

Vorlesung

Webbasierte Informationssysteme

(CS4130)

Cascading Style Sheets (CSS)

Professor Dr. rer. nat. habil. Sven Groppe

<https://www.ifis.uni-luebeck.de/index.php?id=groppe>

Chronologische Übersicht über die Themen

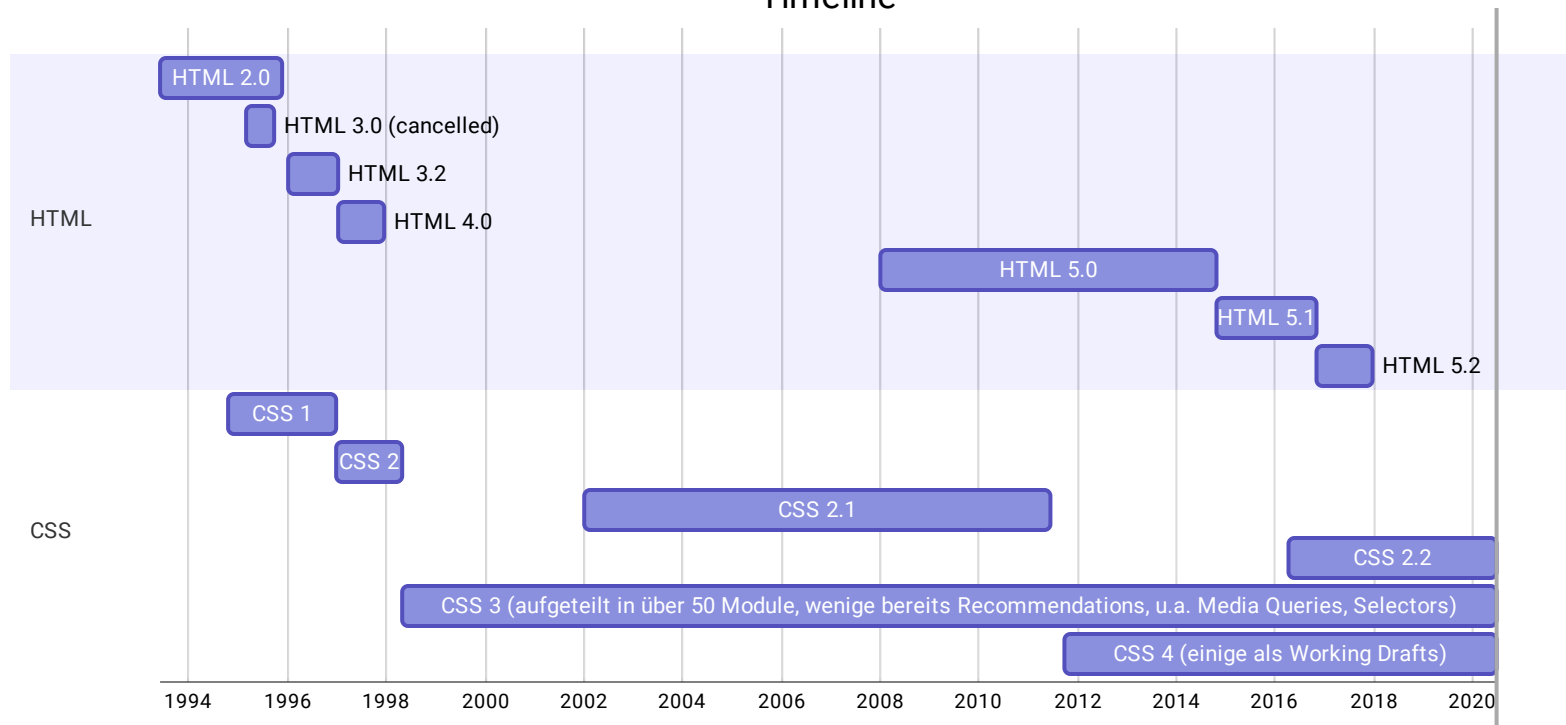
Ziele von Cascading Style Sheets (CSS)

- Trennung zwischen Inhalt und Darstellung von HTML-Dokumenten
- leistungsfähige Layout-Definition für HTML-Dokumente
- Anpassung an verschiedene Ausgabegeräte/-medien
- zentrales Layout-Management

Historie von HTML und CSS

- Entwicklung von CSS in „Leveln“ (**nicht** in Versionen)
 - Level bauen aufeinander auf: $Features(CSS\ x) \subseteq Features(CSS\ (x+1))$

Timeline



„Cascading“

- kombinierte Auswertung unterschiedlicher Arten von Stylesheets
 - Lösung von Konflikten zwischen anwendbaren Layout-Vorgaben mit Rücksicht auf Ursprung, Gewichtung und Spezialisierungsgrad
 - (vom Browser-Hersteller vorgegebenes) Browser-Stylesheet
 - definiert das Standard-Layout eines Browsers für die Elementinstanzen
 - [zum W3C-Vorschlag für dieses Stylesheet](#)
 - Benutzer-Stylesheet
 - definiert die Präferenzen eines Benutzers
 - Spezifikation über einen Browser-Dialog
 - Autoren-Stylesheet(s)
 - höchste Priorität bei der Darstellung
 - vom Autor entwickelte(s) Stylesheet(s) zum Layouten seines HTML Dokumentes

Einbindung bzw. Deklaration von CSS

1. In eigener CSS-Datei:

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

- `<link>`-Element darf nur im `<head>`-Element verwendet werden

2. Im HTML-Dokument:

```
<style type="text/css">  
// ... Hier werden die CSS-Regeln definiert ...  
</style>
```

- `<style>`-Element in dieser Form nur im `<head>`-Element

3. Im style-Attribut einer Elementinstanz:

```
<h3 style="color: red; font: arial">Überschrift!</h3>
```

- Syntax des style-Attributwertes = Syntax des Deklarationsteils einer CSS-Regel
- **Bemerkung:** Obige **unterschiedliche Arten von Deklarationen** sind miteinander **kombinierbar!**

Geltungsbereich von CSS-Deklarationen

- **Global** mittels `<link>` oder `<style>` im `<head>`-Element vorgenommene Deklarationen
- **Elementinstanz** mittels `style`-Attribut
 - Geltungsbereich:
Elementinstanz einschließlich ihrer Kindselemente
- **Geltungsbereich in einem gewissen Scope**
 - Ab HTML 5: Stylesheet-Deklarationen im Rumpf mit `<style scoped>`
- **Konfliktfall:**
lokale Deklarationen haben Vorrang vor globalen
- **Stylesheet-Deklarationen innerhalb HTML-Dokument:**
Widerspruch zum Paradigma der Trennung von Inhalt und Darstellung

Medientypen

- Angabe eines `media`-Attributs in `<link>`:
 - Spezifikation von Medientypen wie `screen`, `print`, `handheld`, `tv`, `all`, ...
 - Auswahl von passenden Stylesheets je nach Endgerät, Bsp.:

```
<link rel="stylesheet" href="base.css">  
<link rel="stylesheet" href="print.css" media="print">  
<link rel="stylesheet" href="s1a.css" media="screen">  
<link rel="stylesheet" href="s1b.css" media="screen">  
<link rel="alternate stylesheet" href="s2.css" media="screen" title="light">
```

Hier gilt:

- `base.css` für alle Medientypen
- `print.css` zusätzlich für den Medientyp `print` (z.B. für Druck/-vorschau)
- `s1a.css` und `s1b.css` oder alternativ `s2.css` zusätzlich für den Medientyp `screen`
- Zumeist im Browser-Menü: Auswahl von alternativen Stylesheets für den Medientyp `screen` (Menüeintrag = Text des `title`-Attributs)

Stylesheet-Regeln

- Stylesheet als Sammlung von Layout-Regeln
- Aufbau einer Layout-Regel, Bsp.:

```
h1, table { color: red; font: arial }
```

- Selektoren (hier `h1` und `table`)
 - Auswahl passender Elementinstanzen
 - Angabe mehrerer Selektoren durch Kommata getrennt, obiges Bsp. äquivalent zu:

```
h1      { color: red; font: arial }  
table   { color: red; font: arial }
```

- Deklarationsteil (hier { `color: red; font: arial` })
 - Menge von (durch Semikola getrennte) Layout-Vorgaben jeweils in Form von Eigenschaftswerte-Paare

Arten von Selektoren

Selektor	Beispiel	Adressierte Elemente
Elementtyp	<code>h2 {color: red}</code>	<code><h2> ... </h2></code>
Klassen	<code>.Classname {color: red}</code> <code>h2.Classname {color: blue}</code>	<code><div class="Classname"> ... </div></code> <code><h2 class="Classname Name2"> ... </h2></code>
ID	<code>#Identifier {color: red}</code>	<code><h2 id="Identifier"> ... </h2></code>
Pseudo-Klassen	<code>a:visited {color: red}</code>	<code> ... </code> link.html bereits besucht
Pseudo-Element	<code>p::first-line {color: red}</code>	<code><p> ... </p></code> 1. Zeile in der Darstellung des Paragraphen

Bemerkungen zu Pseudo-Selektoren

- Pseudoklassen¹

- Addressieren von Elementen mit bestimmten Eigenschaften

strukturelle Pseudoklassen (z.B. Pos. im Dokumentenbaum)	dynamische Pseudoklassen (z.B. Benutzeraktionen)	sonstige
:root :empty :first-child :last-child :nth-child() :nth-last-child() :only-child :first-of-type :last-of-type :nth-of-type() :nth-last-of-type() :only-of-type	:any-link :link :visited :hover :active :focus :focus-within :target :disabled :enabled :checked :valid :invalid :in-range :out-of-range	:lang() :not() :matches()

- Pseudoelemente²

- Selektion von **nicht** durch die Markup-Sprache auszeichenbaren Dokumenteninhalt

Pseudoelemente

::first-line ::first-letter ::before ::after ::backdrop ::selection

Attributselektoren

Selektor	Beispiel	Adressierte Elemente
Attributpräsenz	<code>h2[lang] {color: red}</code>	<code><h2 lang="..."> ... </h2></code>
Attributwert		
Exakte Übereinstimmung	<code>h2[lang=de] {color: red}</code>	<code><h2 lang="de"> ... </h2></code>
In Liste von Werten enthalten	<code>th[title~=Free]{color: red}</code>	<code><th title="Free Gift">...</th></code> Free durch Leerzeichen getrennt!
Teilübereinstimmung des Attributwertes		
Attributwert beginnt mit	<code>td[title^=Fr] {color: red}</code>	<code><td title="Freitag"> ... </td></code>
Attributwert endet mit	<code>td[title\$=tag] {color: red}</code>	<code><td title="Freitag"> ... </td></code>
Enthalten in Attributwert	<code>td[title*=ei] {color: red}</code>	<code><td title="Freitag"> ... </td></code>

Kombinierte Selektoren

- Selektion von **Knotenbeziehungen** im Dokumentenbaum

Selektor	Beispiel	Adressierte Elemente
Kind	<code>h2 > em {color: red}</code>	<code><h2>It's not deep</h2></code>
Nachfolger	<code>h2 em {color: red}</code>	<code><h2>It's very deep </h2></code>
Nächstes-Geschwister	<code>thead > tr + tr {color: orange}</code>	<code><thead><tr>...</tr> <tr>...</tr>...</thead></code>
Geschwister	<code>h1 ~ p {color: blue}</code>	<code><p>...</p> <h1>...</h1> <p>...</p> <table>...</table> <p>...</p></code>

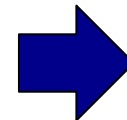
Klassenselektoren

- „Einführung neuer Elementtypen“
 - Meist durch folgende funktionslose HTML-Elementtypen:

Block-Elementtyp	Inline-Elementtyp
<code><div>...</div></code>	<code>...</code>

- Beispiel:

```
<html>
<head>
  <title>Example</title>
  <style type="text/css">
    .myheading { font-family: Roboto;
                  color       : blue;
                  font-size   : 20px; }
  </style>
</head>
<body>
  <div class="myheading">Mein Titelstil</div>
  bla...
</body>
</html>
```



Mein
Titelstil
bla...

Klassenselektoren - Bemerkungen

- (traditionelles) HTML: feste Dokumentstruktur
⇒ Vorgegebene Menge von Elementtypen und deren Darstellung
- 2 Schritte zur Schaffung von Elementinstanzen einer neuen Stylesheet-Klasse
(mit darüber hinaus ohne weitere intendierte Semantik):
 - Definition einer neuen Klasse mittels `.Classname { ... }`
 - Verwendung der neuen Klasse mittels `class="Classname"` in Elementinstanzen des Typs `<div>...</div>` oder `...`
- **Warnung des W3C:** Intendierte Semantik selbstdefinierter Klassen für Außenstehende oft nicht erkennbar
➔ Möglichkeit bitte nicht übertreiben!

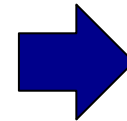
CSS-Selektor zum Adressieren von „folgenden Tabelle“



```
<!DOCTYPE html>
<html>
  <head>
    <title>WebInfo</title>
    <style type="text/css">

        ??? { font-style: italic }

    </style>
  </head>
  <body>
    <h2>Materialien zu WebInfo</h2>
    <p>Folien entnehmen Sie bitte der
      <span class="it" id="id1">
        folgenden Tabelle
      </span>:
    </p>
  </body>
</html>
```



Materialien zu WebInfo

Folien entnehmen Sie bitte
der *folgenden Tabelle* :

Responsive Web Design

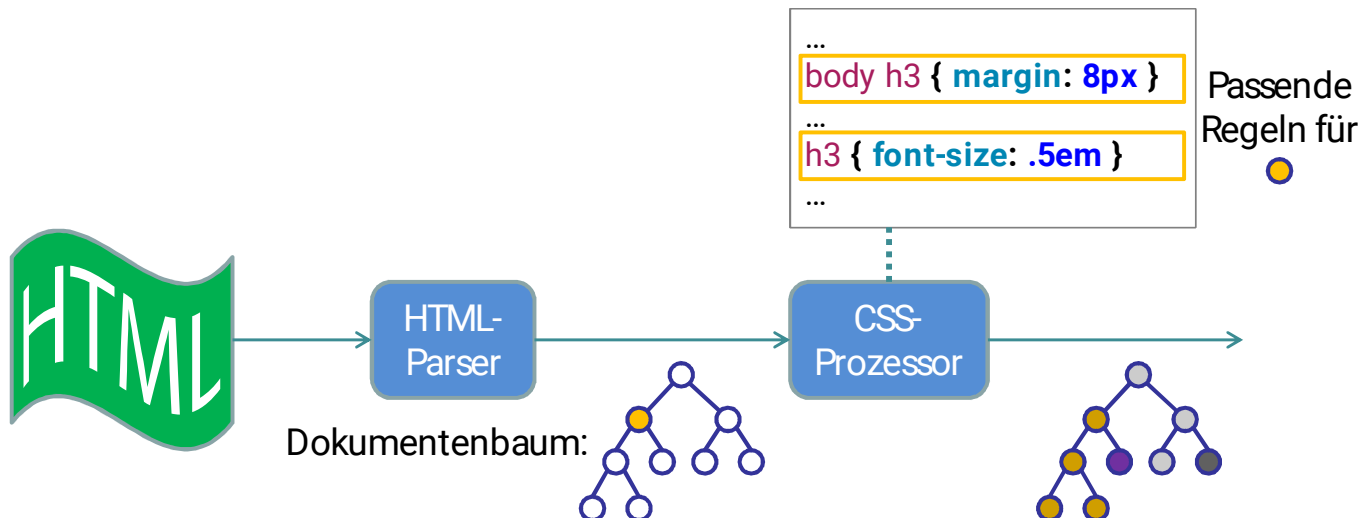
- Entwicklung von Webseiten, die die **Darstellung an unterschiedlich große Bildschirmgrößen** (z.B. von PC, Tablet und Smartphone) **anpasst**
- Unterstützung durch CSS:
 - Relative Angaben in Prozent, z.B. `width: 33%`
 - „Media Queries“
 - Unterschiedliche Stylesheets für unterschiedliche Bildschirmgrößen:

```
<link href="styles.css" rel="stylesheet" media="all and (max-width: 1024px)"/>
```
 - Innerhalb eines Stylesheets unterschiedliche Definitionen:

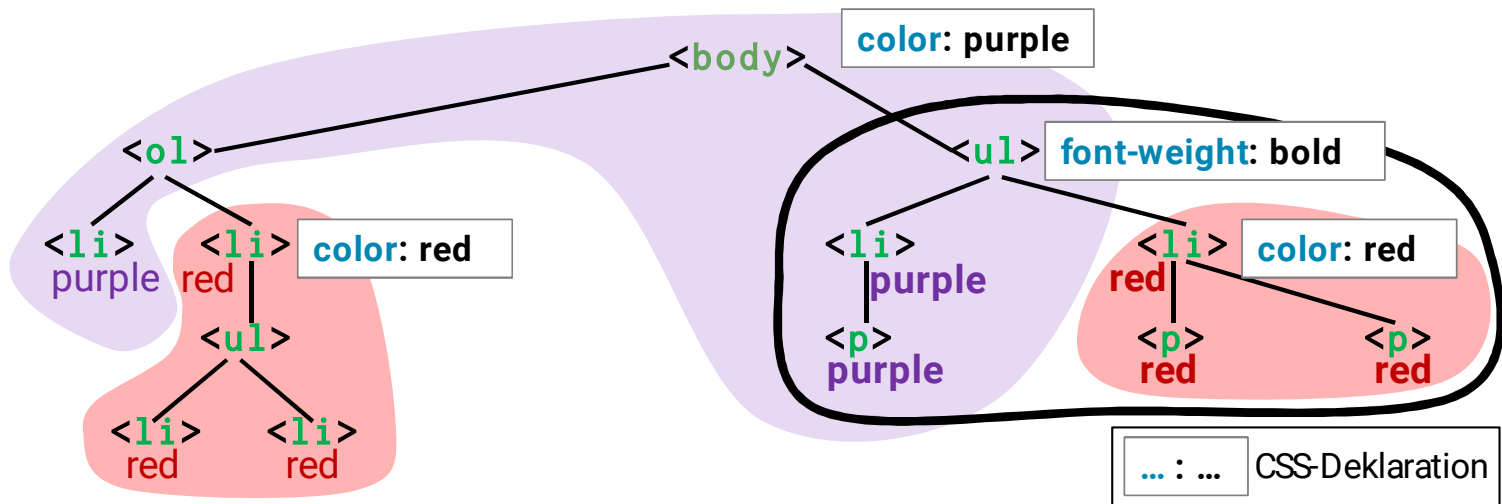
```
@media all and (max-width: 350px) {  
    /* hier folgen die Stylesheet-Definitionen für Smartphones */  
}
```
 - Viele weitere Sprachkonstrukte z.B. für Orientierung des Smartphones, Schwarz-Weiss-Displays, 3D-Brillen, ...

Darstellung einer Elementinstanz

1. Identifizierung der passenden Regeln in den zugehörigen Stylesheets anhand der Selektoren
2. Anwendung in der Reihenfolge von den weniger zu den mehr spezifischen Regeln
3. Beachtung von Vererbung und Reihenfolge (spätere Regeln vor früheren)



Vererbung von CSS-Deklarationen



- Vererbung von Regeln an eingebettete Elementinstanzen
- Lokale Vorgaben überschreiben vererbte und Default-Werte:

`color: purple` → `red`

`font-weight: normal` → **`bold`**

Mögliche Aspekte der Dokumentgestaltung

- Nahezu alle Aspekte der Dokumentgestaltung finden sich in CSS wieder
 - Maßeinheiten
 - Schrift
 - Farben
 - Listen
 - Tabellen
 - Box-Modell

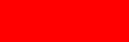


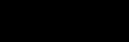


Maßeinheiten

Abkür- zung	Angabe- typ	Bedeutung	Beispiel
pt	absolut	Steht für Punkt. Typografische Maßeinheit. 1 Punkt entspricht 1/72 Inches.	<code>font-size: 12pt;</code>
pc	absolut	Steht für Pica. Typografische Maßeinheit. 1 Pica entspricht 12 Punkt.	<code>line-height: 1.2pc;</code>
in	absolut	Steht für Inch. Allgemeines Längenmaß im angelsächsischen Raum. 1 Inch entspricht 2.54 Zentimetern.	<code>margin-left: 1in;</code>
mm	absolut	Steht für Millimeter. Allgemeines Längenmaß. 1000. Teil des Urmeters.	<code>width: 70mm;</code>
cm	absolut	Steht für Zentimeter. Allgemeines Längenmaß. 100. Teil des Urmeters.	<code>font-size: 1cm;</code>
px	absolut /relativ	Steht für Pixel. Abhängig von der Pixeldichte des Ausgabegeräts, relativ also von Ausgabegerät zu Ausgabegerät, absolut dagegen auf ein und das selbe Ausgabegerät bezogen.	<code>border-width: 3px;</code>
em	relativ	Steht für bezogen auf die Schriftgröße des Elements. Ausnahme: wenn die <code>font-size</code> -Eigenschaft selbst mit dieser Maßangabe gesetzt wird, steht diese für bezogen auf die Schriftgröße des Elternelements.	<code>font-size: 1.2em;</code> <code>line-height: 1.5em;</code>
ex	relativ	Steht für bezogen auf die Höhe des Kleinbuchstaben x in diesem Element. Ausnahme: wenn die <code>font-size</code> -Eigenschaft selbst mit dieser Maßangabe gesetzt wird, steht diese für bezogen auf die Höhe des Kleinbuchstaben x im Elternelement.	<code>text-transform: uppercase;</code> <code>font-size: 1.3ex;</code>
%	relativ	Steht für Prozent. Je nach CSS-Eigenschaft relativ zur elementeigenen Größe, oder zu der des Elternelements, oder zu einem allgemeineren Kontext. Bei nebenstehendem Beispiel ist die Angabe zur Zeilenhöhe (<code>line-height</code>) relativ zur Schriftgröße (<code>font-size</code>) zu interpretieren.	<code>font-size: 10pt;</code> <code>line-height: 120%;</code>

Schrift

Beschreibung	Eigenschaft	Beispieleinträge
Farbe	<code>color</code>	<code>#00F</code> , <code>#0000FF</code> , black, purple
Schrifttypen-Familie	<code>font-family</code>	Arial, Verdana, Helvetica
Schriftgröße	<code>font-size</code>	<code>11px</code>
Gewichtung	<code>font-weight</code>	normal, bold, bolder, lighter, <code>100</code> (100-900 dünn bis fett)
Schrifttyp-Stil	<code>font-style</code>	normal, italic (kursiv)
Schrifttyp Variante	<code>font-variant</code>	normal, small-caps (Kapitälchen)
Zeichenabstand	<code>letter-spacing</code>	<code>0.5em</code>
Einrückung	<code>text-indent</code>	<code>30px</code>
Textumbruch	<code>white-space</code>	normal, pre (Zeilenumbrüche und Leerzeichen werden beachtet), nowrap (ohne Zeilenumbrüche außer bei <code>
</code> , keine Beachtung mehrerer Leerzeichen)
Textausrichtung	<code>text-align</code>	left, right, center, justify (Blocksatz)
Textdekoration	<code>text-decoration</code>	underline (unterstrichen), line-through (durchgestrichen), none,
Umwandlung der Buchstaben	<code>text-transform</code>	lowercase (Kleinschreibung), uppercase (Großschreibung), capitalize (Wortanfang in Großbuchstaben), none
Vertikale Ausrichtung	<code>vertical-align</code>	top, middle, bottom, sub (tiefgestellt), super (hochgestellt), <code>10%</code> (negative Werte erlaubt), baseline (Default)
Wortabstand	<code>word-spacing</code>	<code>4px</code>
Zeilenhöhe	<code>line-height</code>	<code>120%</code>

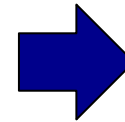
Farben



Farbe	Vordefinierter Name	Code (3-stellig)	Code (6-stellig)	RGB-Funktion	HSL-Funktion
	red	#f00	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)
	lime	#0f0	#00ff00	rgb(0, 255, 0)	hsl(120, 100%, 50%)
	blue	#00f	#0000ff	rgb(0, 0, 255)	hsl(240, 100%, 50%)
	black	#000	#000000	rgb(0, 0, 0)	hsl(0, 0%, 0%)
	white	#fff	#ffffff	rgb(255, 255, 255)	hsl(0, 0%, 100%)
	green	≈ #080	#008000	rgb(0, 128, 0)	hsl(120, 100%, 25%)

- **vordefinierte Namen** (CSS2: 17 Farben, CSS3: 140 Farben)
- Angabe des **Rot-Grün-Blau-Anteiles** einer Farbe
 - durch hexadezimalen Code (3 oder 6-stellig) oder durch rgb-Funktion mit Einzelwerten von 0 bis 255
- Angabe des **HSL-Wertes** durch hsl-Funktion (bei Designern beliebt)
 - Farbwert (englisch *hue*): Farbwinkel auf dem Farbkreis $\in [0^\circ, 360^\circ]$
 - Farbsättigung (*saturation*) $\in [0\%, 100\%]$ (Neutralgrau bis gesättigt, rein)
 - Helligkeit (*lightness*) $\in [0\%, 100\%]$ (keine bis volle Helligkeit)
- **Transparenz** $\alpha \in [0, 1]$ mittels rgba(r, g, b, α) bzw. hsla(h, s, l, α)

Ungeordnete Listen

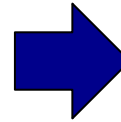
```
<!DOCTYPE html>
<html>
  <head>
    <title>Example</title>
    <style type="text/css">
      li.open {list-style: circle;}
      li.closed {list-style: disc;}
      li.square {list-style: square;}
      li.img {list-style: url(duck_32.png); }
    </style>
  </head>
  <body>
    <ul>
      <li>Normal</li>
      <li class="open">Open</li>
      <li class="closed">Closed</li>
      <li class="square">Square</li>
      <li class="img">Image</li>
      <li class="img">und noch 'ne Ente</li>
    </ul>
  </body>
</html>
```



- Normal
- Open
- Closed
- Square
-  Image
-  und noch 'ne Ente

Geordnete Listen

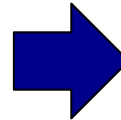
```
<style type="text/css">
  li.decimal      {list-style: decimal;}
  li.decimal-lz   {list-style: decimal-leading-zero;}
  li.lower-roman  {list-style: lower-roman;}
  li.upper-roman  {list-style: upper-roman; }
  li.lower-greek  {list-style: lower-greek; }
  li.lower-latin  {list-style: lower-latin; }
  li.upper-latin  {list-style: upper-latin; }
  li.armenian     {list-style: armenian; }
  li.georgian     {list-style: georgian; }
  li.lower-alpha  {list-style: lower-alpha; }
  li.upper-alpha  {list-style: upper-alpha; }
  li.none         {list-style: none; }
</style> ...
<ol>
  <li>normal</li>
  <li class="decimal">decimal</li>
  <li class="decimal-lz">decimal-leading-zero</li>
  <li class="lower-roman">lower-roman</li>
  <li class="upper-roman">upper-roman</li>
  <li class="lower-greek">lower-greek</li>
  <li class="lower-latin">lower-latin</li>
  <li class="upper-latin">upper-latin</li>
  <li class="armenian">armenian</li>
  <li class="georgian">georgian</li>
  <li class="lower-alpha">lower-alpha</li>
  <li class="upper-alpha">upper-alpha</li>
  <li class="none">none</li>
</ol>
```



1. normal
2. decimal
03. decimal-leading-zero
- iv. lower-roman
- V. upper-roman
- ζ. lower-greek
- g. lower-latin
- H. upper-latin
- Թ. armenian
- ռ. georgian
- k. lower-alpha
- L. upper-alpha
- none

Zähler und Nummerierungen

```
<style type="text/css">
  body {
    /* Create a chapter counter scope */
    counter-reset: chapter;
  }
  h1:before {
    content: "Chapter "
           counter(chapter, upper-latin)
           ": ";
    /* Add 1 to chapter */
    counter-increment: chapter;
  }
  h1 { /* Set section to 0 */
    counter-reset: section;
  }
  h2:before {
    content: counter(chapter) ". "
           counter(section) " ";
    counter-increment: section;
  }
</style> ...
<body>
  <h1>First Chapter</h1>
  <h2>First Section</h2>
  <h2>Second Section</h2>
  <h1>Second Chapter</h1>
  <h2>First Section</h2>
  <h2>Second Section</h2>
</body>
```



Chapter A: First Chapter

1.1 First Section

1.2 Second Section

Chapter B: Second Chapter

2.1 First Section

2.2 Second Section

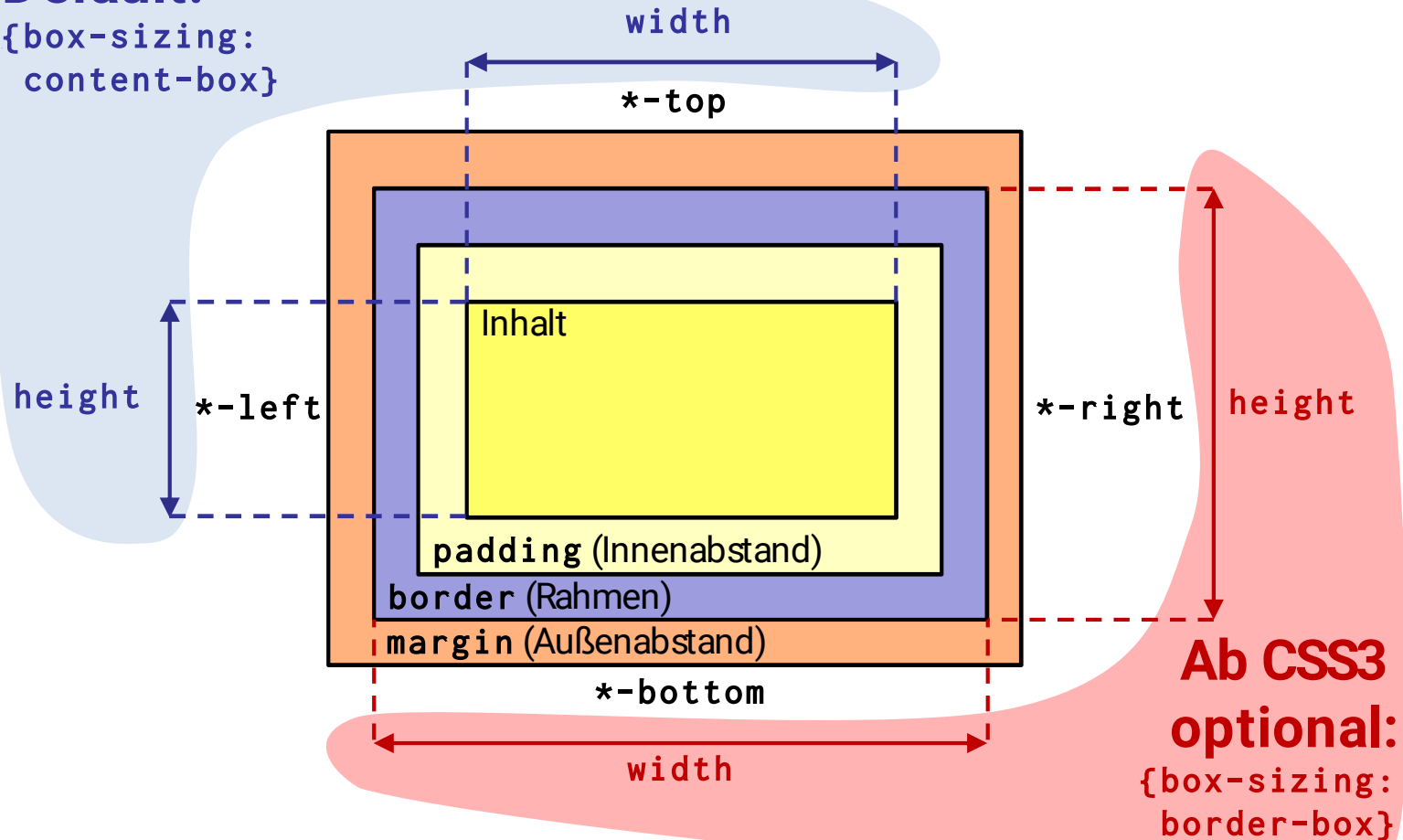
Modell der Visuellen Formatierung

- Jedes Element im Dokumentenbaum generiert 0 oder mehr Boxen
- Jedes dargestellte Element befindet sich in einer Box
 - ⇒ Rahmen und Außen-/Innenabstände
(wenn nicht 0)
- Verschachtelung von Boxen beliebig möglich
- Vererbung der Layout-Vorgaben von äußeren in innere Boxen

Box-Modell

Default:

```
{box-sizing:  
content-box}
```



Arten von Boxen

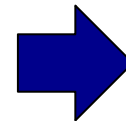
- Steuerung der Art der Box durch `display`-Attribut

Eintrag	Generierte Box
box	Block-Box
inline-box	Atomare Inline-Box, die eine Block-Box beinhaltet
inline	ein oder mehrere Inline-Boxen (für jede Zeile eine separate)
none	keine Box (auch keine unsichtbare), d.h. das Element und dessen Kindselemente (selbst bei anders gesetzten <code>display</code> -Attribut) werden nicht dargestellt

- Weitere Block-Arten für Listen und Tabellen

Arten von Boxen - Beispiel

```
<style type="text/css">
  p      { display: inline; }
  span.block { display: block;
                border: 2px solid #f00;}
  span.inline { display: inline;
                border: 2px solid #000;}
  span.inline_block { display: inline-block;
                      border: 2px solid #000;}
  span.none  { display: none }
</style>
...
<p>
  This is anonymous text before the block-SPAN.
  <span class="block">Block-SPAN</span>
  This is anonymous text between.
  <span class="inline">This is the content
    <span class="block">of</span>
    inline-SPAN.
  </span>
  This is anonymous text between blocks.
  <span class="inline_block">This is the content
    <span class="block">of</span>
    inline-block-SPAN.
  </span>
  <span class="none">This is not here...</span>
  It follows anonymous text after
  the inline-block-SPAN.
</p>
```



This is anonymous text
before the block-SPAN.

Block-SPAN

This is anonymous text
between. This is the content
of

inline-SPAN. This is
anonymous text between

This is the content
of

blocks. inline-block-SPAN. It
follows anonymous text
after the inline-block-SPAN.

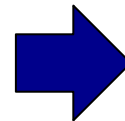
Positionierungsschemata

Fluss	Eigenschaft	Eintrag	Bemerkung
normal	position	static	Default
		relative	Verschiebung* der Box relativ zu seiner normalen Position
absolute		Positionierung* der Box absolut an einer anzugebenden Position	
absolut		fixed	Box wird zusätzlich nicht mitbewegt beim Scrollen und verbleibt immer an derselben Stelle*
	umlaufend	float	left right Box wird links bzw. rechts positioniert, Inhalt umläuft diese Box

* Positionierung durch `top`, `bottom`, `left` oder/und `right`-Eigenschaften

Bsp.: absolutes Positionieren und z-Index

```
<style type="text/css">
  .pile {
    position: absolute;
    left: 0.5in;
    top: 0.5in;
    width: 2in;
    height: 2in;
  }
  .greenBackground {
    width: auto;
    height: auto;
    background-color: #0f0
  }
  .z1 { z-index:1 }
  .z2 { z-index:2 }
  .z3 { z-index:3 }
</style> ...
<p>
  
  <div id="1" class="pile greenBackground z3">
    This text will overlay the duck image.
  </div>
  <div>This will be beneath everything.</div>
  <div class="pile z2">
    This text will underlay text 1,
    but overlay the duck image. Crazy!
  </div>
</p>
```



This will be beneath everything.

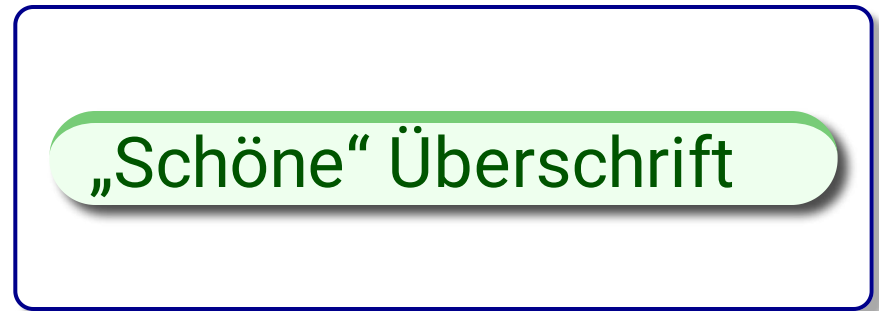
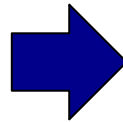
This text will overlay the
duck image.

overlay the duck
image. Crazy!



Bsp.: Rahmen und Schattierung

```
<style type="text/css">
  h3 {
    /* mid green */
    border-top: 6px solid #7c7;
    /* pale green */
    background-color: #efe;
    /* dark green */
    color: #050;
    border-radius: 25px;
    padding-left: 20px;
    padding-right: 20px;
    box-shadow: 5px 5px 5px
               rgba(0,0,0,0.7);
  }
</style>
...
<h3>„Schöne“ Überschrift</h3>
```



Bsp.: Tabelle

```
<table id="demo">
  <caption>Oceans</caption>
  <thead>
    <tr>
      <th></th>
      <th>Area</th>
      <th>Mean depth</th>
    </tr>
    <tr>
      <th></th>
      <th>million km
      <sup>2</sup></th>
      <th>m</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>Arctic</th>
      <td>13,000</td>
      <td>1,200</td>
    </tr>
    <tr>
      <th>Atlantic</th>
      <td>87,000</td>
      <td>3,900</td>
    </tr>
```

```
<tr>
  <th>Pacific</th>
  <td>180,000</td>
  <td>4,000</td>
</tr>
<tr>
  <th>Indian</th>
  <td>75,000</td>
  <td>3,900</td>
</tr>
<tr>
  <th>Southern</th>
  <td>20,000</td>
  <td>4,500</td>
</tr>
</tbody>
<tfoot>
  <tr>
    <th>Total</th>
    <td>361,000</td>
    <td></td>
  </tr>
  <tr>
    <th>Mean</th>
    <td>72,000</td>
    <td>3,800</td>
  </tr>
</tfoot>
</table>
```

Oceans

	Area	Mean depth
	million km ²	m
Arctic:	13,000	1,200
Atlantic:	87,000	3,900
Pacific:	180,000	4,000
Indian:	75,000	3,900
Southern:	20,000	4,500
Total:	361,000	
Mean:	72,000	3,800

Wie muss das CSS-Stylesheet dazu aussehen?

Bsp.: Tabelle

```
#demo {
  font: 100% sans-serif;
  background-color: #efe;
  border-collapse: collapse;
  empty-cells: show;
  border: 1px solid #7a7;
}
#demo > caption {
  text-align: left;
  font-weight: bold;
  font-size: 200%;
  border-bottom: .2em solid #4ca;
  margin-bottom: .5em;
}
#demo th, #demo td {
  text-align: right;
  padding-right: .5em;
}
#demo th {
  font-weight: bold;
  padding-left: .5em;
}
#demo > thead >
  tr:first-child > th {
    text-align: center;
    color: blue;
  }
```

```
#demo > thead > tr+tr > th {
  font-style: italic;
  color: gray;
}
#demo sup {
  font-size: 75%;
}
#demo td {
  background-color: #cef;
  padding: .5em .5em .5em 3em;
}
#demo tbody th:after {
  content: ":";
}
#demo tfoot {
  font-weight: bold;
}
#demo tfoot th {
  color: blue;
}
#demo tfoot th:after {
  content: ":";
}
#demo > tfoot td {
  background-color: #cee;
}
#demo>tfoot>tr:first-child td{
  border-top: .2em solid #7a7;
}
```

Oceans

	Area <i>million km²</i>	Mean depth <i>m</i>
Arctic:	13,000	1,200
Atlantic:	87,000	3,900
Pacific:	180,000	4,000
Indian:	75,000	3,900
Southern:	20,000	4,500
Total:	361,000	
Mean:	72,000	3,800

Quellen zum Nachschlagen und Nacharbeiten

- [MDN, CSS](#)
- [W3C, CSS Level 2](#)
- [W3C, CSS Specifications](#)
- [W3 Schools, CSS](#)
- [W3C, HTML Tidy](#)
(standardisieren und säubern von HTML-Code)
- [W3C, Markup Validation Service](#)

Zusammenfassung

- Ziele von Cascading Style Sheets (CSS)
- Stylesheet-Regeln
 - Geltungsbereich
 - Selektoren
 - Attribute
- Box-Modell und visuelle Formatierung
- Bestimmung anzuwendender Stylesheet-Regeln