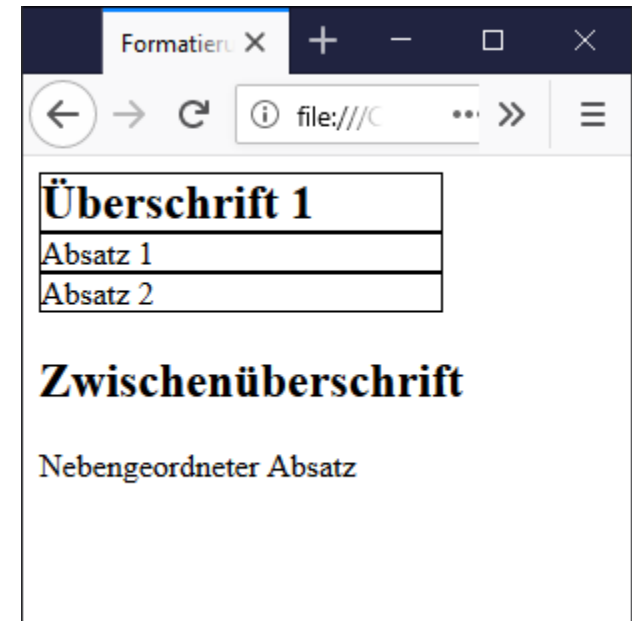
Innen- und Außenabstand

- Elemente haben einen Abstand zu ihrem Rand.
 - `padding`
- Elemente haben untereinander einen Abstand
 - `margin`

CSS

Größe, Position, Ausrichtung

```
...
<style>
  p, h1 { border: solid 1px; width: 200px; margin: 0px; }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p>Absatz 1</p>
    <p id="absatz2">Absatz 2</p>
  </article>
  <aside id="aside1">
    <h2>Zwischenüberschrift</h2>
    <div>Nebengeordneter Absatz</div>
  </aside>
</body>
</html>
```

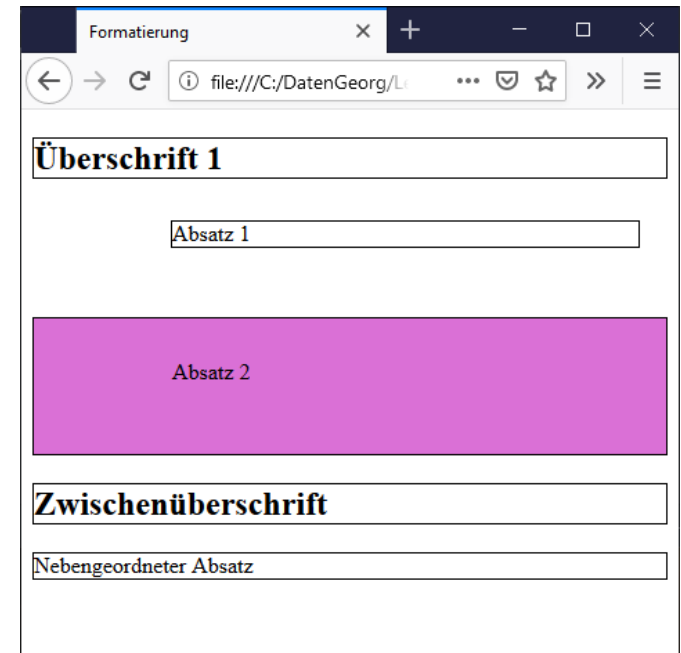


CSS

Größe, Position, Ausrichtung

...

```
<style>
  body * * {border: 1px solid black;}
  #absatz1 { margin-top: 30px; margin-right: 20px;
             margin-bottom: 50px; margin-left: 100px; }
  #absatz2 { padding: 30px 20px 50px 100px; background-color: orchid;}
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p id="absatz1">Absatz 1</p>
    <p id="absatz2">Absatz 2</p>
  </article>
  <aside id="aside1">
    <h2>Zwischenüberschrift</h2>
    <div>Nebengeordneter Absatz</div>
  </aside>
</body>
</html>
```



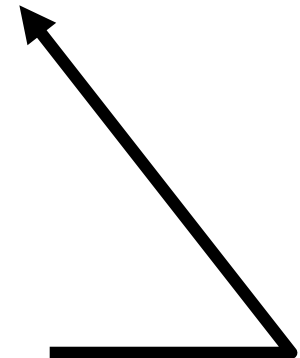
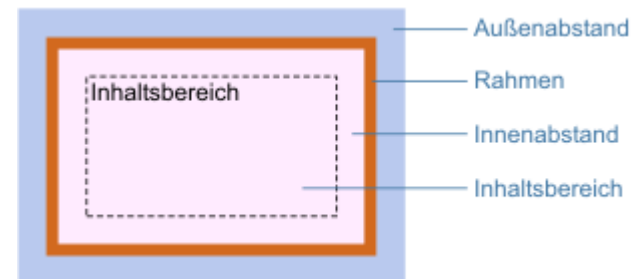
CSS

Boxmodell

Elementbox

Elementtypen:

- Wurzelement
 - Nicht-ersetzte Elemente
 - Ersetzte Elemente
 - Blockelemente
 - Inline-Elemente
-
- Jedes Element erzeugt Elementboxen
 - Elementbox besteht Inhaltsbereich und den Bereichen, die durch den Innenabstand, den Rahmen und den Außenabstand definiert sind
 - Hier nur Elementboxen für Blockelemente betrachtet



CSS

Boxmodell

Linie

- `border: 1px solid black;`

Bereich für Texte oder Bilder

- `width: 70px;`
- `height: 300px;`

Innenabstand

Hintergrundfarbe des Elementinhalts wird für padding-Bereich verwendet

- `padding: 10px;`

Außenabstand

Hintergrund im margin-Bereich ist transparent, d.h. Hintergrund des Elternelements ist sichtbar

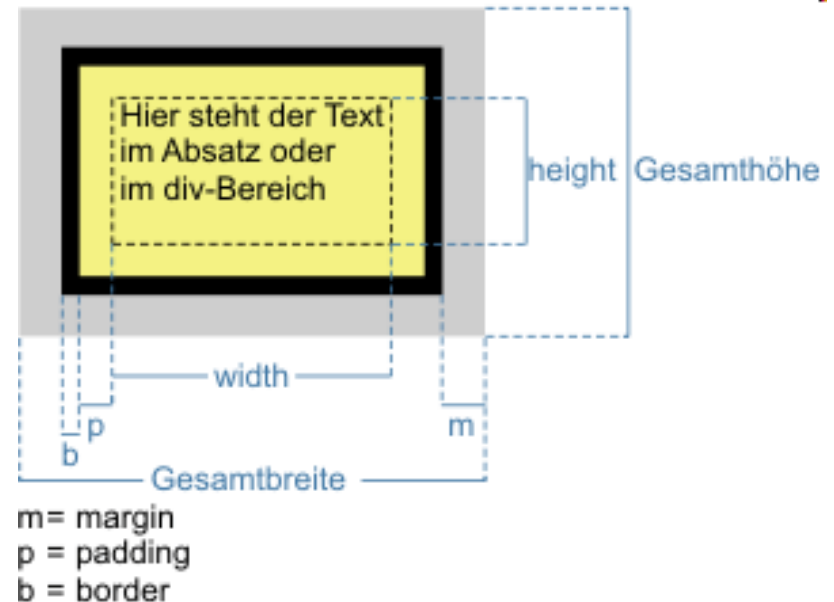
- `margin: 10px;`

CSS

Größenberechnung für Blockelemente 1

Gesamtbreite = width
+ margin-left + margin-right
+ border-left + border-right
+ padding-left + padding-right

Gesamthöhe = height
+ margin-top + margin-bottom
+ border-top + border-bottom
+ padding-top + padding-bottom

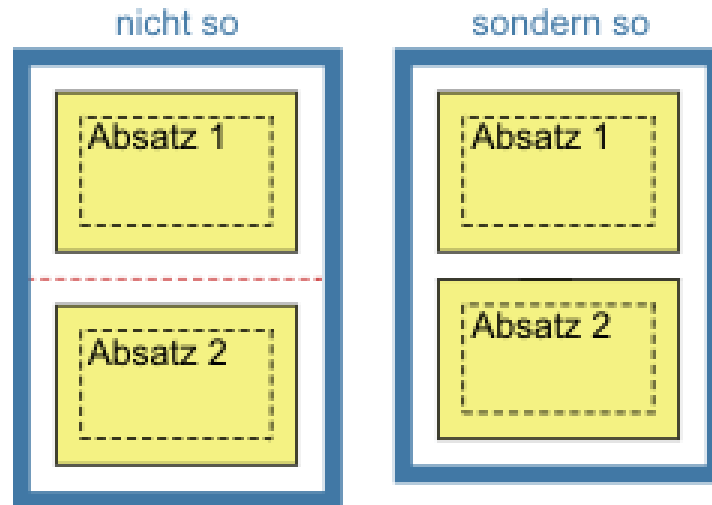


CSS

Größenberechnung für Blockelemente 2

Außenabstände von Blockelementen

- Wenn 2 Blockelemente vertikal aneinander grenzen, dann werden Außenabstände zusammengefasst.
- Bei unterschiedlichen Abständen wird der größere gewählt

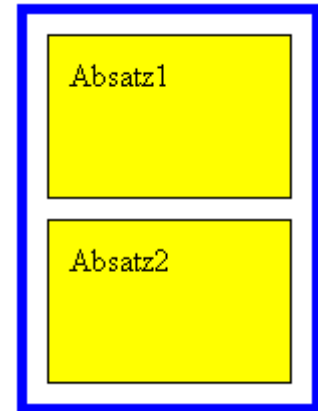


CSS

Beispiel zwei_boxen.html

```
<!DOCTYPE html>
<html>
<head>
<style>
div { border:5px solid blue;
      width:142px;
      height:194px;
    }
p{ border:1px solid black;
   padding:10px;
   margin:10px;
   background: yellow;
   width:100px;
   height:60px;
 }
</style>
</head>
```

```
<body>
<div>
  <p>Absatz1</p>
  <p>Absatz2</p>
</div>
</body>
</html>
```



CSS

Größe, Abstand, Positionen

Absolute Positionierung

- Elemente können im Browserfenster absolut positioniert werden, durch Angabe von Pixelpositionen entlang der Achsen.
- Die Angaben beziehen sich auf das nächste Vorfahrenelement, das mit `position:` positioniert wurde. Falls das nicht existiert, ist es die Wurzel.
- Elemente können sich überlappen.
- Welches Element verdeckt wird, kann über den `z-index` beeinflusst werden.

CSS

Größe, Position, Ausrichtung

...

```
<style>
```

```
#a1 { width: 400px; height: 100px; background-color: orchid;
      position: absolute; top: 100px; left: 50px; }
#a2 { width: 300px; background-color: turquoise; position: absolute;
      top: 100px; left: 500px; }
#ue2 { width: 100px; background-color: blue; position: absolute;
      top: 200px; left: 50px; }
#a3 { width: 100px; background-color: red; position: absolute;
      top: 200px; left: 75px; }
```

```
</style>
```

```
</head>
```

```
...<article>
```

```
  <h1>Überschrift 1</h1>
```

```
  <p id="a1">Absatz 1</p>
```

```
  <p id="a2">Absatz 2</p>
```

```
</article>
```

```
<aside>
```

```
  <h2 id="ue2">Zwischenüberschrift</h2>
```

```
  <div id="a3">Nebengeordneter Absatz</div>
```

```
</aside> ...
```

CSS

Größe, Position, Ausrichtung

...

```
<style>
```

```
#a1 { width: 400px; height: 100px; background-color: orchid;
      position: absolute; top: 100px; left: 50px; }
#a2 { width: 300px; background-color: turquoise; position: absolute;
      top: 100px; left: 500px; }
#ue2 { width: 100px; background-color: blue; position: absolute;
      top: 200px; left: 50px; }
#a3 { width: 100px; background-color: red; position: absolute;
      top: 200px; left: 75px; }
```

```
</style>
```

```
</head>
```

```
...<article>
```

```
  <h1>Überschrift 1</h1>
```

```
  <p id="a1">Absatz 1</p>
```

```
  <p id="a2">Absatz 2</p>
```

```
</article>
```

```
<aside>
```

```
  <h2 id="ue2">Zwischenüberschrift</h2>
```

```
  <div id="a3">Nebengeordneter Absatz</div>
```

```
</aside> ...
```



CSS

Größe, Abstand, Positionen

Relative Positionierung

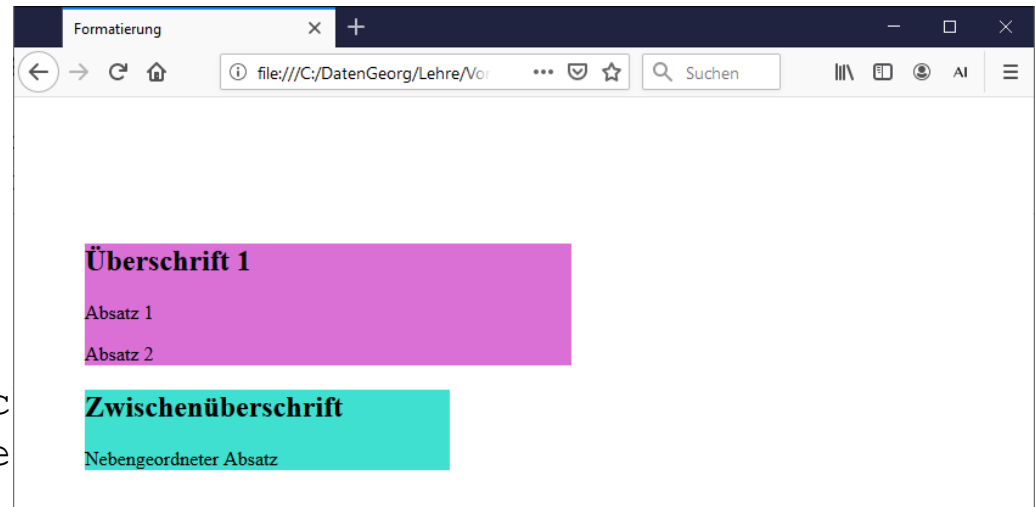
- Elemente können im Browserfenster relativ positioniert werden, durch Angabe von Pixelpositionen entlang der Achsen.
- Die Elemente werden dann verschoben zu ihrer eigentlichen Position dargestellt.

CSS

Größe, Position, Ausrichtung

...

```
<style>
  article { width: 400px; background-color: orchid;
            position: relative; top: 100px; left: 50px; }
  aside { width: 300px; background-color: turquoise;
          position: relative; top: 100px; left: 50px; }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p id="a1">Absatz 1</p>
    <p id="a2">Absatz 2</p>
  </article>
  <aside>
    <h2 id="ue2">Zwischenübersc
    <div id="a3">Nebengeordnete
  </aside>
</body>
</html>
```



CSS

Größe, Abstand, Positionen

Fixe Positionierung

- Elemente können im Browserfenster fixiert werden, durch Angabe von Pixelpositionen entlang der Achsen.
- Die Elemente bleiben auch beim Scrollen immer an der gleichen Stelle.

```
...
aside { width: 300px;
background-color: turquoise; position: fixed;
bottom: 50px; right: 50px; }
...
```

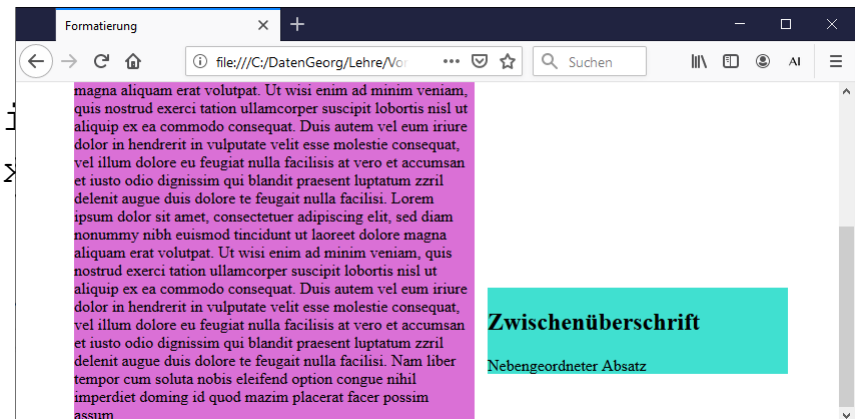
CSS

Größe, Abstand, Positionen

Fixe Positionierung

- Elemente können im Browserfenster fixiert werden, durch Angabe von Pixelpositionen entlang der Achsen.
- Die Elemente bleiben auch beim Scrollen immer an der gleichen Stelle.

```
...
aside { width: 300px;
background-color: turquoise;
bottom: 50px; right: 50px;
...
}
```



CSS

Größe, Abstand, Positionen

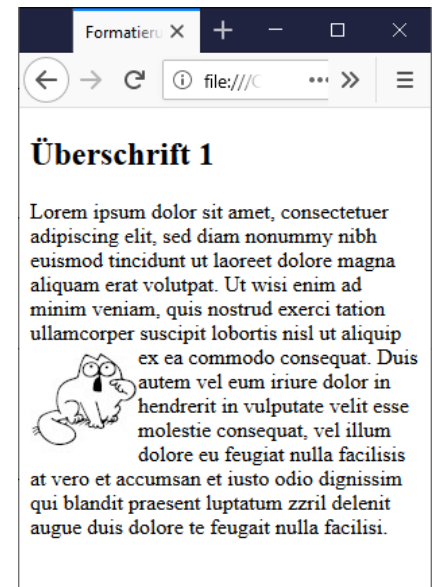
Float

- Elemente werden aus der normalen Positionierung herausgenommen und werden links oder rechts im Browserfenster positioniert.
- Der Positionierungsprozess geht danach normal weiter. Text kann die so positionierte Box umfließen.

CSS

Größe, Position, Ausrichtung

```
...
<style>
  .bild{
    float:left;
  }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p> Lorem ipsum dolor sit amet, ...
       ...
      Duis autem vel eum iriure dolor in hendrerit in.
    </p>
  </article>
</body>
</html>
```



CSS

Float

- `float:left;` Der Bereich gleitet an den linken Browserrand.
- `float:right;` Der Bereich gleitet an den rechten Browserrand.
- `float:none;` Der Bereich beginnt in einer neuen Zeile

`float`-Elemente

- Linke und rechte Außenabstände werden nicht zusammengefasst
- `float`-Elementen überschneiden sich nicht

CSS-Eigenschaft `clear`

- `div.fuss{ clear:left; }`
Nach `float:left` wird in einer neuen Zeile mit dem Boxen-Layout begonnen

Analog

- `clear:right;`
- `clear:both;`

CSS

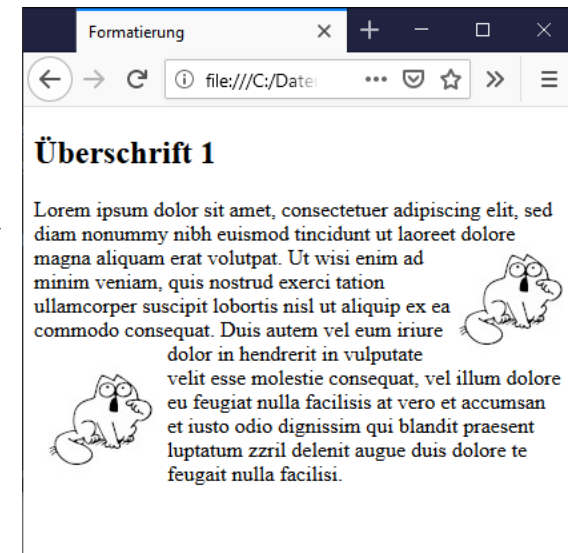
Größe, Position, Ausrichtung

```
...
<style>
  .bild1{ float:right; }
  .bild2 { float:left; margin:10px; clear:right; }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p> Lorem ipsum ...
       Ut wisi enim ad
        ...
        
        ...
    </p>
  </article>
</body>
</html>
```

CSS

Größe, Position, Ausrichtung

```
...
<style>
  .bild1{ float:right; }
  .bild2 { float:left; margin:10px; clear:right; }
</style>
</head>
<body>
  <article>
    <h1>Überschrift 1</h1>
    <p> Lorem ipsum ...
       Ut wisi enim ad
        ...
      
        ...
    </p>
  </article>
</body>
</html>
```



CSS

Größe, Abstand, Positionen

Overflow

- Elemente können den Inhalt der umgebenden Box überragen.
- Mit Hilfe des Attributs `overflow` werden dann Scrollbars eingefügt.

CSS

Größe, Position, Ausrichtung

```
...  
<style>  
  article { width: 300px; height: 200px; background-color: aquamarine;  
            overflow: scroll; }  
</style>  
</head>  
<body>  
  <article>  
    <h1>Überschrift 1</h1>  
    <p> Lorem ipsum ... </p>  
  </article>  
</body>  
</html>
```



CSS

Größe, Abstand, Positionen

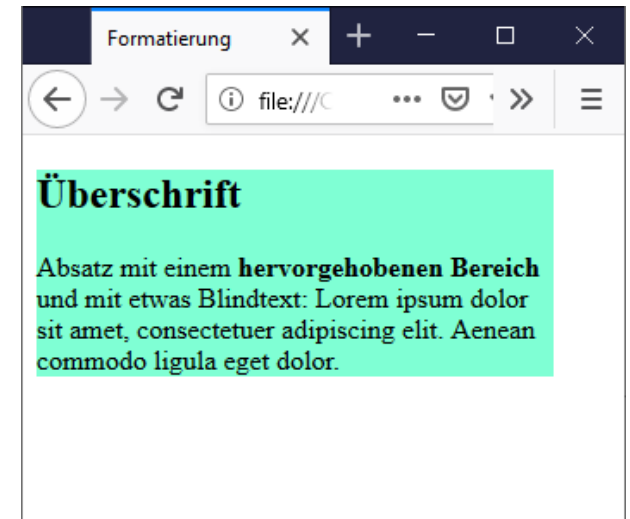
Das Attribut `display`

- Das Attribut dient dazu, die Art der Anzeige zu spezifizieren.
- Aus Blockelementen können Inlineelemente gemacht werden und umgekehrt: `display: block;` **versus** `display: inline;`
- Eine Mischung ist auch möglich: `display: inline-block;`
- Elemente können auch versteckt oder gar nicht dargestellt werden: `display: hidden;` **versus** `display: none;`
- Mit `display: flex;` kann ein neuer Boxtyp verwendet werden, mit mehr Optionen zur Darstellung.

CSS

Größe, Position, Ausrichtung

```
...  
<style>  
  article { width: 300px;  
            background-color: aquamarine; }  
</style>  
</head>  
<body>  
  <article> <h1>Überschrift</h1>  
    <p>Absatz mit einem <strong>hervorgehobenen Bereich</strong>  
    und mit etwas Blindtext: Lorem ipsum dolor sit amet,  
    consectetur adipiscing elit. Aenean commodo ligula eget  
    dolor.</p>  
  </article>  
</body>  
</html>
```



CSS

Größe, Position, Ausrichtung

```
<style>
  article { width: 300px; background-color: aquamarine; }
  strong { display: block; }
</style>
```

`strong` ist eigentlich ein Inline-Element.
Hier wird es als Block-Element dargestellt

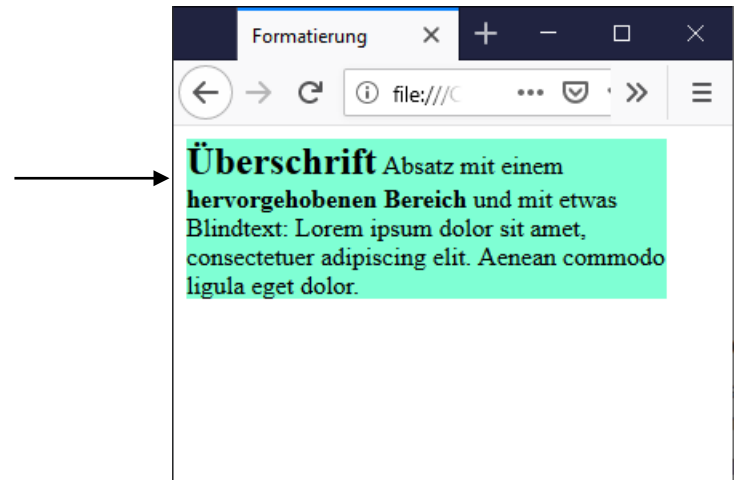


CSS

Größe, Position, Ausrichtung

```
<style>
  article { width: 300px; background-color: aquamarine; }
  h1 { display: inline; }
  p { display: inline; }
</style>
```

h1, p sind eigentlich Block-Elemente.
Hier werden sie als Inline-Element dargestellt.

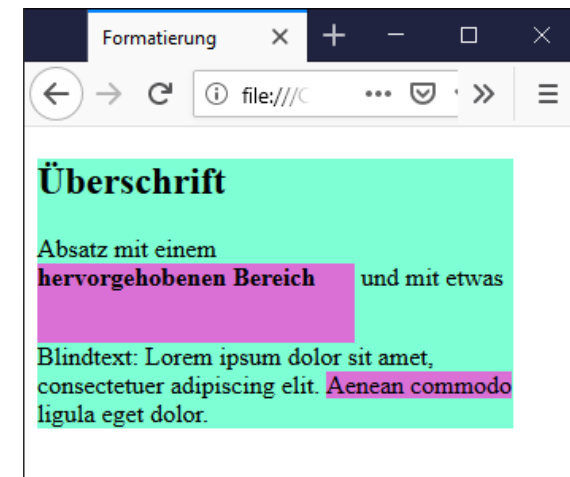


CSS

Größe, Position, Ausrichtung

```
<style>
  article { width: 300px; background-color: aquamarine; }
  strong { width: 200px; height: 50px; background-color: orchid;
           display: inline-block; }
  span { width: 200px; height: 50px; background-color: orchid; }
</style>
```

- Größenangaben sind für Inline-Elemente unwirksam.
- `strong` ist als `inline-block` dargestellt.
- Hierbei kommen die Angaben zur Geltung.
- Das Element wird trotzdem im Textfluss dargestellt.
- Falls keine Größenangaben existieren passt sich die Box an den Inhalt an (anders als bei Blockelementen)

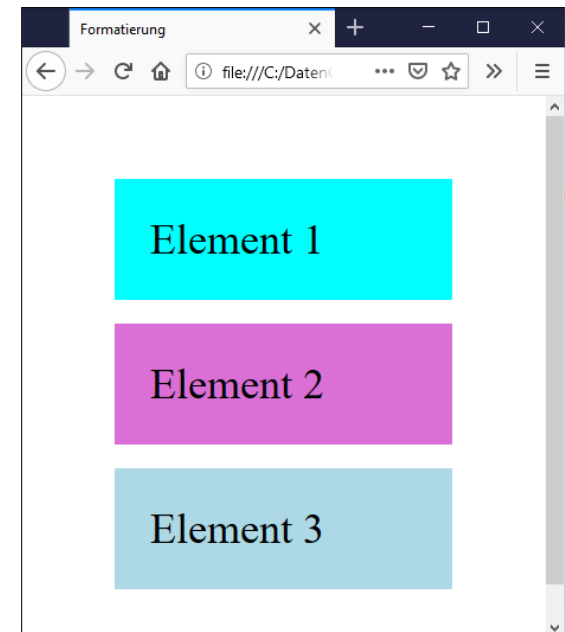


CSS

Größe, Position, Ausrichtung

Normale Darstellung von Blockelementen: Untereinander

```
...  
<style>  
div{ font-size: 1.5em; padding: 30px; margin: 20px; }  
#ele1 { background-color: aqua; }  
#ele2 { background-color: orchid; }  
#ele3 { background-color: lightblue; }  
</style>  
</head>  
<body>  
  <div id = "container">  
    <div id = "ele1">Element 1</div>  
    <div id = "ele2">Element 2</div>  
    <div id = "ele3">Element 3</div>  
  </div>  
</body>  
</html>
```



CSS

Größe, Position, Ausrichtung

Flexbox: Flexible Darstellung

```
div{ font-size: 1.5em; padding: 30px; margin: 20px; }  
#container { display: flex; }  
#ele1 { background-color: aqua; }  
#ele2 { background-color: orchid; }  
#ele3 { background-color: lightblue; }
```

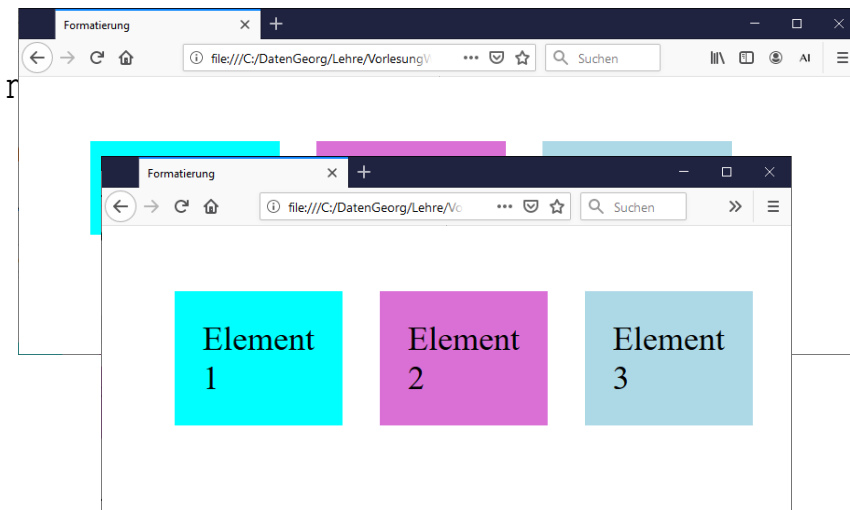
- Es muss ein umgebender Container mit `display: flex` dargestellt werden.
- Die Inhalte sind dann Flex-Items.
- Die Inhalte passen sich an die Größe des zur Verfügung stehenden Platzes so weit wie möglich an.

CSS

Größe, Position, Ausrichtung

Flexbox: Flexible Darstellung

```
div{ font-size: 1.5em; padding: 30px; }
#container { display: flex; }
#ele1 { background-color: aqua; }
#ele2 { background-color: orchid; }
#ele3 { background-color: lightblue; }
```

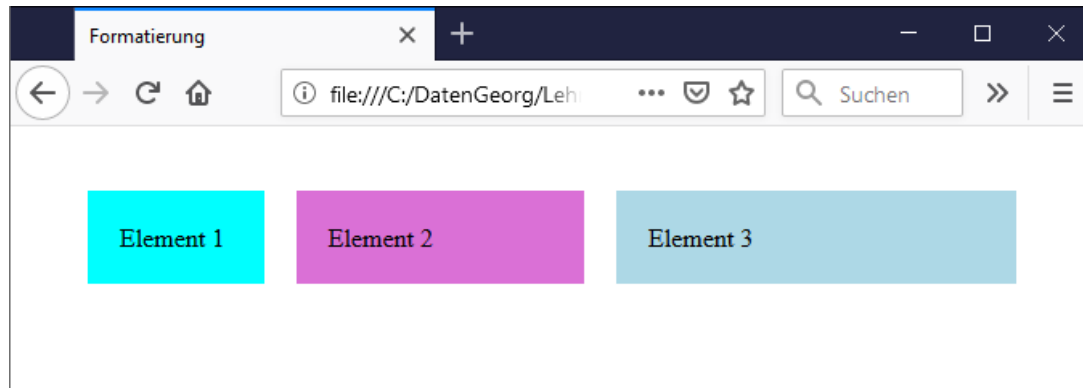


- Es muss ein umgebender Container mit `display: flex` dargestellt werden.
- Die Inhalte sind dann Flex-Items.
- Die Inhalte passen sich an die Größe des zur Verfügung stehenden Platzes so weit wie möglich an.

CSS

Größe, Position, Ausrichtung

```
div{ padding: 20px; margin: 10px; }  
#ele1 { background-color: aqua; flex: 1; }  
#ele2 { background-color: orchid; flex: 2; }  
#ele3 { background-color: lightblue; flex: 3; }  
#container { display: flex; }
```



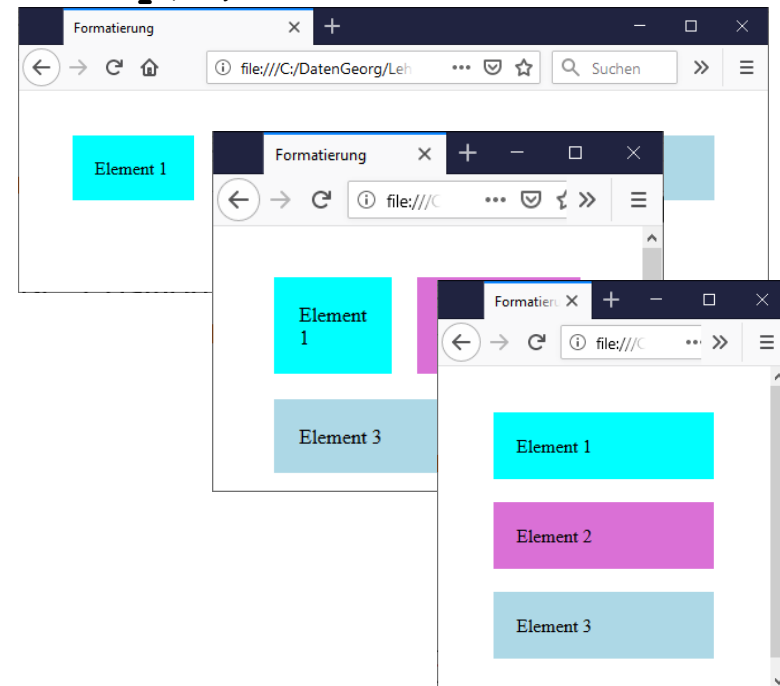
- Es können relative Größen zueinander angegeben werden.

CSS

Größe, Position, Ausrichtung

```
div{ padding: 20px; margin: 10px; }  
#ele1 { background-color: aqua; flex: 1; }  
#ele2 { background-color: orchid; flex: 2; }  
#ele3 { background-color: lightblue; flex: 3; }  
#container { display: flex; flex-wrap: wrap; }
```

- Die Zeile wird umgebrochen, wenn weniger Platz vorhanden ist.



CSS

Tabellen 1

- `width`: Breite der Tabelle
- `border`: Rahmen um die Tabelle
- `border-spacing`: Abstand zwischen den Tabellenzellen

Tabellenbreite

- `table { width:500px; }`
- `table { width:50%; }`

Tabellen unterschiedlicher Breite durch CSS-Klassen

- `table.schmal { width:50%; }`
- `table.breit { width:100%; }`

CSS

Tabellen 2

Tabellenrahmen

- `table { border:5px solid blue; }`
- `table{ border:5px solid; color:blue; }`

Abstand zwischen Zellen

- `table { border:5px solid;
border-spacing:15px; }`

Farben für Tabellen

- `table { background:silver;
color:blue; }`

CSS

CSS-Eigenschaften für Tabellenzellen

- `padding`: Innenabstand der Tabellenzelle
- `border`: Rahmen um die Zelle
- `text-align`: Horizontale Ausrichtung des Inhalts
- `vertical-align`: Vertikale Ausrichtung des Inhalts

Innenabstand

- `td.c { padding:10px; }`

Zellenrahmen

- `td { border:1px solid red; }`
- `td { border:1px solid; color:blue; }`

CSS

CSS-Eigenschaften für Tabellenzellen

Horizontale Ausrichtung

- `td.c1 { text-align:left; }`
- `td.c2 { text-align:right; }`
- `td.c3 { text-align:center; }`

Vertikale Ausrichtung

- `td.c1 { vertical-align:top; }`
- `td.c2 { vertical-align:middle; }`
- `td.c3 { vertical-align:bottom ; }`

CSS

Beispiel Tabellen 1

```
<style type="text/css">
table.t { width:500px;
          border:5px solid;
          border-spacing:15px;
          background:silver;
          color:blue;
        }
tr.head { color:white;
          background:navy;
        }
th.h { padding:10px;
        text-align:left;
      }

td.c { border:1px solid;
        padding:10px;
        background:yellow;
        color:red;
      }
td.c1 { text-align:right; }
td.c2 { vertical-align:top; }
</style>
```

CSS

Beispiel Tabellen 2

```
<table class="t">

<tr class="head">
  <th class="h">Spalte 1</th>
  <th class="h">Spalte 2</th>
</tr>
<tr>
  <td class="c c1">rechtsbündig</td>
  <td class="c">Standard</td>
</tr>
<tr>
  <td class="c c1 c2">rechtsbündig und<br /> oben</td>
  <td class="c"> <textarea name="area" cols="20" rows="5">
    Eingabebereich</textarea>  </td>
</tr>
</table>
```

CSS

Beispiel Tabellen 3

```
<table class="t">

<tr class="head">
  <th class="h">Spalte 1</th>
  <th class="h">Spalte 2</th>
</tr>
<tr>
  <td class="c c1">rechtsbündig</td>
  <td class="c">Standard</td>
</tr>
<tr>
  <td class="c c1 c2">rechtsbündig</td>
  <td class="c"><textarea name="Eingabebereich">
    Eingabebereich</textarea>
</td>
</tr>
</table>
```

| Spalte 1 | Spalte 2 |
|--------------------------|----------------|
| rechtsbündig | Standard |
| rechtsbündig und oben | Eingabebereich |

| Spalte 1 | Spalte 2 |
|--------------------------|----------------|
| rechtsbündig | Standard |
| rechtsbündig und oben | Eingabebereich |

CSS

Beispiel Tabellen 4

stylesheet.css

```
body { font-family:Verdana, sans-serif;
        font-size:small; }
h1 { font-size:large;
      font-weight:bold;
      color:#c33b00; }
...
table { width:500px; }
th { background-color:#a9a9a9;
      color:white;
      padding:10px; }
td { padding:10px;
      vertical-align:top; }
td.rechts { text-align:right; }
tr.row1 { background-color:#dddddd; }
tr.row2 { background-color:#eeeeee; }
```

CSS

Beispiel Tabellen 4

```
stylesheet.css
```

```
body { font-family:  
        font-size:
```


```
h1 { font-size:  
      font-weight:  
      color:#c333
```

```
...
```

Fit werden und fit bleiben - Mozilla Firefox

Startseite Die Vitamine Die Sportarten Ihre Fitness-Tipps

Die Vitamine



Vitamine sind wichtig für die Fitness.
Aber Vorsicht: Zu viel davon ist ungesund.

| Vitamin | Notwendig für | Enthalten in | Empfohlene Tagesmenge |
|-------------------|--|--|-----------------------|
| A (Retinol) | Wachstum und Aufbau von Haut und Schleimhaut, Sehen im Dunkeln | Obst und Gemüse (Beta-Carotin), Fisch, Leber | 2600 - 3300 IE |
| C (Ascorbinsäure) | die Bildung von Bindegewebe | frischem Obst und Gemüse | 100 mg |
| D (Calciferol) | den Knochenaufbau | Lachs, Kalbfleisch, Hühnerei | 200 - 400 IE |

Fertig

```
}  
color:#a9a9a9;
```

```
top; }  
align:right; }  
background-color:#dddddd; }  
background-color:#eeeeee; }
```

CSS

Ausgewählte Neuerungen (teilweise noch in Planung)

- Media Queries
- Präfixe
- Mehrspaltiges Layout, Rasterlayout
- Boxen: Flexbox, box-sizing
- CSS-Übergänge
- Dynamische Berechnungen: calc()
- Schlagschatten, Farbverläufe, Rotationen, Transparenz

Vergleich der Features bzgl. Browserunterstützung:

- <http://caniuse.com/>
- <http://css3test.com>
- http://www.w3schools.com/cssref/css3_browsersupport.asp
- [https://en.wikipedia.org/wiki/Comparison_of_layout_engines_\(Cascading_Style_Sheets\)](https://en.wikipedia.org/wiki/Comparison_of_layout_engines_(Cascading_Style_Sheets))

CSS

Medienspezifische Anweisungen

- `media="all"` CSS-Datei gilt für alle Medientypen.
- `media="speech"` CSS-Datei gilt für computergesteuerte, synthetische Sprachausgabe.
- `media="print"` CSS-Datei gilt für den Ausdruck auf Papier. Beachte CSS-Eigenschaften für Printmedien.
- `media="screen"` CSS-Datei gilt für die Bildschirmanzeige.

CSS

Beispiel medienspezifische Anweisungen

```
<head>
<title>Titel der Datei</title>
<style>
@media screen {
  h1 {
    text-align: right;
    border-bottom: 3px dashed #00f;
    color: #008;
    background-color: inherit;
  }
}

@media print {
  h1 {
    color: #000;
    background-color: #fff;
  }
}
</style>
</head>
```

Alternativ

```
<link rel="stylesheet"
type="text/css" href="print.css"
media="print">
```


CSS

Media Queries

Mobile First!

- Die Auflösung von Geräten, mit denen im Web gesurft wird, hat sich erhöht.
- Smartphones haben eine andere Auflösung und Größe, wie Tablets oder PCs.
- Mit Media Queries kann die Auflösung eines Gerätes abgefragt werden und es können darauf zugeschnittene Layouts geladen werden.
- „Breakpoints“ stellen die Grenze dar, bei der von einem zu anderen Layout umgeschaltet wird.
- Man spricht im Zusammenhang von Designs, die sich an die Auflösung und Größe anpassen von responsive Design, liquid Design und fluid/elastic Design.

Vgl.: https://wiki.selfhtml.org/wiki/HTML/Tutorials/responsive_Webdesign

CSS

Media Queries

Skalierung des Inhalts unterbinden

- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`

Beispiel:

```
aside {  
  /* nur Farb- und Hintergrundformatierungen */  
}  
  
@media (min-width: 50em) {  
  aside {  
    float: left;  
    width: 50%;  }  
}  
  
@media (min-width: 60em) {  
  ...}
```

CSS

Weitere Techniken

Flexbox

- Container mit `display:flex` darstellen.
- Layout passt sich bis zu einem gewissen Grad selbstständig an.

Grid-Layout

- Container mit `display:grid` darstellen.
- Layout passt sich bis zu einem gewissen Grad selbstständig an.
z.B.: <https://wiki.selfhtml.org/wiki/CSS/Tutorials/Grid/Grid-Container>
https://www.w3schools.com/css/css_rwd_intro.asp

CSS

Browserweichen durch Präfixe (Vendor Specific Extensions)

Beispiel:

background: -webkit-gradient(linear, left top, right top, from #2F2727, color-stop(0.5, #2F2727), color-stop(0.95, #1a82f7), to #1a82f7);

/* Safari 5.1+, Chrome 10+ */

background: -webkit-linear-gradient(left, #2F2727, #1a82f7);

/* Firefox 3.6+ */

background: -moz-linear-gradient(left, #2F2727, #1a82f7);

/* IE 10 */

background: -ms-linear-gradient(left, #2F2727, #1a82f7 50%, #1a82f7 50%, #1a82f7);

/* Opera 11.10+ */

background: -o-linear-gradient(left, #2F2727, #1a82f7 50%, #1a82f7 50%, #1a82f7);

| prefix | organization |
|------------|---|
| -ms-, mso- | Microsoft |
| -moz- | Mozilla |
| -o-, -xv- | Opera Software |
| -atsc- | Advanced Television Standards Committee |
| -wap- | The WAP Forum |
| -khtml- | KDE |
| -webkit- | Apple |
| prince- | YesLogic |
| -ah- | Antenna House |
| -hp- | Hewlett Packard |
| -ro- | Real Objects |
| -rim- | Research In Motion |
| -tc- | TallComponents |