

CSS: Seiten gestalten

CSS

- CSS (Cascading Style Sheets) ist eine Sprache, die verwendet wird, um das Layout und die Gestaltung von Webseiten zu beschreiben. Im Laufe der Jahre gab es mehrere Versionen und Updates von CSS.



Versionen

1996: CSS Level 1

- **Textgestaltung und Farben:** CSS Level 1 ermöglichte es, Texte auf Webseiten zu gestalten und zu färben. Man konnte Schriftarten, Schriftgrößen, Textfarben und Hintergrundfarben festlegen.
 - **Internet Explorer 3 unterstützt teilweise CSS 1:** Der Webbrowser Internet Explorer 3 konnte einige, aber nicht alle Funktionen von CSS Level 1 verwenden.
-

Version

1998: CSS Level 2

- **Positionierung:** Mit CSS Level 2 konnte man Elemente auf der Webseite genau positionieren. Man konnte festlegen, wo ein Element auf der Seite erscheinen soll, zum Beispiel oben, unten, links oder rechts.
 - **unterschiedliche Ausgabemedien:** CSS Level 2 ermöglichte es, verschiedene Layouts für verschiedene Ausgabemedien zu erstellen. Zum Beispiel konnte man ein Layout für den Bildschirm und ein anderes für den Drucker festlegen.
 - **Aural stylesheets:** Diese Funktion ermöglichte es, Webseiten so zu gestalten, dass sie auch für Sprachausgabe-Geräte geeignet sind. Das war besonders nützlich für Menschen mit Sehbehinderungen.
-

Version

CSS Level 2.1

- **absolute Positionierung:** Diese Funktion erlaubte eine genauere Positionierung von Elementen auf der Webseite. Man konnte festlegen, dass ein Element immer an einer bestimmten Position bleibt, unabhängig von anderen Elementen.
 - **automatische Nummerierung (Seitenzahlen etc.):** Mit dieser Funktion konnte man automatisch Seitenzahlen und andere nummerierte Listen erstellen, ohne sie manuell einfügen zu müssen.
 - **Linksläufige Schriften (Arabisch, Hebräisch etc.):** CSS Level 2.1 unterstützte Sprachen, die von rechts nach links gelesen werden, wie Arabisch und Hebräisch. Man konnte Texte in diesen Sprachen korrekt darstellen.
-

Versionen

- mit HTML5 entwickelt, der neuesten Version der Sprache, die verwendet wird, um Webseiten zu erstellen. Zusammen bieten sie viele neue und erweiterte Funktionen für die Gestaltung moderner Webseiten.
-

Highlights von CSS3

- **Transparenzen erstmals möglich:** Mit CSS3 kannst du Elemente durchsichtig machen, sodass man durch sie hindurchsehen kann.
- **Selektoren mit neuen Layoutmöglichkeiten:** CSS3 bietet neue Wege, um bestimmte Teile einer Webseite auszuwählen und zu gestalten, was das Layouten einfacher und flexibler macht.
- **Mehr Typografie möglich:** Du hast mehr Möglichkeiten, wie Texte auf deiner Webseite aussehen können, zum Beispiel mit verschiedenen Schriftarten und -größen.
- **Runde Ecken und Farbverläufe ohne Bildbearbeitung:** Du kannst Ecken von Kästchen abrunden und schöne Farbverläufe direkt mit CSS machen, ohne extra Bilder erstellen zu müssen.
- **Elemente drehen, verzerren mit CSS Transforms:** Mit CSS3 kannst du Elemente drehen, vergrößern oder verkleinern und sogar verzerren, um interessante Effekte zu erzeugen.
- **CSS-Angaben nach Eigenschaft des Ausgabegeräts für Responsive Designs mit Media Queries:** Du kannst festlegen, wie eine Webseite auf verschiedenen Geräten (wie Handys, Tablets und PCs) aussehen soll, damit sie überall gut funktioniert und gut aussieht.

CSS

- **CSS ist wie HTML eine Klartextsprache:** CSS ist eine Sprache, die Menschen leicht lesen und schreiben können, ähnlich wie HTML.
- **Freie Sprache (keine Lizenzen notwendig):** Du musst keine Gebühr bezahlen oder eine spezielle Erlaubnis haben, um CSS zu benutzen.
- **Kann mit jedem Texteditor bearbeitet werden:** Du kannst CSS-Dateien mit einem einfachen Programm bearbeiten, das Text schreiben und speichern kann, wie z.B. Notepad.
- **Moderne Webseiten sind in der Regel komplett mit CSS in Verbindung mit Containern gestylt (DIV-Layout):** Heutzutage werden Webseiten meistens mit CSS gestaltet. Dabei werden sogenannte Container (oft DIV-Elemente) benutzt, um das Layout zu strukturieren.
- **Attribute, wie z.B. bgcolor, cellspacing, ... durch CSS ersetzt:** Früher hat man bestimmte Eigenschaften direkt im HTML-Code festgelegt (wie Hintergrundfarbe oder Abstände zwischen Tabellenzellen), aber jetzt macht man das meistens mit CSS.
- **font-Tag wird nur noch beschränkt angewandt:** Das "font"-Tag, das früher benutzt wurde, um Schriftarten und Schriftgrößen festzulegen, wird heutzutage fast nicht mehr benutzt, weil CSS dafür besser geeignet ist.
- **Anordnung von Elementen mit CSS:** CSS wird verwendet, um zu bestimmen, wie die verschiedenen Teile einer Webseite angeordnet und dargestellt werden.

Allgemeine HTML-Elemente, die keine eigenen Formatierungen mitbringen:

Allgemeines Block-Element: <div>

1. Was ist ein <div>?

- Ein <div> ist ein allgemeines Element in HTML, das verwendet wird, um Abschnitte einer Webseite zu erstellen.
- Es ist ein Block-Element, was bedeutet, dass es den gesamten verfügbaren Platz in einer Zeile einnimmt und einen neuen Absatz beginnt.

2. Standard-Verhalten:

- Ein <div> verhält sich ähnlich wie ein Absatz (<p>), aber es hat keine eigenen Formatierungen wie Textgröße oder Abstand.
- Es wird hauptsächlich verwendet, um verschiedene Teile einer Webseite zu organisieren.

Allgemeine HTML-Elemente, die keine eigenen Formatierungen mitbringen:

- **Beispiel:**

```
<div></div>
```

- **Mit Inhalt:**

- Sie können Inhalte wie Text, Bilder oder andere HTML-Elemente in ein `<div>` einfügen.

- `<div>`
- Das ist ein Beispieltext im Div-Element.
- `</div>`

- **Mit einer ID:**

- IDs helfen dabei, ein bestimmtes `<div>`-Element auf einer Webseite eindeutig zu identifizieren.
- Sie können IDs verwenden, um das Element mit CSS zu gestalten oder mit JavaScript zu manipulieren.

- `<div id="bereichA">`
- Inhalt von Bereich A.
- `</div>`

- **Warum `<div>` verwenden?**

- Strukturieren Sie den Inhalt Ihrer Webseite in logische Abschnitte.
 - Erstellen Sie Layouts und Design-Strukturen mit Hilfe von CSS.
-

- Ein "Allgemeines Inline-Element" in HTML ist ein Element, das Teil eines Textes ist und keine neue Zeile beginnt. Ein Beispiel dafür ist das ``-Element. Es wird häufig verwendet, um bestimmte Teile eines Textes zu stylen, also das Aussehen des Textes zu ändern, ohne dabei den Textfluss zu unterbrechen.
 - Hier ist ein Beispiel, das erklärt, was der gezeigte HTML-Code macht:
 - ``
 - Das erste ``-Element ist leer. Es hat keine Wirkung, weil kein Text oder Stil darin enthalten ist.
 - `Hallo Welt`
 - Das zweite ``-Element enthält den Text "Hallo Welt" und hat spezielle Stilregeln:
 - `color:#ff0000;` bedeutet, dass der Text rot gefärbt wird (die Farbe #ff0000 steht für Rot).
 - `font-weight:bold;` bedeutet, dass der Text fett gedruckt wird.
 - Zusammengefasst: Der zweite ``-Block sorgt dafür, dass "Hallo Welt" in roter, fatter Schrift angezeigt wird.
-

CSS

Stylesheets können in verschiedenen Varianten eingebunden werden

- Von einer externen Datei
- In einem *style-Element* im Kopf der HTML-Datei
- Inline, direkt in einem *style-Attribut*

Externe Datei

Externe Datei:

- Man speichert seine CSS-Regeln in einer separaten Datei mit der Endung .css.
- Diese Datei fügt man in deinem HTML-Dokument ein, indem man einen Link im Kopfbereich (<head>) deiner HTML-Datei hinzufügt.
- Beispiel:

```
<link rel="stylesheet" href=„Dateiname.css">
```

- Vorteile: Alle Styles sind in einer separaten Datei, was es einfacher macht, sie zu verwalten und wiederzuverwenden.
-

Im Kopf der HTML-Datei

Im Kopf der HTML-Datei:

man kann CSS-Regeln direkt in einem <style>-Tag im Kopfbereich (<head>) deiner HTML-Datei schreiben.

Beispiel:

```
<style>
  body {
    background-color: lightblue;
  }
  h1 {
    color: navy;
  }
</style>
```

Vorteile: Praktisch, wenn man nur ein paar CSS-Regeln hat und keine separate Datei verwenden möchte.

Inline-Stil:

Inline-Stil:

man kann CSS-Regeln direkt im style-Attribut eines HTML-Elements schreiben.

- Beispiel:
 - `<h1 style="color: red; background-color: lightblue;">Hallo Welt</h1>`
- Vorteile: Gut für einzelne Elemente, die speziell formatiert werden sollen, ohne dass du den Rest der Seite beeinflusst.



CSS Syntax

- In CSS, gestalten wir das Aussehen von Webseiten durch Regeln. Jede Regel hat zwei Hauptbestandteile: einen Selektor und einen Regelblock.

1. Selektor: Er gibt an, welches HTML-Element oder welche Elemente die Regel anwenden soll.

2. Regelblock: Hier schreiben wir die Eigenschaften und ihre Werte, um festzulegen, wie das Element aussehen soll. Der Regelblock wird von geschweiften Klammern {} umgeben.

- Jede Eigenschaft bekommt einen Wert zugewiesen, und nach jeder Zuweisung setzen wir ein Semikolon ;.

- Beispiele:

- Einzelner Selektor:

```
p { color: red; font-size: 16px; }
```

Hier wird gesagt, dass alle <p>-Elemente (Absätze) rot sein sollen (color: red;) und die Schriftgröße 16 Pixel beträgt (font-size: 16px;).

- Mehrere Selektoren:

- h1, h2 { margin: 10px; }

Hier wird festgelegt, dass sowohl <h1>- als auch <h2>-Elemente einen Rand von 10 Pixeln haben sollen (margin: 10px;).

CSS

CSS im Head

```
<style>
```

```
<head>
```

```
  <style>
```

```
    ...
```

```
  </style>
```

```
</head>
```

CSS

<head>

<style>

h1{font-size:17px;}

h3, h4{font-weight:normal; font-style:italic;}

h4{color:#ff0000;}

p{

color:#2d2d2d;

line-height:1.5em;

margin:2px 4px 2px 4px;

}

</style>

</head>

CSS: Ausgewählte Attribute

- **color**: "Farbe"
- **font-style**: [italic, normal]
- **font-weight**: [bold, normal]
- **font-size**: "Zahl in px, %, em oder [medium, small, x-small, xx-small, smaller, large, x-large, xx-large, larger]"
- **font-variant**: [normal, small-caps]
- **border-top-width**: "Zahl in px, % "
- **border-color**: "Farbe"
- **border**: [z.B. 2px solid green]
- **height**: "Zahl in px, % oder em"
- **width**: "Zahl in px, % oder em"
- **background-color**: [Farbe]
- **background-image**: "URL zu Bild" [z.B.:url(einBild.jpg)]
- **background-repeat**: [no-repeat, repeat-x, repeat-y]
- **padding-left**: "Zahl in px, % oder em"
- **padding-top**: "Zahl in px, % oder em"
- **margin**: "Zahl in px, % oder em"
- **margin-left**: "Zahl in px, % oder em"
- **position**: [absolute, normal, auto, fixed, relative]
- **visible**: [visible, hidden]
- **display**: [none, show]

Schriftbild

Schriftart

- Eigenschaft: font-family
- Wert: Name einer Schriftart z.B. Arial

font-family: georgia, "times new roman", serif;



Ist eine Schriftart nicht vorhanden, wird die nächste verwendet



Schriftbild

Schriftstil

- Eigenschaft: **font-style**
- Wert:
 - normal
 - italic (kursiv)
 - oblique (schräg)
 - inherit (geerbt vom Elternelement)



Schriftbild

Schriftvariante

- Eigenschaft: **font-variant**
- Wert:
 - normal
 - small-caps (Kapitälchen)
 - inherit (geerbt vom Elternelement)



Schriftbild

Numerische Angabe:

- Du kannst die Schriftgröße direkt in Pixeln (px) angeben, z.B. 11px. Das ist eine genaue Größe.
 - `p { font-size: 16px; }`

Relative Angabe:

- **em**: Die Größe wird relativ zur Schriftgröße des Eltern-Elements berechnet. Zum Beispiel, 1em ist gleich der Schriftgröße des Eltern-Elements.
 - `p { font-size: 1.5em; /* Die Schriftgröße ist 1,5-mal so groß wie die Schriftgröße des Eltern-Elements */ }`
- **ex**: Diese Einheit basiert auf der Höhe des Buchstabens „x“ in der Schriftart, ist aber weniger gebräuchlich.
 - `p { font-size: 2ex; /* Die Schriftgröße ist zweimal so groß wie die Höhe des Buchstabens „x“ */ }`
- **%**: Du kannst die Schriftgröße auch in Prozent angeben. Zum Beispiel bedeutet 120%, dass die Schriftgröße 120% der Größe des Eltern-Elements ist.
 - `p { font-size: 120%; /* Die Schriftgröße ist 120% der Größe des Eltern-Elements */ }`



Schriftbild

2. Vordefinierte Werte:

- **xx-small**: Sehr kleine Schriftgröße.
 - `p { font-size: xx-small; /* Sehr kleine Schriftgröße */ }`
- **x-small**: Etwas größere Schrift als xx-small.
 - `p { font-size: x-small; /* Etwas größere Schrift als xx-small */ }`
- **small**: Kleine Schriftgröße.
 - `p { font-size: small; /* Kleine Schriftgröße */ }`
- **medium**: Mittelgroße Schriftgröße (die Standardgröße).
 - `p { font-size: small; /* Kleine Schriftgröße */ }`
- **large**: Größere Schriftgröße.
 - `p { font-size: large; /* Größere Schriftgröße */ }`
- **x-large**: Noch größere Schriftgröße.
 - `p { font-size: x-large; /* Noch größere Schriftgröße */ }`
- **xx-large**: Sehr große Schriftgröße.
 - `p { font-size: xx-large; /* Sehr große Schriftgröße */ }`

3. Verhältnis:

- **smaller**: Macht die Schriftgröße kleiner als die aktuelle Größe.
 - `p { font-size: smaller; /* Macht die Schriftgröße kleiner als die aktuelle Größe */ }`
- **larger**: Macht die Schriftgröße größer als die aktuelle Größe.
 - `p { font-size: larger; /* Macht die Schriftgröße größer als die aktuelle Größe */ }`

4. inherit:

- Die Schriftgröße wird von dem übergeordneten Element übernommen.
 - `p { font-size: inherit; /* Die Schriftgröße wird von dem übergeordneten Element übernommen */ }`

Mit diesen Optionen kannst du die Größe deiner Schrift auf verschiedene Weisen anpassen, je nach Bedarf.

Schriftbild

Zeilenhöhe

- Eigenschaft: **line-height**
- Wert:
 - normal
 - Fließkommazahl oder Prozentangabe



Schriftbild

Schriftgewicht

- Eigenschaft: **font-weight**
- Wert:
 - lighter
 - normal
 - bold
 - bolder
 - 100,200,300,400,500,600,700,800,900 (extra-dünn bis extra-fett)



Schriftbild

Schrift allgemein – Zusammenfassung der möglichen Einzelangaben

- Eigenschaft: **font**
- Reihenfolge:
font: *font-style font-variant font-weight font-size line-height font-family*

z.B.

- **font**:small-caps 700 24px georgia,serif
- **font**:italic 28px arial,serif
- **font**:bold .9em georgia,times,sans-serif



Aufgabe: Schriftbild

Experimentieren Sie mit dem Schriftbild

- font-family
- font-style
- font-variant
- font-size
- line-height
- font-weight
- font



Textformatierung

Textfarbe

- Eigenschaft: **color**
- Wert:
 - Farbname - z.B. green
 - RGB-Wert - z.B. rgb(40,255,12)
 - Hexadezimalwert - z.B. #00ff00



maroon #800000	red #ff0000	orange #ffa500	yellow #ffff00	olive #808000
purple #800080	fuchsia #ff00ff	white #ffffff	lime #00ff00	green #008000
navy #000080	blue #0000ff	aqua #00ffff	teal #008080	
	black #000000	silver #c0c0c0	gray #808080	



Textformatierung

Zeichenabstand

- Eigenschaft: **letter-spacing**
- Wert:
 - numerische Angabe - z.B. 1px
 - Normal
 - Beispiel
 - `<style>`
 - `.normal-spacing { letter-spacing: normal; }`
 - `.wide-spacing { letter-spacing: 2px; }`
 - `.narrow-spacing { letter-spacing: -1px; }`
 - `</style>`



Textformatierung

Wortabstand

- Eigenschaft: **word-spacing**
- Wert:
 - numerische Angabe - z.B. 1px
 - Normal
 - Beispiel
 - `<style>`
 - `.normal-spacing { word-spacing: normal; }`
 - `.wide-spacing { word-spacing: 10px; }`
 - `.narrow-spacing { word-spacing: -2px; }`
 - `</style>`



Textformatierung

Textdekoration

- Eigenschaft: **text-decoration**
- Wert:
 - underline
 - overline
 - line-through
 - none
 - (blink)



Textformatierung

Text-Transformation

- Eigenschaft: **text-transform**
- Wert:
 - capitalize (Wortanfänge als Großbuchstaben)
 - uppercase (nur Großbuchstaben)
 - lowercase (nur Kleinbuchstaben)
 - None
 - `<style> .capitalize { text-transform: capitalize; }`
 - `.uppercase { text-transform: uppercase; }`
 - `.lowercase { text-transform: lowercase; }`
 - `.none { text-transform: none; }`
 - `</style>`



Textformatierung

Erklärung der CSS-Klassen:

- `.capitalize`: Transformiert den ersten Buchstaben jedes Wortes in Großbuchstaben.
 - `.uppercase`: Transformiert alle Buchstaben in Großbuchstaben.
 - `.lowercase`: Transformiert alle Buchstaben in Kleinbuchstaben.
 - `.none`: Keine Transformation, der Text bleibt unverändert.
-

Aufgabe: Textformatierung

Experimentieren Sie mit den Textformatierungen

- color
- letter-spacing
- word-spacing
- text-decoration
- text-transform



Textausrichtung

Texteinrückung

- Eigenschaft: **text-indent**
- **Zweck:** text-indent wird verwendet, um den Einzug des ersten Absatzes in einem Blockelement (wie einem Absatz <p>) festzulegen. Dies bedeutet, dass der erste Textabschnitt eines Absatzes um einen bestimmten Abstand von der linken (oder rechten) Randlinie eingerückt wird.
- Wert:
 - positive oder negative Längenangabe
 - prozentuale Angabe relativ zur Breite des beinhaltenden Blocks
- **Positive Werte:** Schieben den ersten Satz des Absatzes weiter nach rechts.
- **Negative Werte:** Verschieben den ersten Satz des Absatzes nach links (kann auch außerhalb des Sichtbereichs sein).
- **Prozentuale Werte:** Die Einrückung basiert auf der Breite des umgebenden Blocks, nicht auf der Schriftgröße.



Textausrichtung

text-indent: 50px:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

text-indent: -2em:

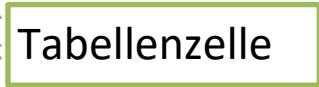
m ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

text-indent: 30%:


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

Textausrichtung


Vertikale Ausrichtung von Text in Zelle/von Inline-Objekt

- Eigenschaft: **vertical-align**
 - Wert:
 - positive oder negative Längenangabe
 - prozentuale Angabe
 - baseline
 - middle
 - sub
 - super
 - text-top
 - text-bottom
 - top
 - bottom
- 
- ```
graph LR; A[Tabellenzelle] --> B[middle]; A --> C[text-top]; A --> D[text-bottom];
```


### **vertical-align: baseline (default):**

An  image with a default alignment.


### **vertical-align: text-top:**

An  image with a text-top alignment.


### **vertical-align: text-bottom:**

An  image with a text-bottom alignment.

### **vertical-align: sub:**

An  image with a sub alignment.

### **vertical-align: sup:**

An  image with a super alignment.

# Textausrichtung

## Horizontale Ausrichtung

- Eigenschaft: **text-align**
- Wert:
  - left
  - right
  - center
  - justify (Blocksatz)
  - start (CSS 3.0)
  - end (CSS 3.0)





# Aufgabe: Textausrichtung

Experimentieren Sie mit der Textausrichtung

- text-indent
- vertical-align
- text-align



# Hintergrundfarben

## Hintergrundfarbe

- Eigenschaft: **background-color**
- Wert:
  - Farbname - z.B. green
  - RGB-Wert - z.B. rgb(40,255,12)
    - Beispiel:  
[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_background-color4](https://www.w3schools.com/cssref/tryit.php?filename=trycss_background-color4)
  - Hexadezimalwert - z.B. #00ff00
    - Beispiel:[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_background-color2](https://www.w3schools.com/cssref/tryit.php?filename=trycss_background-color2)



# Hintergrundbild

## Hintergrundbild

- Eigenschaft: **background-image**
- Wert:
  - Pfad zu Hintergrundgrafik mit url(Pfadangabe)
  - z.B. url(images/bg.png)
  - Beispiel:  
[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_background-image](https://www.w3schools.com/css/tryit.asp?filename=trycss_background-image)



# Hintergrundbild

Kachelung (wie wird mit dem Hintergrundbild verfahren, wenn das Element größer ist)

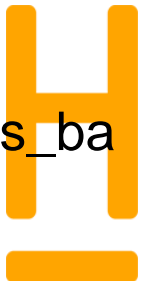
- Eigenschaft: **background-repeat**

- Wert:

- repeat
- repeat-x
- repeat-y
- no-repeat

- Beispiel:

[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_background-repeatx](https://www.w3schools.com/cssref/tryit.php?filename=trycss_background-repeatx)



# Hintergrundbild

Fixierung (wie verhält sich das Bild, wenn gescrollt wird)

- Eigenschaft: **background-attachment**
- Wert:
  - fixed
  - scroll
  - Local
    - Beispiet:  
[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_background-attachment](https://www.w3schools.com/cssref/tryit.php?filename=trycss_background-attachment)



# Hintergrundbild

## Positionierung

- Eigenschaft: **background-position**

- Wert:

- Paar von Längenangaben (x, y)

- z.B. 10px 20px

- Paar von Schlüsselwörtern

- left, right, top, bottom, center

- Beispiel:

- [https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_background-position](https://www.w3schools.com/cssref/tryit.php?filename=trycss_background-position)



# Hintergrund

Hintergrund allgemein – Zusammenfassung der möglichen Einzelangaben

- Eigenschaft: **background**
- z.B.
  - **background**:#ff0000 url(einBild.gif) left;
  - **background**:url(einBild.gif) repeat-x;



# Aufgabe: Hintergrund

Experimentieren Sie mit Hintergrundfarbe/-Bild

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background





# Listen

## Darstellungstyp

- Eigenschaft: **list-style-type**
- Wert für ungeordnete Listen (<ul>):
  - disc (gefüllter Kreis Bullet-Zeichen)
  - circle (leerer Kreis Bullet-Zeichen)
  - square (quadratisches Bullet-Zeichen)
  - none (kein Bullet-Zeichen)

Beispiel:

[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_list-style-type\\_ex](https://www.w3schools.com/cssref/tryit.php?filename=trycss_list-style-type_ex)



# Listen

- Wert für geordnete Listen (<ol>):
  - decimal (1., 2., 3., usw.)
  - decimal-leading-zero (01., 02., 03., usw.)
  - lower-roman (i., ii., iii., iv., usw.)
  - upper-roman (I., II., III., IV., usw.)
  - lower-alpha (a., b., c., d., usw.)
  - lower-latin (a., b., c., d., usw.)
  - upper-alpha (A., B., C., D., usw.)
  - upper-latin (A., B., C., D., usw.)
  - none (kein)



# Listen

## Einrückungsverhalten

- Eigenschaft: **list-style-position**
- Wert:
  - inside (eingerückt)
  - outside (ausgerückt)

Beispiel:

[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_list-style-position](https://www.w3schools.com/cssref/tryit.php?filename=trycss_list-style-position)



# Listen

## Grafik als Bullet-Zeichen

- Eigenschaft: **list-style-image**
- Wert:
  - URL Pfad
    - z.B. url(mybullet.png)

Beispiel:

[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_list-style-image](https://www.w3schools.com/cssref/tryit.php?filename=trycss_list-style-image)



# Listen

## Allgemein

- Eigenschaft: **list-style**

**list-style**:lower-roman inside;






# Hausaufgabe

---

- Verfeinern Sie die HTML-Struktur ihrer Webseiten und schaffen Sie individuelle Formatierungen mit CSS.

# Aufgabe: Kapitel 4 im Web Design Playground

 WEB DESIGN PLAYGROUND

Run Code ▶ MENU ☰

4.1: Specifying a Generic Font

◀ Previous Page

Next Page ▶

## Specifying a Generic Font

To apply a font to a web page element, use the CSS `font-family` property. To specify one or more system fonts as well as a related generic font, use the following general syntax:

HTML

font-family: "Font Name", Font, generic-font;

Copy to Clipboard

HTML

1


CSS

1

Hide Editors

New Sandbox

# Aufgabe: Projekt „Creating a Personal Homepage“

 WEB DESIGN PLAYGROUND

Run Code ▶ MENU ≡

◀ Previous Page

Next Page ▶

## Creating a Personal Home Page

This project takes you through the process of putting together a simple personal home page. The Results window shows the initial sketch for the home page.

### HTML

```
1
```

### CSS

```
1 body {
2 width: 550px;
3 color: #444;
4 font-size: 16px;
5 text-align: left;
6 }
```

Hide Editors New Sandbox