OPEN EDUCATIONAL RESOURCES

```
<!doctype html>
<html lang="de">
 <head>
  <meta charset="utf-8">
  <title>HTML5 und CSS3</title>
  <meta name="author" content="Thomas Maier">
  <meta name="date" content="2022-03-15">
  <meta name="website" content="www.css4.at">
 </head>
 <body>
   <h1>Inhalt</h1>
   <ul>
     <li>Lernhandouts</li>
     <li>Übungsblätter</li>
   </ul>
 </body>
</html>
```

HTML &CSS

2024

HTML & CSS
von Thomas Maier

www.css4.at | html@css4.at

Copyright: CC Lizenz BY NC SA 2020, Version 2.1

Diese Arbeit wurde mit LibreOffice auf einem Linux Betriebssystem erstellt und entspricht damit einem Grundsatz der CreativeCommons. Alle Inhalte sind als OpenEducationalResources gekennzeichnet. Bitte prüfen Sie auf https://oer.css4.at ob eine neuere Version dieses Scriptes vorhanden ist!

Inhaltsverzeichnis

EDV Infrastruktur

HTML & CSS ist äußerst genügsam, was die Leistung der Hard- und Software betrifft. Im Grunde reicht ein gängiger PC mit entsprechender Standardperipherie (z. B. Tastatur, Maus, Bildschirm usw.). Das Betriebssystem spielt nur eine untergeordnete Rolle, da HTML und CSS plattformunabhängig ist. Ob nun Windows, OSX oder Linux – egal!

Dennoch sind folgende Software-Spezifikationen empfehlenswert:

- Texteditor (Kate, Notepad ++)
- HTML IDE (MS Visual Code, Atom)
- Bildbearbeitungssoftware (Photoshop, Gimp)
- sämtliche gängige Browser (Chrome, Firefox, Edge usw.)
- Audioeditor (Audacity)
- Textverarbeitungssoftware (MS Office, OpenOffice)
- weitere Tools: Color Picker, ZIP Software, Screenshot-Tools, PDF Viewer
- FTP Software (z. B. FileZilla)

Beispiele für eine Basisinstallation

		
Microsoft Windows	**Apple OSX**	**Linux**
• XAMPP (WAMPP) • Notepad ++ • Adobe Dreamweaver • ATOM • Edge, Chrome, Firefox • MS Office • MS Access • Color Picker • 7Zip • Snipping Tool • Adobe PDF Acrobat • FileZilla • OneNote • TeamViewer • Veyon (iTalc) • Gimp bzw. Adobe Photoshop • Corel Draw, Adobe Illustrator, Inkscape • Audacity • Oracle VirtualBox • MS SQL Server	• XAMPP • TextWrangler • Adobe Dreamweaver • ATOM • Safari, Chrome, Firefox • Apple iWork • MS Access • Color Picker • 7Zip • Skitch • Adobe PDF Acrobat • FileZilla • Evernote • TeamViewer • Veyon (iTalc) • Gimp bzw. Adobe Photoshop • Corel Draw, Adobe Illustrator, Inkscape • Audacity • Oracle VirtualBox • SQL Server	• XAMPP • Sublime • Bluefish • ATOM • Chrome, Firefox • LibreOffice • Kexi • KColorChooser • 7Zip • Shutter • Adobe Reader • FileZilla • Evernote • TeamViewer • Veyon (iTalc) • Gimp • Inkscape • Audacity • Oracle VirtualBox • SQL Server

Die Hardware ist definitiv genügsamer:

- Bei Laptops sollte ein externes Keyboard bereitgestellt werden.
  Ebenso eine Maus.
- Ein zweiter Monitor ist ein Nice-to-Have, aber nicht zwingend notwendig.
- Eine Breitband-Internet-Verbindung.
- Ein funktionierendes Netzwerk mit Serverseitigen Profilen und Netzlaufwerken.
- USB Anschlüsse für die Verwendung von USB-Speicher-Sticks.
- Ein Netzwerk-Drucker
- Mobile Devices (Smartphone und Tabletts) um die Scripte zu testen

Blendet Learning

Abbildung: Die Bannernavigation von www.css4.at

Auf der Lernplattform www.css4.at befindet sich links oben ein Suchfeld. Über dieses kann man nach notwendigen Dateien, Suchbegriffen und Refcodes suchen.

Auf der Startseite befindet sich der Lernbereich als dunkelblauer Kasten. Dort findet man unter Web Development einen Link zur Lernplattform: HTML und CSS.

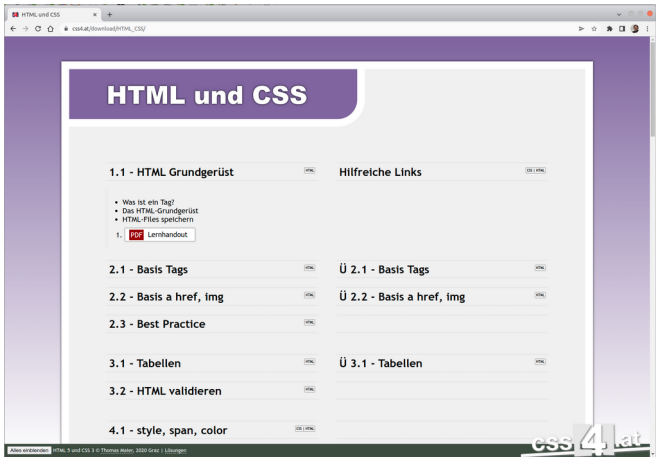


Abbildung: Lernplattform für HTML und CSS

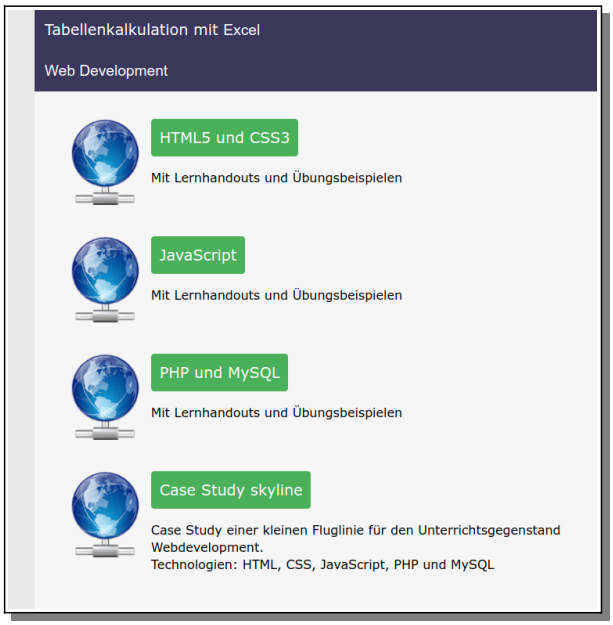


Abbildung: Im Lernbereich

Die Lernplattform für HTML und CSS erlaubt zu jeder Einheit die Downloads der Lern- und Übungshandouts und der benötigten Dateien für die Übungsbeispiele.

Zusätzlich findet man auf der Lernplattform noch hilfreiche Links zum Unterricht. In der linken Spalte findet man die Lernhandouts und korrespondierend rechts die dazu passenden Übungshandouts.

Handouts und Assignments

Die Handouts sind in Einheiten gegliedert. Jede Einheit beschreibt einen Themenkomplex aus dem jeweiligen Bezugssystem. Alle Handouts wurden nach dem Prinzip des "One-Page-Management" erstellt und können auch als Kopiervorlage genutzt werden. Die Handouts wurden mit Screenshots bebildert.

Die Symbole auf den Handouts sollen den Schüler_innen eine Ordnung im Lernprozess vermitteln.

 ⇦ ***Übungsbeispiel***

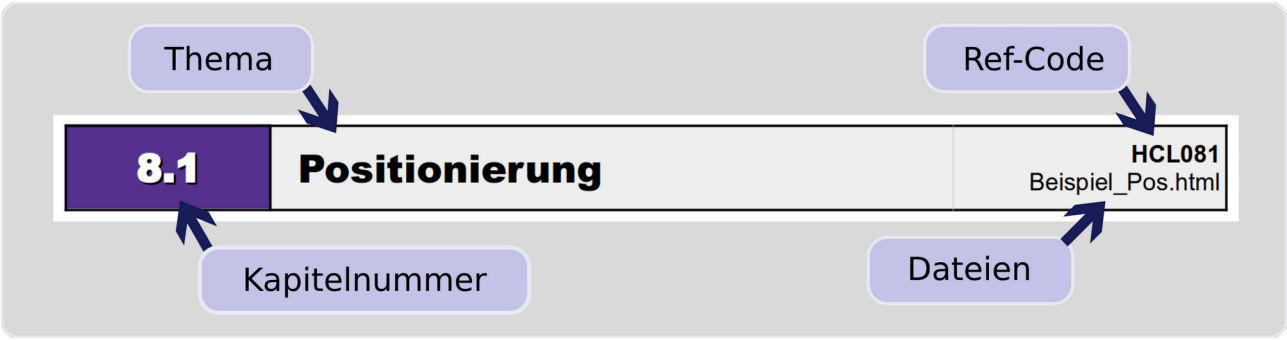
 ⇦ ***Information und Hervorhebung***

 ⇦ ***Nachschlagen und Bibliothek***

 ⇦ ***Tools und Werkzeuge***

 ⇦ ***Notizen und Anmerkungen***

Dieses One-Page-Management wurde der Wirtschaft entnommen. Es ist eine Methode um Führungspersönlichkeiten einen raschen Überblick über die aktuellen Fakten zu geben. Rasch und Einfach – was nun also für Manager von großen Betrieben stimmig ist, sollte für den Schüler bzw. die Schülerin genauso gelten. Ein neues Thema – ein neues Blatt Papier.



Über die Kapitelnummer werden Themenbereiche gegliedert. Übungsblätter haben zusätzlich noch ein "Ü" bei der Kapitelnummer. Der Ref-Code identifiziert ein Handout eindeutig und kann im Suchfeld von www.css4.at abgerufen werden. Dateien, die für das Thema notwendig sind, findet man als rechts, unter dem Ref-Code. Die Dateien können von www.css4.at herunter geladen werden! In dieser Arbeit folgt immer zuerst die didaktische Konzeption (Feinzielen und Anmerkungen) und im Anschluss daran das Schüler_innen Handout bzw. Übungsblatt.

1.1 HTML Grundgerüst

Ein HTML-Tag beginnt mit einem Start-Tag und endet mit einem End-Tag. Dazwischen ist der zu formatierende Bereich (z. B. Text).

Das Grundkonstrukt jeder HTML-Seite beginnt mit einem **<html>** *Tag. Darin befindet sich der* **<head>** *und der* **<body>**. *Im* **<head>** *stehen Metadaten wie z. B. der Titel. Im* **<body>** *ist der Inhalt der Website, der dann im Browser angezeigt wird.*

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <title>Titel der Webseite</title>
    <meta name="author" content="John Doe">
    <meta name="date" content="2022-04-20">
  </head>
 <body>
   Hier ist der sichtbare Teil der Website!
 </body>
</html>
```

<!doctype html>
└ *Definiert den Code als HTML Seite*

<html lang="de">
└ *HTML-Starttag mit einem Hinweis auf die Sprache. z. B. de für Deutsch*

<meta charset="utf-8">
└ *UTF8 Zeichenkodierung mit deutschen Sonderzeichen (ä, ö, ß usw.)*

<title>Titel der Webseite</title>
└ *Der Titel der Webseite*

<meta name="date" content="2022-04-20">
└ ***Ein Metatag beschreibt das Dokument. z. B. das Erstelldatum***

Speichern unter …
HTML Dateien haben die Extension (Dateierweiterung) ***.htm** bzw. ***.html**
Die **index.html** wird angezeigt, wenn man im Browser die URL eines Ordners eingibt – z. B: **www.johndoe.com/projekte**

Übung A: Das HTML Grundkonstrukt

- ❑ Starte einen Webeditor (Notepad ++, Dreamweaver, Atom odgl.)
- ❑ Erstelle ein HTML Grundgerüst
- ❑ Trage deinen Namen als Autor ein
- ❑ Speichere es als **vorlage.html** ab

Wir arbeiten in Zukunft immer mit der **vorlage.html** – speichere sie also so ab, dass du immer Zugriff darauf hast (USB-Stick, Cloud, Netzlaufwerk).

2.1 Basis Tags

Es folgen die geläufigsten HTML-Elemente. Nicht vergessen: Jeder Tag muss geschlossen werden! Ausnahmen bilden die Standalone-Tags (z. B. `<br>` oder `<hr>`).

```
<h1> Überschrift der 1. Ordnung </h1>
<h6> Überschrift der 6. Ordnung </h6>
```

```
<br>   ← eine Zeilenschaltung
<hr>   ← horizontale Linie
```

```
<p> Absatz </p>
```

Tags können auch verschachtelt werden.

```
<p>Das hier ist <i>besonders Wichtig</i> und <b>muss hervorgehoben</b> werden!</p>
```

Im Browser wird folgendes dargestellt:
Das hier ist *besonders Wichtig* und **muss hervorgehoben** werden!

```
<i> kursive Schrift </i>
<b> Fettschrift </b>
<sub> tiefgestellt </sub>
<sup> hochgestellt </sup>
<s> durchgestrichen </s>
```

Nicht nummerierte Liste
(unordered list)

Beispiel:

- HTML lernen
- HTML anwenden
- CSS lernen

Nummerierte Liste
(ordered list)

Beispiel:

1. Handout lesen
2. Im Internet suchen
3. Fragen stellen

```
<ul>
  <li>HTML lernen</li>
  <li>HTML anwenden</li>
  <li>CSS lernen</li>
</ul>
```

```
<ol>
  <li>Handout lesen</li>
  <li>Im Internet suchen</li>
  <li>Fragen stellen</li>
</ol>
```

`<ul>` ← ist das Elternelement
`<li>` ← ist das Kindelement

`<li>` ← bedeutet List-Item

`<ul type="circle">`
Aufzählung mit nicht ausgefüllten Kreisen

`<ul type="square">`
Aufzählung mit Rechtecken

Achtung: Das Type-Attribut im `<ul>` Tag wird unter HTML5 nicht mehr unterstützt. Die meisten Browser zeigen es aber trotzdem an.

`<ol type="I">`
Nummerierte Liste mit Römischen Zeichen.
Weitere Werte für das type-Attribut:

- `1` … Arabische Zahlen
- `A` … Großbuchstaben
- `a` … Kleinbuchstaben
- `i` … kleine Römische Zeichen

`<ol start="3">`
Startet die nummerierte Liste mit 3.

`<ol reversed>`
Die Reihenfolge wird umgedreht!

Bei `type=""` im `<ul>` oder `<ol>` Tag handelt es sich um ein Attribut. Attribute bestimmen im HTML-Element zusätzliche Eigenschaften/Informationen. Sie stehen immer im Starttag und werden in der Regel als `Name = "Value"` verwendet. (`Name` ist die Bezeichnung des Attributs, `Value` ist der Wert)

Web-Tipp: Der HTML Validator vom W3C
http://validator.w3.org ← den eigenen HTML Code auf Fehler überprüfen

Übung A: TXT in HTML

Öffne die Datei ‚**LW_Salzburg_2013.txt**' mit einem Texteditor und speichere diese als ‚**LTW_Salzburg_2013.html**' ab.

- ❑ Füge um den Text das HTML-Grundkonstrukt ein.

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <title>Landtagswahl Salzburg 2013</title>
    <meta name="autor" content="Vorname Nachname">
    <meta name="date" content="jjjj-mm-tt">
  </head>
 <body>
    {Der Inhalt von LW_Salzburg_2013.txt}
 </body>
</html>
```

- ❑ Weise den Überschriften die passenden Tags zu. `<h1>` bis `<h3>`
- ❑ Verteile `<p>` Tags im Text um Absätze zu bestimmen.
- ❑ Die Aufzählung mit den Parteien soll als "nicht nummerierte Liste" (unordered list) dargestellt werden. `<ul>` und `<li>`

Folgende Parteien traten zur Wahl an:

- BZÖ - Bündnis Zukunft Österreich
- FPÖ - Freiheitliche Partei
- Grünen - Die Alternativen
- KPÖ - Kommunistische Partei Österreich
- ÖVP - Österreichische Volkspartei
- PIRAT - Piratenpartei
- SPÖ - Sozialdemokraten
- TS - Team Stronach

- ❑ Die Liste mit den Verteilungen der Sitze soll als "nummerierte Liste" (ordered list) dargestellt werden! `<ol>` und `<li>`

Verteilung der Sitze

Nach der Wahl wurden die Sitze im Landtag wie folgt verteilt:

1. ÖVP - 11 Sitze (Schwarz)
2. SPÖ - 9 Sitze (Rot)
3. Grüne - 7 Sitze (Grün)
4. FPÖ - 6 Sitze (Blau)
5. TS - 2 Sitze (Gelb)

- ❑ Alle Vor- und Nachnamen in dem Text sollen kursiv dargestellt werden. `<i>`
- ❑ Das Wort "Salzburg" soll immer fett geschrieben werden. `<b>`

2.2 Basis Elemente (img, a href)

Hyperlinks

HTML	`<a href="" target="" > … </a>`

```
<a href="https://www.google.at" target="_blank">Google Suche</a>
```

Hyperlinks sind Querverweise zu anderen Webseiten oder Dateien.

href="" Innerhalb der Anführungszeichen wird der Verweis eingetragen.
Ein absoluter Link ist ein Hyperlink zu einer fixen Adresse
z. B. **"https://www.google.at"**
Der relativer Link verweist abhängig vom Speicherort
z. B.: **"../html/grundbegriff.html"**

"mailto: john@google.at"	← mit E-Mail-Client öffnen
"skype: johndoe90"	← für Skype
"tel:+4301511452"	← für Telefonnummern

target="" Legt das Zielfenster für die Verweise fest.
_self öffnet im aktuellen Fenster (Standardwert)
_blank öffnet in neuem Fenster.
_parent Elternfenster
_top oberstes Fenster

Bilder

HTML	`<img src="" alt="" title="" height="" width="">`

```
<img src="../pix/bild.png" alt="Ein Logo" height="200">
```

Bilder werden mit dem **<img>** Tag eingefügt. Es handelt sich dabei um ein Standalone-Tag ohne Elementinhalt und ohne End-Tag!

src="" Verweis auf den Speicherort des Bildes.
(relativ oder absolut)

alt="" Beschriftung des Bildes.

title="" Zeigt einen Tooltip, wenn man mit der Maus darüber fährt.

height="" Höhe des Bildes (in Pixel) ohne einer Angabe wird die Originalhöhe des Bildes verwendet.

width="" Breite des Bildes (in Pixel) ohne einer Angabe wird die Originalbreite des Bildes verwendet.

Ein Hyperlink kann auch auf ein Bild gelegt werden:

```
<a href="https://www.google.at" >
  <img src="logo.png" alt="Logo">
</a>
```

Die wichtigsten Bildtypen im Web sind: ***.gif, *.png, *.jpg, *.svg**

Ü 2.2 Übungen Basis a href, img

Übung A: Ein Inhaltsverzeichnis

- ❑ Erstelle das HTML-Grundgerüst mit deinem Namen und dem aktuellen Datum. Der Titel ist Inhaltsverzeichnis.
- ❑ Schreibe ein Inhaltsverzeichnis! (wie neben dargestellt).
- ❑ Die Listen müssen verschachtelt sein!
- ❑ Du benötigst folgende Tags: `<h1>, <ol>, <ul>, <li>, <b>, <s>, <br>`
- ❑ **Als HTML-Dokument speichern!**

Inhaltsverzeichnis

I. **Einleitung**
Einige Worte zur Begrüßung
II. **Projektbeschreibung**
Ein kurzes Abstract über das Projekt
III. **Projektplanung**
a. Ziele setzen
b. Soll-Ist-Vergleich
c. Ziele überprüfen
IV. **Realisierung**
Das Projekt wird ausgeführt
V. **Checkliste**
- *Kopieren*
- *Verteilen*
- *Erklären*
- *Abhacken*

VI. ~~**Risikomanagment**~~

Übung B: Top-Ten Websites

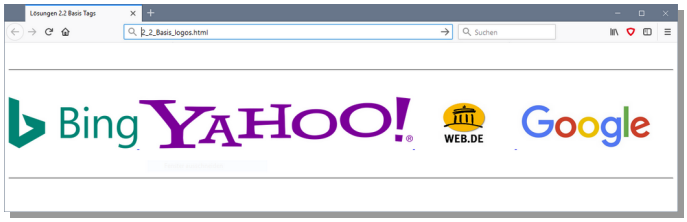
- ❑ Erstelle das HTML-Grundgerüst mit deinem Namen und dem aktuellen Datum. Der Titel ist Top-Ten.
- ❑ Schreibe eine "Top-Ten Liste" von beliebten Webseiten (wie neben dargestellt).
- ❑ Verwende hier Hyperlinks zu den Webseiten!
- ❑ Vollende die Liste bis Platz 1 mit deinen Lieblingsseiten im Netz!
- ❑ Du benötigst folgende Tags: `<h1>, <ol>, <li>, <a href>, <s>, <br>`
- ❑ **Als HTML-Dokument speichern!**

Top-Ten Websites

10. Selfhtml.org
9. HTML Code validieren
8. Google Suche
7. Facebook
6. ...

Übung C: Suchmaschinen

- ❑ Erstelle das HTML-Grundgerüst mit deinem Namen und dem aktuellen Datum. Der Titel ist Suchmaschinen.
- ❑ Suche im Internet nach den Logos von mind. vier Suchmaschinen.
- ❑ Die Logos sollen nebeneinander dargestellt werden (siehe unten).
- ❑ Mit einem Klick auf das Logo soll sich die jeweilige Suchmaschine in einem neuen Browserfenster öffnen.
- ❑ Du benötigst folgende Tags: `<img>, <a href>`
- ❑ **Als HTML-Dokument speichern!**



2.3 Best Practice

Dem Browser ist es relativ egal wie man den Code strukturiert. Zeilen- und Absatzschaltungen werden im Browser nicht dargestellt – dafür verwendet man **
** *oder* **<p>**. *Man kann theoretisch auch eine Website in nur einer Zeile schreiben:*

```
<body><div>Aufzählung<ul><li>Punkt 1</li><li>Punkt 2</li></ul></div></body>
```

Um nun aber den Code für den Menschen lesbarer zu machen, rückt man die Child-Elemente (Kindelemente) mit Leerzeichen ein:

```
<body>                          ← Parent (Elternelement)
  <div>Aufzählung               ← First Child (Kindelement) von <body>
    <ul>                        ← Child von <div>, Parent von <li>, Sibling von <p>
      <li>Punkt 1</li>          ← Child von <ul>
      <li>Punkt 2</li>
    </ul>
    <p>Etwas Text</p>           ← Second Child von <div>, Sibling (Geschwister) von <ul>
  </div>
</body>
```

Der Tag ist ein Container Element! (engl. division ⇔ Bereich).

Besondere Zeichen

Wenn man im Text Zeichen verwenden möchte, die in HTML eine Bedeutung haben (<, >, &, "), dann verendet man HTML-eigene Zeichen. Beispiele dafür sind:

<	*öffnende Spitze Klammer (lower than)*	**<**
>	*schließende Spitze Klammer (greater than)*	**>**
&	*Kaufmännisches Und (Ampersand)*	**&**
"	*Anführungszeichen oben*	**"**
	geschütztes Leerzeichen (non-breaking space)	** **

Ein umfangreiche Zeichenreferenz findet man bei selfthml.org
https://wiki.selfhtml.org/wiki/Zeichenreferenz

HTML Kommentare

Der Browser ignoriert Kommentare und stellt diese auch nicht dar. Sie dienen dem Autor um innerhalb des Codes Anmerkungen zu platzieren oder Codeteile auszukommentieren. Eingeleitet wird ein Kommentar mit **<!--** *dann folgt der Kommentartext. Abgeschlossen wird es mit* **-->**

```
<!-- einzeiliger Kommentar -->
<p>viel Text</p>
<!-- und das ist ein mehrzeiliger Kommentar zu dem Text mit <p>...</p>
     Letzte Zeile des Kommentars -->
```

3.1 Tabellen

HCL031
Testtabelle.html

Man verwendet zur übersichtlichen Darstellung von Daten Tabellen. Diese sind in Zeilen (waagerecht) und Spalten (senkrecht) gegliedert. Am Schnittpunkt einer Zeile und einer Spalte ist eine Zelle. HTML-Tabellen werden nur verwendet, um tabellarische Daten darzustellen, nicht, um das Layout einer Website zu gestalten.

```
<table>
   <thead>
      <tr> <th> </th> <th> </th> <th> </th> </tr>
   </thead>
   <tbody>
      <tr> <td> </td> <td> </td> <td> </td> </tr>
      <tr> <td> </td> <td> </td> <td> </td> </tr>
   </tbody>
</table>
```

<table>	leitet eine Tabelle ein
<thead>	Der Tabellenkopf ist für eine bessere Gliederung angedacht. Der Tag wird für die Beschriftung einer Tabelle verwendet und ist für die Darstellung nicht zwingend notwendig.
<tbody>	Der Tabellenkörper ist ebenfalls für eine bessere Gliederung der Inhalte und kann genauso wie der **<thead>** weggelassen werden.
<tr>	leitet eine Tabellenzeile ein (tr = table row = Tabellenzeile)
<th>	eine Zelle im Tabellenkopf (th = table header)
<td>	eine normale Datenzelle (td = table data)

Damit unsere Tabellen sichtbar werden, schreiben wir in den **<table>** *Tag das Attribut* **border="2"**. *Achtung: Diese Vorgehensweise ist in HTML5 obsolet da Rahmen mit CSS definiert werden müssen!*

Die Zellen lassen sich selbstverständlich auch verbinden. Dies wird mit den Attributen **colspan** bzw. **rowspan** im **<td>** Tag umgesetzt.

<td colspan="3">	**colspan** (Spalte überspannen) erlaubt eine Tabellenzelle nach rechts über mehrere Spalten auszudehnen (hier über 3 Zellen)
<td rowspan="2">	**rowspan** (Zeile überspannen) verbindet Tabellenzellen über mehrere Zeilen (hier über 2 Zellen)

```
<table border="2">
  <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr>
  <tr> <td> </td> <td colspan="3"> </td> </tr>
  <tr> <td rowspan="2"> </td> <td> </td> <td> </td> <td> </td> </tr>
  <tr> <td> </td> <td> </td> <td> </td> </tr>
</table>
```

Ü 3.1 Übungen Tabellen

Übung A: Stundenplan

- Erstelle mit dem **<table>** Element deinen Stundenplan.
- Im Tabellenkopf sind folgende Beschriftungen anzuführen:
  Stunde | von | bis | Montag | Dienstag | Mittwoch | Donnerstag | Freitag | Samstag
- Ändere die Unterrichtszeiten (wenn nötig) und verbinde die Zellen für die Pausen mit **colspan**
- **Trage deine Unterrichtsfächer in die freien Zellen ein.**

Stunde	von	bis	Montag	Dienstag	Mittwoch	Donnerstag	Freitag	Samstag
1.	07^{45}	08^{35}						
2.	08^{35}	09^{25}						
	09^{25}	09^{40}	Pause (15 Minuten)					
3.	09^{40}	10^{30}						
4.	10^{30}	11^{20}						
	11^{20}	11^{30}	Pause (10 Minuten)					
5.	11^{30}	12^{20}						
6.	12^{20}	13^{10}						
	13^{10}	13^{15}	Pause (5 Minuten)					
7.	13^{15}	14^{05}						
8.	14^{05}	14^{55}						
	14^{55}	15^{05}	Pause (10 Minuten)					
9.	15^{05}	15^{55}						
10.	15^{55}	16^{45}						
	16^{45}	17^{35}						

Übung B: Ländervergleich

- ❑ Erstelle mit dem **<table>** Element einen Ländervergleich (Schweiz, Österreich, Deutschland) wie unten dargestellt.
- ❑ Durch einen Klick auf die Landesflagge soll sich die jeweilige Wikipedia-Seite zum passenden Land öffnen.
- ❑ **Füge noch ein weiteres Land deiner Wahl dazu!**

Land	Schweiz	Österreich	Deutschland
Amtssprache	Deutsch	Deutsch	Deutsch
Hauptstadt	Bern	Wien	Berlin
Fläche	41 285 km^2	83 878 km^2	357 385 km^2
Einwohner	8,4 Mio	8,7 Mio	82,5 Mio
Staatsform	föderale Republik	Bundesrepublik	Bundesrepublik
Währung	Schweizer Franken	Euro	Euro
Internet-TLD	.ch, .swiss	.at	.de
Telefon-Vorwahl	+41	+43	+49

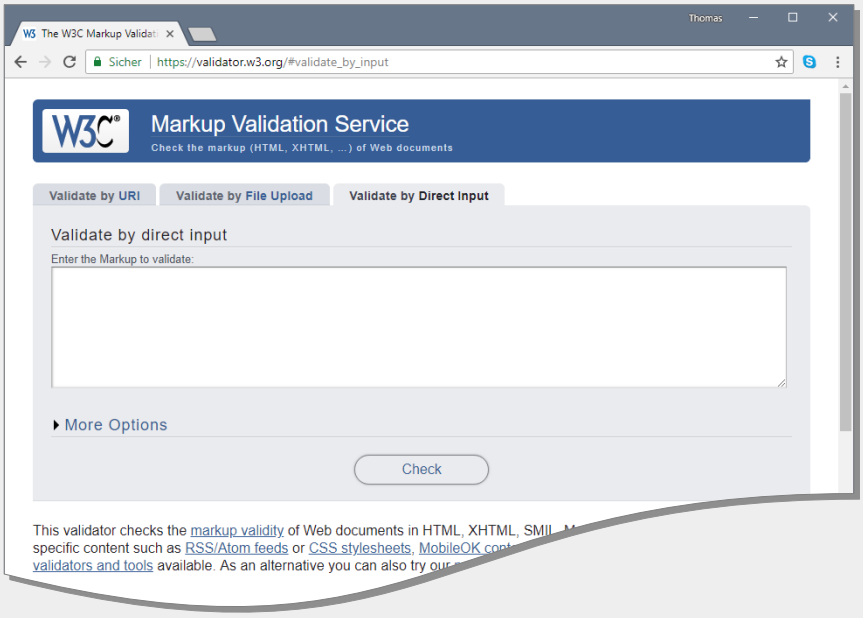
3.2 HTML validieren

HTML Code wird in der Regel von einem Browser interpretiert. Jeder Browser hat seine Eigenheiten – Firefox ist nicht Chrome – Chrome ist nicht Edge usw. Manche Elemente oder Attribute gibt es nur für bestimmte Browser. Wir wollen aber für alle schreiben. Wie können wir nun sicher sein, das unser Code richtig ist? Dafür gibt es das W3C. Das W3C Konsortium entwickelt Standards und Richtlinien für das Web.
Die meisten Browser-Entwickler halten sich an diese Richtlinien.

Wie können wir nun aber sicher sein, dass unser Code richtig ist?

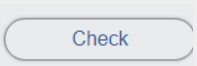
Dafür hat das W3C auf seiner Website ein **Markup Validation Service** (Prüfungsdienst, Validator) eingerichtet.

Auf **www.w3.org** findet man auch weitere Prüfdienste wie z. B. für CSS oder einen Link Checker.
Der HTML-Validator bietet drei Möglichkeiten an um den HTML Code zu überprüfen.

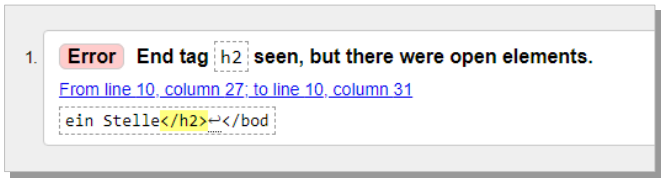


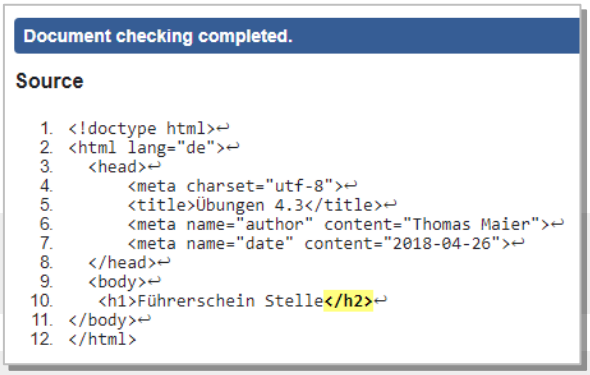
https://validator.w3.org

Validate by URI.....Man gibt die Adresse zu einem Online Dokument an.
Validate by File Upload.....Hier ladet man das HTML Dokument hoch.
Validate by Direct Input.....Man kopiert den HTML Code ins Textfeld.



Nach einem Klick auf Check wird der Code überprüft und man bekommt einen umfangreichen Bericht.





Error *bedeutet Fehler und muss unbedingt korrigiert werden!*

Übung A: HTML Code validieren

- ❑ Überprüfe den HTML Code von der **Übung 2.2 A: Ein Inhaltsverzeichnis**.
- ❑ Tipp: Jeder Browser kann den Quelltext anzeigen.
- ❑ Erstelle einen Screenshot vom Prüfergebnis.

4.1 style, span, color

HCL041

Bislang hatten unsere HTML Seiten nur das Standard Erscheinungsbild des Browsers. Um nun einem Element ein besonderes Design zu verleihen, verwendet man das **`style`** *Attribut. Das* **`style`** *Attribut gilt für nahezu alle HTML Tags und hat folgende Schreibweise:*

CSS Styles

HTML	`<tagname style="property: value;">`
	**`property`** ist der CSS-Befehl (Eigenschaft), **`value`** ist der Wert. Wichtig ist das Semikolon (;) am Ende jeder CSS Anweisung. Im folgenden Beispiel wird die Schriftfarbe des Textes eines <p> Tags auf die Schriftfarbe rot gesetzt.
CSS	**`color: [Wert];`**

```
<p style="color: red;">Die Schrift ist jetzt rot</p>
```

Man kann auch mehrere CSS Anweisungen (Stile) mit dem **`style`** *Attribut zuweisen.*

CSS	**`background-color: [Wert];`** ← Hintergrundfarbe (siehe Farben unten)

```
<p style="color: white; background-color: blue;">
     Der Schnee ist weiß und der Himmel ist blau.</p>
```

HTML	**`<span> ... </span>`**
	Wenn man nur ein Wort oder eine Textstelle auszeichnen möchte, benutzt man das **`<span>`** Element (span = engl. überspannen). Der **`<span>`** Tag alleine bewirkt nichts – erst in Kombination mit CSS erfüllt er eine Aufgabe!

```
<h1>Die Lehre von <span style="color: green;">Energie</span> und
    <span style="color: red;">Wärme</span></h1>
```

Farben

CSS	**`color: [Wert];`**		
	Um den **`WERT`** (value) für Farben festzulegen, gibt es 3 wichtige Möglichkeiten:		
	1. Farbname	Die Namen werden in Englisch ausgeschrieben	`color: DeepSkyBlue;`
	2. RGB	RGB (Rot, Grün, Blau) ist das Standard Farbmodell für Bildschirme. Die einzelnen Werte reichen von 0 bis 255.	`color: rgb(0, 191, 255);`
	3. HEX	Hier werden die RGB Werte Hexadezimal dargestellt.	`color: #00bfff;`

Weitere Farbmodelle sind HSL, HWB und CMYK. CMYK (Cyan, Magenta, Yellow und Schwarz) ist ein Farbmodell für den Druck.

4.2 Rahmen

Mit der **`border`** *Eigenschaft (CSS) kann man den Rahmen um ein Element bestimmen.*
`border` *hat drei Werte:* ***Rahmendicke, Rahmentyp*** *und* ***Rahmenfarbe****!*
Zwischen den Werten wird ein Leerzeichen gesetzt. Am Ende ein Semikolon (;).

CSS **`border: [dicke] [typ] [farbe];`**

[dicke] *Die Rahmendicke kann in Pixel angegeben werden (z. B:* **`5px`***) bzw.* **`thin`** *(dünn)* **`medium`** *(mittel) oder* **`thick`** *(dick). Die Rahmendicke alleine reicht noch nicht für eine Darstellung – es muss der Rahmentyp angegeben werden!*

`[typ]` *Folgende Rahmentypen kennt CSS:*

`solid`	*fester Rahmen, ohne besondere Erscheinung*
`dotted`	*gepunkteter Rahmen*
`groove`, **`ridge`**, **`inset`**, **`outset`**	*simuliert einen 3D Effekt!*
`double`	*doppelter Rahmen*
`dashed`	*gestrichelter Rahmen*

Das Beispiel erzeugt einen doppelten Rahmen mit einer Rahmendicke von 8 Pixel um das **`<h1>`** Element.

```
<h1 style="border: 8px double;">Kapitel 16 …</h1>
```

[farbe] *Nun kann man die Eigenschaft noch um einen Farbwert (siehe 4.1) ergänzen. (*`#00f` *ist der Hex-Wert für Blau).*

```
<h1 style="border: 10px dotted #00f;">Deutsche Automarken</h1>
```

Man kann auch einzelne Rahmen bestimmen:

`border-top` *für den oberen Rahmen.* **`border-bottom`** *für den unteren Rahmen.* **`border-left`** *für den linken und* **`border-right`** *für den rechten Rahmen.*

```
<p style="border-bottom: 5px solid red;">Unser Jahrbuch</p>
```

Abgerundete Ecken

CSS **`border-radius: [li-oben] [re-oben] [re-unten] [li-unten];`**

Der Wert wird in Pixel angegeben. Gibt man nur einen Wert an, dann werden alle vier Recken gleichmäßig abgerundet. Die Ecken werden im Uhrzeigersinn definiert.

```
<h2 style="border: 10px solid rgb(255,126,0);
           border-radius: 25px 0px 25px 0px;
           background-color: blue;
           color: white;">... in einem fernem Land <br>
                           ... vor gar nicht allzu langer Zeit!</h2>
```

4.3 Höhe und Breite

Die CSS Eigenschaften **`height`** *(Höhe) und* **`width`** *(Breite) bestimmen die Größenangaben eines HTML Elements.*

CSS

`height: [Wert]; width: [Wert];`

[Wert] Für die Größenangaben gibt es in CSS eine breite Palette an Möglichkeiten. Wir unterscheiden zwischen absoluten und relativen Längenmaßen.

Absolute Längenmaße *sind fixe Größe und ändern sich in der Regel nicht!*

`px`	Pixel	Bildpunkte des Screens (Bildschirm, Smartphone udgl.)
`cm`	Zentimeter	Entspricht ca. 37,8 Pixel
`mm`	Millimeter	Entspricht ca. 3,78 Pixel, bzw. 0,1 cm
`in`	Zoll	Ein Zoll sind 2,54 cm oder 96 Pixel
`pt`	Punkt	Werden für Schriftgrößen verwendet. 1 pt entspricht ca. 1,33 Pixel

```
<p style="background-color: blue; width: 4cm;"> </p>
<p style="background-color: red; width: 400px;"> </p>
```

Relative Längenmaße *beziehen sich auf die Größe eines Elements (meist dem Elternelement). Wird die Größe verändert – z. B. Browserfenster verkleinern/vergrößern, dann verändert sich auch die Größe des Kind Elements – und zwar relativ.*

`%`	Prozent %	z. B. **`width: 50%;`**
`em`	em	Vertikale Größe einer Schrift. Praktisch für Schriftarten.
`rem`	Wurzel-Em	Relativ zum Wurzelelement (<html>).
`vw`	Viewport-Breite	entspricht dem 100. Teil des Anzeigebereichs.
`vh`	Viewport-Höhe	entspricht dem 100. Teil der Höhe des Anzeigebereichs. **`height: 25vh;`** hat ¼ der Höhe des Screens.

```
<div style="border: 5px solid; height: 50vh;"> </div>
```

Das Beispiel oben zeigt das Container-Element **`<div>`** mit der halben Höhe des Browser-Fensters an. Verändert sich die Browsergröße (nicht jeder Browser wird in Vollbild gestartet), dann verändert sich auch die Höhe des **`<div>`** Elements.

Relative Größen haben das Problem, dass sie entweder zu groß oder zu klein sind wenn sich die Bezugsgröße verändert. Dafür gibt es die Eigenschaften:

CSS

`max-height: [Wert]; max-width: [Wert];`

Sie beschreiben die maximale Breite oder Höhe. z. B. eine relative Breite von 50% endet bei 300 Pixel, wenn man **`max-width: 300px;`** *definiert hat.*

`min-width` *und* **`min-height`** *geben jene Größen an, die auf jeden Fall dargestellt werden sollen. Dieses Minimum darf nicht unterschritten werden!*

```
<div style="background-color: green;
            width: 50%;
            min-width: 200px;"> </div>
```

Ü 4.3 Übungen Farben, Rahmen und Größen

Übung A: Loveparade

- ❑ Erstelle nur mit HTML und CSS (`style=""` Attribut) einen Loveparade-Schriftzug wie unten dargestellt.
- ❑ Bilddateien sind **nicht** erwünscht!
- ❑ Verwende nur ein einziges `<h1>` Element.
- ❑ Jeder Buchstabe hat eine eigene Hintergrundfarbe.
- ❑ **TIPP: Verwende `<span>` Tags!**

L O V E P A R A D E

Übung B: Farbtabelle

- ❑ Erstelle nur mit HTML und CSS eine Farbtabelle.
- ❑ **Ermittle die RGB und HEX Werte der fehlenden Farben und trage Sie in die Tabelle ein!**
  (Türkis, Blaugrün, Himmelblau, Magenta)

Farbname	R	G	B	HEX
Gold	255	215	0	#FFD700
Gelbgrün	154	205	50	#9aCd32
Türkis				
Blaugrün				
Himmelblau				
Magenta				

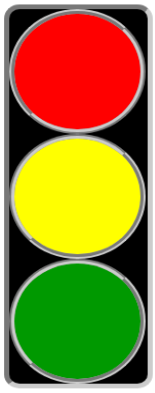
Übung C: Einfache Buttons

- ❑ Designe drei unterschiedliche Button!
- ❑ FREIES Design
  (alles ist erlaubt, solange es nur in HTML und CSS ist, keine Bilder)
- ❑ **Die Buttons haben noch keine Funktion – noch nicht!**

Bestellung absenden | Bestellung abbrechen | Bestellung neu erstellen

Übung D: Eine Ampel

- ❑ Erstelle nur mit HTML und CSS eine Verkehrsampel.
- ❑ Die Größe der Ampel wird durch die Breite des Browsers bestimmt.
  (z. B. wird der Browser in der Breite größer, dann soll die Ampel auch größer werden – sie skaliert).
- ❑ **TIPP: Verwende `border-radius` und relative Größenangaben!**



5.1 CSS im Head

Bisher haben wir unsere CSS Anweisungen direkt in das HTML Element geschrieben. Praktischer ist es jedoch, die CSS Anweisungen innerhalb des **<head>** *Tags zu schreiben. Dafür öffnet man mit* **<style>** *und schließt dann mit* **</style>.** *Für die Zuweisung der CSS Stile gibt es drei Möglichkeiten (drei Selektoren: Typ, ID und Klasse).*

```
<head>
  <style>
    h1 {border-bottom: thin solid black;}
    #hauptinhalt {background-color:#CFD5EB; color:#00C;}
    .wichtig {color:red;}
  </style>
</head>
```

1. Typselektor: Ein HTML Element überschreiben

Man definiert für einen Tag ein neues Aussehen. Jedes mal wenn dieser Tag dann im HTML-Dokument verwendet wird, erhält er das Aussehen aus dem **<style>** *im* **<head>.**

CSS	TAG {CSS-Anweisungen;}

```
CSS im <head><style> ... </style></head>:
h2 {color:red;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
HTML im <body> ... </body>:
<h2>Empfehlungen</h2>
```

2. ID-Selektor: Eine ID festlegen für ein einziges Element

Eine ID (Identifikation) gilt nur für ein einziges HTML Element. Im gesamten Code darf die ID nur einmal vorkommen. Im CSS schreibt man eine Raute (#) vor dem Namen.
Ins HTML Element schreibt man das Attribut **id=" "**

CSS	#ID {CSS-Anweisungen;}

```
CSS im <head><style> ... </style></head>:
#haupt {background-color:blue;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
HTML im <body> ... </body>:
<p id="haupt">Empfehlungen</p>
```

3. Klassenselektor: Eine Klasse definieren für mehrere Elemente

Mit einer Klasse kann man beliebig viele Elemente ansprechen. Eine Klasse kann auch auf unterschiedliche Tags angewandt werden. Im CSS schreibt man einen Punkt vor den Klassennamen. Ins HTML Element schreibt man das Attribut **class=" "**

CSS	.CLASS {CSS-Anweisungen;}

```
CSS im <head><style> ... </style></head>:
.sr1 {color:red;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
HTML im <body> ... </body>:
<h1 class="sr1">Vorlagen</h1>
<span class="sr1">WICHTIG</span>
```

5.2 Schriftarten

Um eine Schriftart für ein Element zu bestimmen, verwenden wir **`font-family`**. *Der Browser greift auf die installierten Schriftarten des Empfänger-System zu. Da es sein kann, dass der Endbenutzer nicht jede Schriftart installiert hat, geben wir mehrere Schriftarten als Wert mit. Zusätzlich noch eine Generische Schriftart (allgemeine Schriftfamilie). Die Schriftnamen werden durch einen Beistrich getrennt. Wie bei allen anderen CSS Anweisungen endet* **`font-family`** *ebenfalls mit einem Semikolon (;). Besteht eine Schriftart aus mehreren Wörtern, dann wird sie unter Anführungszeichen (") geschrieben.*

CSS	`font-family: [Schriftname] ["Schrift 2"] [Generische-Schrift];`
	**`[Generische-Schrift]`** Folgende generische Schriftfamilien können ausgewählt werden: **`serif`** eine Schriftart mit Serifen (z. B. Times New Roman, Palatino) **`sans-serif`** ohne Serifen (z. B. **Univers**, Calibri) **`cursive`** Schreibschriftarten (z. B. Mistral, Vivaldi) **`fantasy`** ungewöhnliche Schriften (z. B. Impact Label, Rosewood) **`monospace`** alle Zeichen haben die gleiche Breite (z. B. Courier)

```
h1 {font-family:Tahoma, Geneva, sans-serif;}
```

Will man nun eine eigene Schriftart verwenden, dann benutzen wir die CSS Anweisung **`@font-face`**. *Innerhalb der* **`@font-face`** *Anweisung, wird der Name der Schriftfamilie, eine gültige URL zur Schriftart (wo ist diese abgespeichert?) und das Format der Schriftart definiert.*

```
@font-face {font-family: 'neue-schrift';
            src: url('pfad/zu/neueschrift.ttf') format('truetype'); }
```

Der Name der Schriftfamilie muss nicht immer **`'neue-schrift'`** *lauten. Es kann ein beliebiger Name gewählt werden – jedoch bitte kein Name einer schon bestehenden anderen Schriftart. Danach kann die Schriftart mit* **`font-family`** *zugewiesen werden!*

```
<p style="font-family: neue-schrift, Arial, sans-serif">
   ... Symbole erleichtern das Verständnis ...</p>
```

Achte auf die Dateigrößen einer Schriftart. Schriftarten von mehr als 300 KiB verlangsamen die Ladegeschwindigkeit. Nach der erstmaligen Abholung wird die Schriftart im Browsercache gespeichert.

Schriftarten sind durch Urheberrechte geschützt. Ohne eine Genehmigung (erworben/gekauft, CC, gratis Schriftarten) kann die Nutzung im Internet teuer werden. Manche Schriftarten sind technisch geschützt und werden im Browser nicht angezeigt.

Google bietet eine Vielzahl von freien und offenen Schriftarten auf ***https://fonts.google.com*** *an.*
Diese Schriftarten lassen sich praktisch und einfach in eine Webseite einbinden!

5.3 Schriftgrößen

CSS

`font-size: [Wert]`

Mit **`font-size`** bestimmt man die Darstellungsgröße der Schrift. Als Werte können alle Größe (siehe 4.3 Höhe und Breite) angewendet werden. Zusätzlich gibt es noch Schlüsselwörter:

Schlüsselwörter mit absoluten Angaben

`xx-small` ..... *winzig*	**`large`** ..... *groß*
`x-small` ..... *sehr klein*	**`x-large`** ..... *sehr groß*
`small` ..... *klein*	**`xx-large`** ..... *rießig*
`medium` ..... *mittel*	

Schlüsselwörter mit relativen Angaben

`smaller` ..... *Kleiner als das Elternelement*
`larger` ..... *Größer als das Elternelement*
`inherit` ..... *Die gleiche Größe wie im Elternelement*

Über die Schlüsselwörter entscheidet der Browser selbstständig in welcher Größe die Schrift dargestellt werden soll.

```
        h2 {font-size: 45px;}
#haupttext {font-size: medium;}
     .tipp {font-size: larger;}
      span {font-size: 1.2em;}
```

CSS

`font-weight: [Wert];`

Mit **`font-weight`** wird die Strichstärke (Dicke) des Textes festgelegt. Folgende Werte sind möglich:

`lighter` ..... dünner als im Elternelement
`normal` ..... normale Strichstärke
`bold` ..... fett
`bolder` ..... fetter als im Elternelement
`100,200,300 … 900,` ..... extra-dünn (**`100`**) bis extrafett (**`900`**)
`inherit` ..... Strichstärke des Elternelements

```
   .fetter {font-weight: bolder;}
#impressum {font-weight: 400;}
```

CSS

`font-variant: [Wert];`

*Mit **`font-variant: small-caps;`** kann man einen Text mit Kapitälchen darstellen. Kapitälchen sind Großbuchstaben in der Höhe von Kleinbuchstaben.*

Beispiel ohne Kapitälchen: Was it the cat i saw?
Beispiel mit Kapitälchen: **WAS IT THE CAT I SAW?**

```
<p>Was it the cat i saw?</p>
<p style="font-variant: small-caps;">Was it the cat i saw?</p>
```

5.4 Text Dekoration

Mit der **`text-decoration`** *Eigenschaft kann man einen Text über-, unter- oder durchstreichen. Zusätzlich können Stiel und Farbe der Linien definiert werden! Für Color-Angaben gelten die allgemeinen Regeln (siehe* ***4.1 style, span, color).***

CSS

```
text-decoration: [line] [style] [color];
```

[line]

`underline`.....*unterstrichen*
`overline`.....*überstrichen*
`line-through`.....*durchgestrichen*
`none`.....*keine Textdekoration*

[style]

`solid`.....*durchgezogene Linie*
`dotted`.....*gepunktete Linie*
`wavy`.....*Wellenlinie*
`double`.....*doppelte Linie*
`dashed`.....*gestrichelte Linie*

```
.falsch {text-decoration: underline wavy red;}
```

```
<h1 style="text-decoration: underline;">Fehlerquellen</h1>
<p>Standard wird oft mit einem t geschrieben
<span class="falsch">(Standart)</span></p>
```

Hyperlinks (`a href`) werden vom Browser meist blau mit einer durchgezogenen Linie dargestellt. Will man die Unterstreichung von den Links entfernen, dann kann das mit **`text-decoration: none;`** *umgesetzt werden.*

```
a {text-decoration: none;}
```

Design Tipp: *Achte darauf, ob die Unterstreichung die Unterlänge des Textes nicht übermalt. Buchstaben mit Unterlänge sind: g, j, y usw.*

Textschatten

Mit der **`text-shadow`** *Eigenschaft fügt man dem Text einen Schatten hinzu.*

CSS

```
text-shadow: [H] [V] [blur] [color];
```

[H]	Die Horizontale Position des Schattens. (Negative Werte sind erlaubt)
[V]	Die vertikale Position des Schattens. (Negative Werte sind erlaubt)
[blur]	Radius der Unschärfe (Weichzeicheneffekt).
[color]	Farbangabe

```
h1 {text-shadow: 5px 5px 10px blue;}
h2 {color: white; text-shadow: 2px 2px 4px #000000;}
```

```
<h1>Moderne Medizin</h1>
<h2>Die chinesische Medizin im Überblick</h2>
```

Übung A: Einfache Buttons im Head

- ❑ Öffne deine Arbeit/Lösung von **4.3 C: Einfache Buttons**
- ❑ Schreibe die Arbeit so um, dass alle CSS Anweisungen im Head sind!
- ❑ **Solltest du 4.3 C noch nicht gelöst haben, dann erstelle drei Buttons mit CSS Anweisungen im Head.**

Übung B: Buttons im Head

- ❑ Öffne deine Lösung von **5.4 A: Einfach Buttons im Head.** (siehe oben)
- ❑ Erstelle einen weiteren Button – dieser soll eine außergewöhnliche Schriftart haben. (also keine Standard Schriftart).
  - Suche dafür im Internet nach einer **freien** Schriftart die du downloaden kannst und die dir gefällt. Beachte die Urheberrechte!
  - Binde die neue Schriftart mit CSS ein. Tipp: `@font-face`
  - **Der neue Button soll als Klasse verfügbar sein und vier mal dargestellt werden. Tipp:** `class=" "`

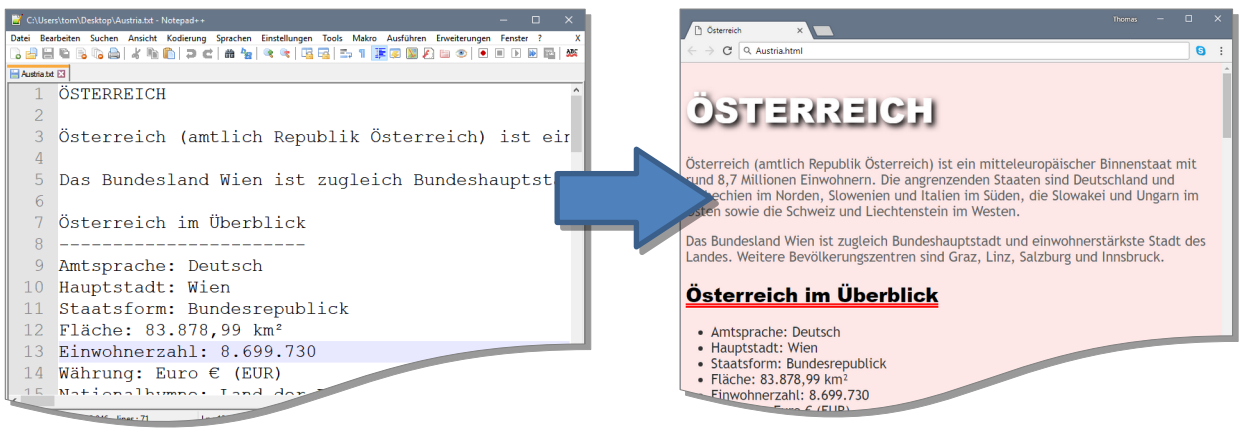
TERMINE DISCOTIME KONTAKT

Übung C: Google Fonts

- ❑ Finde heraus, wie man die Schriftarten von font.google.com nutzen kann!
- ❑ **Dokumentiere es schriftlich. (z. B. mit Word, PowerPoint, HTML …)**

Übung D: Österreich

- ❑ Öffne die Austria.txt
- ❑ Kopiere den Inhalt der **Austria.txt** in den **`<body>`** eines neuen HTML Dokuments.
- ❑ Formatiere den Text bzw. das Dokument mit CSS (Freie Übung).
- ❑ **Speichere die fertige Arbeit als `Austria.html`**



6.1 Ausrichtung

Natürlich muss der Text nicht immer Linksbündig sein. Mit der **`text-align`** *Eigenschaft (CSS) kann der Text (ebenso andere inline-Element, z.B.* **`<img>`***) auch Zentriert, Rechtsbündig oder als Blocksatz dargestellt werden.*

CSS

`text-align: [Wert];`

Für **[Wert]** können folgende Werte eingesetzt werden:

`left`	Linksbündig
`center`	Zentriert
`right`	Rechtsbündig
`justify`	Blocksatz (mit **`start`** und **`end`** kann man noch das Verhalten der letzten Zeile bestimmen).

```
<h1 style="text-align:center;
           border-bottom:3px solid;">Mitteilung</h1>
<p style="text-align:right; font-size:smaller">by John Doe</p>
```

Einen ***Erstzeileneinzug*** *erzeugt man mit* **`text-indent.`** *Als Wert [value] sind positive oder negative absolute Längenangaben, oder prozentual % relativ zur Breite möglich.*

CSS

`text-indent: [Wert];`

```
<p style="text-indent: 5%;" >Auf der Registerkarte …</p>
```

Spalten

Besonders viel Text lässt sich mit Spalten übersichtlicher gestalten – besonders wenn es zu einer Darstellung der Webseite auf breiten Monitoren kommt.

CSS

`columns: [count] [width];`

`[count]`	Wie viele Spalten sollen angezeigt werden? Eine positive Zahl oder **`auto`**. z. B. **`2`** für 2 Spalten.
`[width]`	Eine Längenangabe für die Mindestspaltenbreite. z. B. 6em, 500px,

Zwischen den zwei Werten wird ein Leerzeichen gesetzt.

```
.absatz {columns:3 7em; text-align:justify;}
```

Weitere CSS Eigenschaften in Verbindung mit Spalten (columns):

`column-gap`	*bestimmt den Abstand zwischen den Spalten.*
`column-rule`	um eine Linie zwischen den Spalten zu bestimmen.

CSS

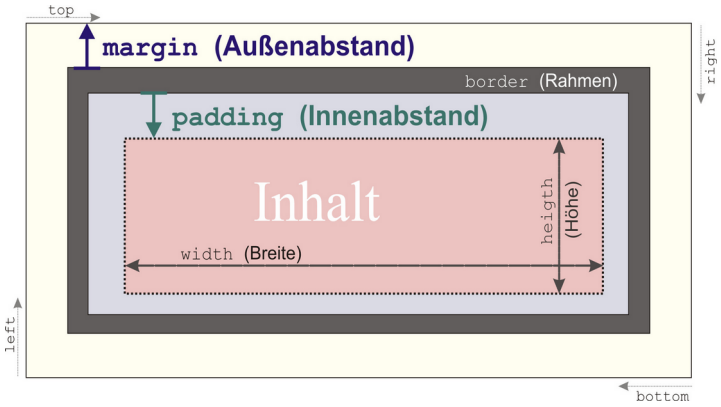
`column-rule: [width] [style] [color];`

`[width]` Linienstärke, **`[style]`** alle border-style-Werte,
`[color]` alle Farben.

```
.absatz {columns: 2; column-gap: 1.8em; column-rule: 5px dotted red;}
```

6.2 Abstände

*Im "klassischen" Boxmodell wird eine Box (rechteckiges HTML Element z. B. ** ** oder **<h1>**) durch seine Höhe und Breite, durch den Innen- und Außenabstand sowie durch den Rahmen bestimmt. Mit der CSS Eigenschaft **margin** bestimmt man den Außenabstand – mit **padding** den Innenabstand. Beide beziehen sich auf den Rahmen (auch wenn kein Rahmen definiert wurde!)*



CSS **padding: [top] [right] [bottom] [right];**

[top]	Innenabstand oben. (auch **padding-top**:).
[right]	Innenabstand rechts (auch **padding-right**:).
[bottom]	Innenabstand unten (auch **padding-bottom**:).
[left]	Innenabstand links (auch **padding-left**:).

Als Werte sind alle numerischen Längenmaße erlaubt (z. B. cm, px, em). Negative Werte sind nicht erlaubt. Gibt man nur einen Wert an, dann gilt dieser für alle vier Seiten.

```
h1 {padding-left: 3em;}
td {padding: 10px 5px 5px 10px;)
```

*Der Außenabstand (**margin**) funktioniert gleich wie **padding**. Man kann den Außenabstand auch einzeln ansprechen (**margin-top**, **margin-right** usw.). Zusätzlich erlaubt die **margin** Eigenschaft noch den Wert **auto** und negative Längenmaße.*

CSS **margin: [top] [right] [bottom] [right];**

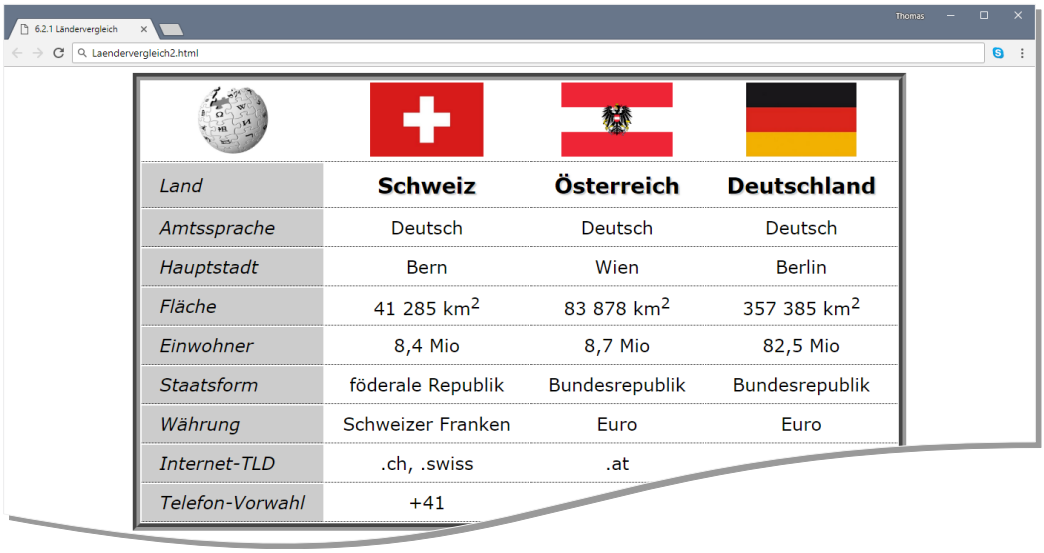
```
.wichtig {margin: 10px 4px 10px 5px; border: 2px solid;}
      h2 {margin: -10px;}
```

*Um ein Box-Element zentriert dazustellen muss man den **margin** Wert auf **auto** setzten und eine Breite angeben!*

```
.dieBox {margin:auto; width:600px;
         background:#FC0; height:3em; border:2px solid; }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<h1 class="dieBox">Wir verwenden Cookies!</h1>
<div class="dieBox" style="text-align: right;">Ist das Okey?</div>
```

Übung A: Ländervergleich

- ❑ Öffne deine HTML-Arbeit **3.1 B: Ländervergleich** (Tabellenübung).
- ❑ Die Ländernamen sollen hervorgehoben sein (Font-weight, Font-size).
- ❑ Die Zellen zu den Ländern sollen zentriert sein – die Beschriftungszellen (Land, Amtssprache, Hauptstadt usw.) bleiben linksbündig.
- ❑ Zwischen den Zeilen soll ein "Rahmenstrich unten" sein.
- ❑ Die Beschriftungszellen sollen einen grauen Hintergrund haben.
- ❑ Ändere die Schriftart und Schriftgröße der Tabelle (z. B. Arial, Verdana).
- ❑ Füge einen Innenabstand für Zellen hinzu.
- ❑ Zentriere die gesamte Tabelle über die Browserbreite! Tipp: `margin`
- ❑ **Die Tabelle soll einen dicken groovie Rahmen haben.**



Land	Schweiz	Österreich	Deutschland
Amtssprache	Deutsch	Deutsch	Deutsch
Hauptstadt	Bern	Wien	Berlin
Fläche	41 285 km^2	83 878 km^2	357 385 km^2
Einwohner	8,4 Mio	8,7 Mio	82,5 Mio
Staatsform	föderale Republik	Bundesrepublik	Bundesrepublik
Währung	Schweizer Franken	Euro	Euro
Internet-TLD	.ch, .swiss	.at	
Telefon-Vorwahl	+41		

Übung B: Stundenplan

- ❑ Öffne deinen Stundenplan (siehe **3.1 A: Stundenplan**).
- ❑ **Verschönere ihn mit CSS – freie Kreativübung.**

Übung C: Text mit Spalten

- ❑ Öffne die Datei `Wordelemente.html`
- ❑ Lege um den gesamten Text ein `<div>` Element mit der `id="Hauptdiv".`
- ❑ Das Hauptdiv hat eine Breite von 75%, schwarzen Hintergrund und weiße Schrift. Einen Innenabstand (`padding`) von 2em.
- ❑ !!! Das Hauptdiv soll ganz **Rechts am Browserrand** ausgerichtet werden. !!!
- ❑ Der Seitenrand vom `body` soll entfernt werden.
- ❑ Alle Absätze sind linksbündig ausgerichtet und haben einen Erstzeileneinzug von 3em.
- ❑ **Der zweite Absatz (Lorem ipsum dolor sit …) soll mit drei Spalten und als Blocksatz dargestellt werden.**

6.3 Best Practice

Bisher haben wir unsere CSS Anweisungen entweder im **<head>** *oder im HTML Element selbst geschrieben. Was aber, wenn man nicht nur eine Webseite hat, sondern viele? Natürlich will man, dass jede Seite das gleiche Design erhält. Dafür kann man CSS-Dateien (externe Stylesheets) einbinden. Externe Stylesheets haben die Endung* **.css** *und werden mit dem* **<link>** *Element im* **<head>** *eingebunden.*

HTML

<link rel="…" href="…" media="…">

rel="…"	Ist der Beziehungstyp. Wir verwenden **stylesheet** – das sagt dem Browser, dass es sich um CSS Eigenschaften handelt.
href="…"	die Adresse, wo die **.css** Datei zu finden ist. (relativ oder absolut) **href="https://www.css4.at/hauptdesign.css"** **href="../Design/Style.css"**
media="…"	Ausgabemedium (z.B. **screen** = Bildschirm, **print** = Druck)

In der externen **.css** *Datei, schreibt man gleich wie wir es von der* **<head>** *Lösung kennen.* **<style>** … **</style>** *ist nicht nötig. Nötig ist aber, eine Regel für unseren Zeichensatz (UTF-8). Eine Regel wird mit dem At-Zeichen (@) eingeleitet.*

```
<!doctype html>
<html lang="de">
  <head>
    <link rel="stylesheet" href="stylesheet.css" media="screen">
    ... </head>
~~~~~~~~~~~~~~ In der externen CSS Datei ~~~~~~~~~~~~~~~~~~~~~~~~~~~
      @charset "UTF-8";
      /* CSS Document */
      body {margin: 0px;}
```

Man kann auch mehreren Selektoren die gleichen Eigenschaften zuweisen. Diese werden dann durch einen Beistrich getrennt. In folgendem Beispiel wird der Klasse **.tipp** *und der Klasse* **.hinweis** *und dem* **<p>** *Tag eine blaue Schriftart zugewiesen.*

```
.tipp, .hinweis, p {color: blue;}
.wichtig {background: #CCC;}
.fett {font-weight: bolder;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<h1 class="tipp">Denkspiele</h1>
<h2 class="hinweis">Online oder Offline</h2>
<p>Es gibt so viele Möglichkeiten</p>
```

Ein HTML Element kann mehrere Klassen haben. Hier im Beispiel hat das **<h1>** *Element eine Klassenzugehörigkeit zu* **.wichtig** *und zu* **.fett.**

```
<h1 id="haupt" class="wichtig fett">Dokumente</h1>
```

*Um alle Elemente anzusprechen, kann man den **Universalselektor** verwenden: Der Universalselektor wird mit einem Stern eingeleitet:*
z. B.: `* {color: green; font-family: monospace;}`

7.1 Hintergrund

HCL071

Wir können schon die Hintergrundfarbe eines Elements bestimmen. Jetzt wollen wir Bilder als Hintergründe verwenden. Dafür gibt es die **`background`** *Eigenschaft.*

Hintergrundgrafik einbinden

CSS

`background-image: url("[pfad]");`

`[pfad]` Es wird der Pfad (absolut oder relativ) zum Bild angegeben. Bei einer externen .css Datei ist der Speicherort der .css Datei ausschlaggebend.

```
body {background-image: url("../pix/hintergrund.jpg");}
```

Hintergrundbild skalieren (Größe verändern)

CSS

`background-size: [value];`

`[value]` Längenangaben wie z. B. px, % sind möglich. Dabei wird zuerst die Breite und dann die Höhe definiert.
`background-size: 200px 400px;`
Wird nur ein Wert angegeben, dann wird die Höhe unter Beibehaltung des Seitenverhältnis skaliert.

Folgende Schlüsselwörter sind möglich:

`auto`	Originalgröße des Bildes, keine Skalierung
`contain`	Beibehaltung des Seitenverhältnis, die größere Seite wird angepasst.
`cover`	Beibehaltung des Seitenverhältnis, die kleinere Seite wird angepasst. Damit wird der Anzeigebereich vollständig gefüllt.

```
 #haupt {background-size: cover;}
.bilder {background-size: 300px;}
```

Hintergrund wiederholen

CSS

`background-repeat: [value];`

`[value]` Wenn das Bild kleiner als das Element ist, kann man es wiederholen (kacheln). Dabei wird der gesamte zur Verfügung stehende Platz ausgefüllt. Folgende Werte sind möglich:

`repeat`	Wiederholung des Hintergrundbildes
`repeat-x`	Das Bild wird horizontal wiederholt.
`repeat-y`	Das Bild wird vertikal wiederholt
`no-repeat`	keine Wiederholung
`space`	Das Hintergrundbild wird wiederholt ohne dass ein Bild beschnitten wird.
`round`	Genauso wie **`space`**, hier wird aber das Bild skaliert.

```
#haupt {background-repeat: repeat-x;}
.bilder {background-repeat: space;}
```

7.2 Hintergrund

Position der Hintergrundgrafik

CSS

`background-position: [value-x] [value-y];`

`[value-x]` Bestimmt die Position entlang der x-Richtung (horizontal). Erlaubt sind negative und positive Längenmaße und die Werte: **`left center right`**.

`[value-y]` Bestimmt die Position entlang der y-Richtung (vertikal). Erlaubt sind negative und positive Längenmaße und die Werte: **`top center bottom`**.

Der Ausgangswert ist `0 0` – Das ist in der linken oberen Ecke des Elements.

```
#haupt {background-position: 50px 100px;}
```

Mit **`background-origin`** *wird ein Ausgangspunkt festgelegt.*
z. B: **`background-origin: border-box;`** *richtet den Hintergrund auf die Außenkante des Rahmens.*

Hintergrundbild befestigen

CSS

`background-attachment: [value];`

Damit definiert man das Verhalten, wenn ein Element oder sein Inhalt bewegt wird. Folgende Werte **`[value]`** stehen zur Verfügung:

`fixed`	**Bewegung des Elements:**	ein anderer Bereich des Hintergrund wird sichtbar.
	Bewegung des Inhalts:	der Hintergrund bleibt unverändert.
`scroll`	**Bewegung des Elements:**	Hintergrund bleibt unverändert und scrollt mit dem Element mit.
	Bewegung des Inhalts:	der Hintergrund bleibt unverändert.
`local`	**Bewegung des Elements:**	Hintergrund bleibt unverändert.
	Bewegung des Inhalts:	der Hintergrund bewegt sich mit.

```
body {background-attachment: fixed;}
```

Hintergrundeigenschaften für bestimmte Bereiche

CSS

`background-clip: [value];`

Die Hintergrundeigenschaften (z. B. Bild oder Farbe) werden hier für einen bestimmten Bereich festgelegt. Dafür gibt es folgende Werte **`[value]:`**

`content-box`	Hintergrund wird nur im Inhalt angezeigt.
`padding-box`	Hintergrund im Inhalt und im Innenabstand
`border-box`	Hintergrund im Inhalt, Innenabstand und im Rahmen.

Der Ausgangswert ist `0 0` – Das ist in der linken oberen Ecke des Elements.

```
#haupt {background-clip: content-box;}
```

7.3 Farbverlauf

Um einen Farbverlauf im Hintergrund eines Elements zu erstellen, kann man eine Graphik bzw. Bilddatei erstellen oder man nutzt CSS mit der Funktion **`linear-gradient()`**. *Diese Funktion lässt sich z. B. mit der CSS Eigenschaft* **`background-image`** *darstellen.*

CSS

```
background-image:
linear-gradient([to value], [farbeVon], [farbeBis]);
```

`[to value]` Gibt die Richtung an. Dafür kann man ein Winkelmaß (z. B. **`75deg`**) angeben oder in Verbindung mit dem Schlüsselwort **`to`** die Werte **`top, right, bottom, left`**. Der Standartwert (ohne Angabe) ist **`to bottom = 180deg = 200grad`**.

`[farbeVon]` Der Color-Start-Wert kann eine beliebige Farbangabe sein. Es ist auch der Wert **`transparent`** möglich. Zusätzlich kann noch ein Längenmaß mitgegeben werden.

`[farbeBis]` Für den Color-End-Wert gelten die gleichen Bedingungen wie für den Color-Start.

Die Wertangaben werden durch Beistriche getrennt.

```
#dahinter {background-image:
           linear-gradient(to top right, red, transparent);}
```

Natürlich kann man mehrere Farben zuweisen.
Im Beispiel: 75° rot bis orange bis gelb bis #FFC bis Weiß.

```
#hinten {background-image:
         linear-gradient(75deg, red, orange, yellow, #FFC, white);
         width:21cm; height:2cm; margin:auto;  border: 4px groove;}
```

```
<div id="hinten"> </div>
```

Mit CSS eine Österreich-Flagge erstellen.
Dafür benötigen wir Längenangaben.

```
#austria {background-image:
     linear-gradient(red 100px, white 100px, white 200px, red 100px);
     width:400px; height:300px; margin:auto; border: 4px groove;}
.schrift {margin-top:125px; font-size:50px; text-align:center;}
```

```
<div id="austria"><p class="schrift">Österreich</p></div>
```

Übung A: Carnary Wharf

- ❑ Öffne die HTML Datei **`carnary_wharf.html`** (in dieser HTML Seite sind zwei divs. Gekennzeichnet als **`ERSTES DIV`** und **`ZWEITES DIV`**).
- ❑ Im Seitenhintergrund soll das Bild **london.png** eingebettet werden. Das Hintergrundbild soll am unteren Seitenrand fixiert und auf die Seitenbreite skaliert dargestellt werden.
- ❑ Ändere die Farbe des Seitenhintergrund in ein angenehmes Blau.
- ❑ **`ERSTES DIV`**: Suche im Internet nach einem Bild der Englischen Fahne (Union Jack) und binde es als Hintergrundgraphik im ersten Div ein.
- ❑ **`ZWEITES DIV`**: Definiere einen Farbverlauf, von Oben nach Unten und von Gelb nach Transparent. Entferne den Rahmen um das zweite Div.
- ❑ **Überprüfe ob in den Meta-Tags des Dokuments dein Name steht!**

Übung B: Rainbow Warrior

- ❑ Definiere einen linearen Farbverlauf (in Regenbogenfarben) für den Schriftzug ***RAINBOW WARRIOR***.
- ❑ Du brauchst vielleicht noch folgende CSS Eigenschaften:
  `background-clip` **und** `text-fill-color;`
- ❑ **Recherchiere selbstständig nach den zusätzlichen CSS Eigenschaften und löse diese Aufgabe nur mit HTML und CSS!**

RAINBOW WARRIOR

8.1 Positionierung

HCL081
Beispiel_Pos.html

*HTML Elemente müssen nicht immer nur in Laufrichtung des Textes positioniert werden. Man kann jedes Element (**div**, Bilder usw) an jeder Stelle im Browserfenster (viewport) anzeigen. Dafür gibt es die CSS-Eigenschaften **top**, **bottom**, **left** und **right**. Diese können durch numerische Angaben (z. B. **top: 30px;**) bestimmt werden. Zuvor muss man dem Element eine Positionierungsart zuweisen.*

CSS

position: [value];

Für **[value]** sind folgende Werte möglich:

static	Ist der Defaultwert. Das Element bleibt im Textfluss. **top**, **bottom**, **left** und **right** werden ignoriert.
fixed	Richtet das Element am Viewport aus und fixiert es dort. Das Element bleibt beim Scrollen an seiner Position. (z. B. eine fixe Kopf- oder Fußzeile).
relative	Richtet das Element von seiner eigenen Position im Textfluss aus. Es hinterlässt eine Lücke.
absolute	Orientiert sich bei **top**, **bottom**, **left** oder **right** am Elternelement. Jedoch nur, wenn dieses Elternelement selbst mit **position**: positioniert wurde. Sonst bezieht es sich am **<html>** Element. Es behält beim Scrollen also seine Position.
sticky	Behaltet seine Position im Textfluss, bis das obere oder untere Seitenende erreicht wurde. Dort bleibt es dann kleben!

Hier ein Beispiel für eine fixe Fußzeile:

```
#fusszeile {position: fixed;
            bottom: 0px; right: 0px;
            font-size: 1.4em; text-align: right;
            width: 100%; background: #F0F0F0;
            padding: 10px;}
```

```
<div id="fusszeile">techcom GmbH - &copy; 2023</div>
```

***Beispiel:** Eine Beschriftung über einem Bild!*

```
.bilddiv {position: relative;}
   .bild {height: 300px; z-index: 1;}
.beschriftung {position: absolute;
               bottom: 50px; left: 10px;
               padding: 10px; background: white; font-size: 1.5em;
               z-index: 2;}
```

```
<div class="bilddiv">
   <img src="zug.jpg" class="bild" alt="Ein Zug">
   <div class="beschriftung">Rail Jet</div>
</div>
```

*Wenn Elemente sich überlappen, dann kann man mit **z-index** die Reihenfolge bestimmen, ähnlich wie die Ebenen im Photoshop oder Gimp.*

8.2 display Eigenschaft

*Mit der **display**-Eigenschaft bestimmt man die Art eines Elements näher.*

CSS

display: [value];

Für **[value]** sind folgende Werte möglich:

inline	Inline Boxen verlaufen in einer Zeile entlang der Schreibrichtung. Die Breite und Höhe wird allein durch den Inhalt bestimmt. **width** und **height** sind wirkungslos.
block	Block-Boxen haben die gleiche Breite wie das Elternelement. Die Höhe wird durch den Inhalt beeinflusst. Ein Beispiel für eine Block-Box ist der **<p>** Tag.
inline-block	Die Breite ist so schmal als möglich. Man kann die Breite mit **width** festlegen. Die Höhe ist vom Inhalt abhängig, kann aber mit **height** festgelegt werden.
flex	für flexible Layouts ohne fixe Größen (Flexible Box Layout Module)
grid	für flexible Layouts ohne fixe Größen. Im Gegensatz zu **flex**, werden hier komplexe Raster erzeugt. (Grid Layout)
list-item	behandelt ein Element wie einen ** ** Tag.
table	Um ein Element als eine Tabelle darzustellen. In Folge kann ein Block auch als Tabellenzelle mit **table-cell** dargestellt werden.
none	Das Element wird nicht erzeugt. Es ist unsichtbar und hat keinen Einfluss auf den Elementfluss. (es hinterlässt keine Lücke im Text). **display: none;** wird oft verwendet wenn man das Medium (z. B. Print) bzw. den Viewport (z. B. für Smartphones) wechselt.

Das Beispiel zeigt, wie man mehrere ** ** Elemente nebeneinander darstellen kann.

```
.nebenan {display: inline-block;
          width: 200px; height: 200px;
          margin: 10px; border: 3px groove blue;
          text-align: center;}
```

```
<div class="nebenan">EINS</div>
<div class="nebenan">ZWEI</div>
<div class="nebenan">DREI</div>
```

*Der Unterschied zwischen **visiblitiy: hidden;** und **display:none;** besteht darin, dass **visibility** das Element ausblendet, während **display: none;** es nicht erzeugt. Der Unterschied liegt dann im Element- bzw. Textfluss.*

Zwei Beispiele wie man einen Text horizontal und vertikal über den gesamten Viewport zentriert: `<div id="haupt">Eilmeldung</div>`

```
#haupt {display: table-cell;
        background: #CCC;
        height: 100vh;
        width: 100vw;
        vertical-align: middle;
        text-align: center;}
```

```
#haupt {display: flex;
        background: #CCC;
        height: 100vh;
        align-items: center;
        justify-content: center;}
```

Übung A: Textgestaltung

- ❑ Öffne die Lösung von **6.2 C: Text mit Spalten** bzw. die Datei **Wordelemente.html**
- ❑ Lösche die Überschrift

```
<h1>Microsoft Word: Einfügen von Elementen</h1>
```

1 ❑ Links oben soll ein ** Textverarbeitung ** hinzugefügt werden. Gestalte es ansprechend. Das ** ** soll nicht über gesamte Bildschirmbreite erscheinen und links über das schwarze ** ** hinausragen.

2 Beim scrollen soll das ** ** mitwandern!

❑ Füge eine weitere Überschrift-Ebene mit dem Text "**Einfügen von Elementen**" hinzu. Die Überschrift soll beim Scrollen am oberen Browserrand 'kleben' bleiben. Gestalte die Überschrift ansprechend!

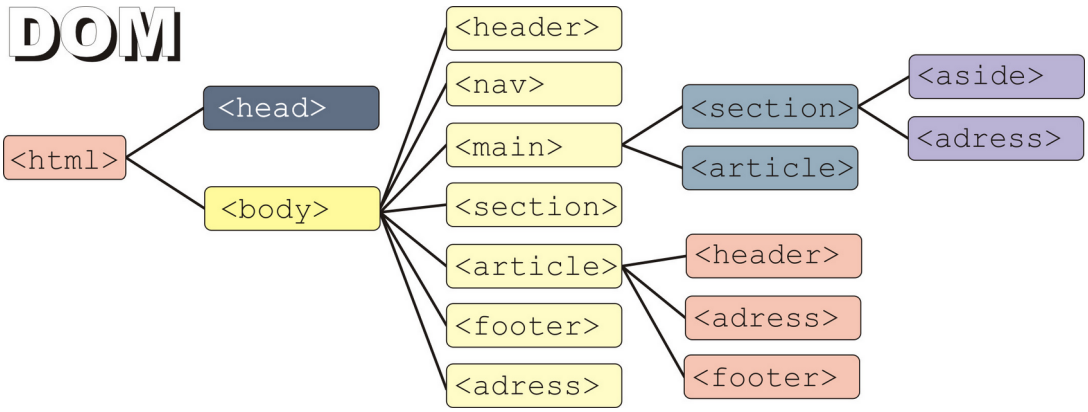
3 ❑ **Erstelle eine Fußzeile für das HTML-Dokument, die am unteren Browserrand fixiert werden und immer im Vordergrund sein soll. Der Text für die Fußzeile lautet: © by Vorname Nachname, Datum**

Übung B: Textgestaltung mit Bild

- ❑ Öffne deine Lösung von **8.2 A: Textgestaltung**.
- ❑ Besorge dir einen Screenshot von LibreOffice Writer und füge ihn als Bilddatei in den HTML Code ein.
- ❑ Beschrifte das Bild mit "Screenshot LibreOffice" Die Beschriftung soll auf das Bild gelegt werden.

8.3 Strukturierung

Für die Seitenstruktur haben wir bisher **<body>, , <p>** *usw. verwendet. HTML5 bietet aber noch weitere Tags zur Strukturierung. Alle Tags müssen noch mit CSS formatiert werden. Die Strukturierung ermöglicht auch ein barrierefreies Web, weil Screenreader* **<header>, <nav>** *oder* **<footer>** *überspringen und nur die wichtigen Inhalte vorliest.*



HTML

<nav> … </nav>

Das **<nav>** Element bezeichnet die Hauptnavigation und bildet einen eigenen Inhaltsbereich.

HTML

<main> … </main>

Das **<main>** Element enthält den Hauptinhalt der Webseite.

HTML

<header> … </header> und **<footer> … </footer>**

Das **<header>** Element wird für den sichtbaren Kopfbereich verwendet.
Das **<footer>** Element enthält Informationen zur Website und steht am Ende.
Beispiele für Infos im **<footer>** sind Hinweise zu Urheberrecht, Autor oder ein Link zum Impressum.

HTML

<section> … </section> und <article> … </article>

Das **<section>** Element umspannt eine thematische Gruppierung. Es dient zur semantischen Gliederung von Inhalten. **<article>** ist mit einem Zeitungsartikel vergleichbar und stellt einen sich geschlossenen Abschnitt dar.

Unter Semantik versteht man die Bedeutung bzw. den Inhalt eines Wortes, Satzes oder Textes.

HTML

<adress> … </adress> und <aside> … </aside>

Das **<adress>** Element enthält Kontaktinformationen (z. B. eMail-Adressen, Postadressen, Link zu Seiten mit Kontaktinformationen usw.).
<aside> dienst für Randbemerkungen. Die Darstellung muss über CSS festgelegt werden.

8.4 Float

HCL084
Beispiel_Pos.html

Alle HTML Elemente sind im Fluss. Um nun aber Bilder oder andere Elemente links oder rechts vom Fließtext darzustellen, gibt es die **`float`** *Eigenschaft. Dabei wird das Objekt links oder rechts an die Innenkante des Elternobjekts verschoben.*

CSS

`float: [Value];`

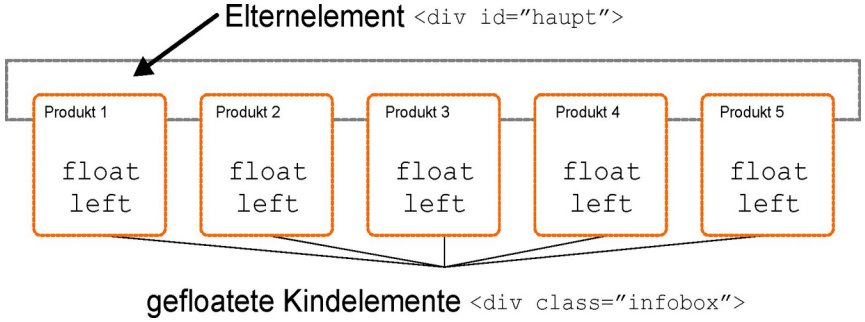
Für **`Value`** sind folgende Werte erlaubt:

`none`	Standardwert, das Element wird nicht verschoben
`left`	Das Element wird nach links verschoben.
`right`	Das Element wird nach rechts verschoben.
`inherit`	Erbt die **`Float`**-Eigenschaft des Elternelements.

```
<p><img src="../pix/bild1.jpg" style="float:right;"
        alt="Beispielbild">... weiterer Lauftext ...</p>
```

Wenn man mehrere Elemente mit einer **`float`** Eigenschaft einbindet, dann sind die Kindelemente manchmal größer als das Elternelement und ragen über dieses hinaus. Dieser Effekt wird auch **collapse** genannt. Der Grund: Die Kindelemente werden wegen der **`float`** Eigenschaft aus dem Fließtext herausgenommen obwohl sie den Text verdrängen!

```
  #haupt {border:2px dashed gray; min-height:50px;}
  .infobox {float:left; margin:10px; height:200px; width:200px;
            border:2px dotted orange;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   <div id="haupt">
      <div class="infobox">Produkt 1</div>
      <div class="infobox">Produkt 2</div>
      <div class="infobox">Produkt 3</div>
      <div class="infobox">Produkt 4</div>
      <div class="infobox">Produkt 5</div>
   </div>
```



Um dem entgegenzuwirken gibt es mehrere Möglichkeiten. Eine davon ist es, am Ende ein leeres HTML Element mit der CSS Eigenschaft **`clear:both;`** *hinzuzufügen.*

```
<div id="haupt">
     ...
  <div class="infobox">Produkt 5</div>
  <div style="clear:both;"> </div>
</div>
```

9.1 Pseudoklassen (Buttons)

HCL091

Um elegante Buttons mit HTML und CSS zu erstellen, kann man dynamische Pseudoklassen verwenden. Sie werden mit einem Doppelpunkt an die CSS Eigenschaft angeschlossen. z. B: **`a:link {…} .imlink:hover {…} #derlink:visited {…}`**
Hier in Verbindung mit dem **`a href`** *Element (Hyperlink):*

CSS

`a:link {…}`

Kennzeichnet nur unbesuchte Links. Die Verwendung ist in den **`a-, area-`** oder **`link-`** Elementen mit **`href`**-Attribut möglich.

```
a:link {text-decoration: none;}
```

CSS

`a:visited {…}`

Kennzeichnet besuchte Links (also der User hat darauf geklickt). Achtung: Aus Datenschutzgründen unterstützen moderne Browser nur eine eingeschränkte Möglichkeit von Eigenschaften für diese Pseudoklasse!

```
a:visited {text-decoration: line-through;}
```

CSS

`a:hover {…}`

Kommt zur Anwendung, wenn der Mauszeiger das Element berührt. Verlässt der Mauszeiger das Element, dann wird der vorherige Zustand wieder hergestellt. Die Pseudoklasse **`:hover`** lässt sich auf beinahe alle HTML Elemente anwenden.

```
a:hover {color:red;}
```

CSS

`a:focus {…}`

Sobald der Fokus (z. B: durch die Tabulatortaste) auf ein HTML Element mit einer **`:focus`** Pseudoklasse gesetzt wird.

```
a:focus {border: 2px solid green;}
```

Weitere Pseudoklassen von Interesse im Zusammenhang mit Hyperlinks sind: **`a:any-link`** *(Kennzeichnung von besuchten und unbesuchten Links) und* **`a:active`** *(sobald ein Hyperlink aktiviert wird).*

Hier ein Beispiel für einen HTML-Button mit CSS für die Navigation:

```
.schalter:link {text-decoration: none; font-size: x-large;
                padding: 10px; border: 2px groove gray;
                border-radius: 10px;
                background: LightGray; color: black;}

.schalter:focus, .schalter:hover, .schalter:active {background: blue;
                                                    color: white;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  <a href="kontakt.html" class="schalter">Kontakte</a>
  <a href="about.html" class="schalter">Über uns</a>
```

9.2 Pseudoklassen

Mit der Pseudoklasse **`:target`** *werden Eigenschaften eines gerade aktiven Verweis dargestellt. Den aktiven Verweis sieht man in der URL Leiste*

`http://html.css4.at#L91` *öffnet die Webseite und springt zum Element mit der* **`id="L91".`** *Man spricht hier auch von Anker oder Sprungmarken.*

Weitere Sprungmarken:

`href="#"`	*Springt zum Seitenanfang*
`href="#top"`	*Springt ebenfalls zum Seitenanfang*
`href=""`	*erzwingt ein Neuladen der Seite!*

CSS **`:target {…}`**

Zur Aktivierung ist die Vergabe einer ID notwendig.
Die Pseudoklasse eignet sich besonders gut in Verbindung mit einer Klasse.

Beispiel: Ein- und ausblenden eines Elements.
Zuerst werden alle Elemente der Klasse **`.details`** ausgeblendet. Wegen **`.details:target`** wird ein Element wieder dargestellt, sobald die ID in der aufgerufenen URL steht (öffnen der Seite mit Sprungmarke)!

```
.details {display: none; background:#F0F0F0; padding:2em;}
.details:target {display:inline-block;}
```

```
<h2 id="ueber">Nachrichten</h2>
<p>Die USA plant ein Freihandelsabkommen ...
   <a href="#d1">--- mehr lesen ---</a></p>

<div id="d1" class="details">
  <h3>Weltweiter freier Handel</h3>
  <p>Die USA und die EU haben sich geeinigt.
     Die Strafzölle werden wieder abgeschafft.</p>
  <a href="#ueber" title="schließen">schließen</a>
</div>
<p>Frieden in Korea
   <a href="#d2">--- mehr lesen ---</a></p>

<div id="d2" class="details">
  <h3>Koreas Atompolitik hat positive Folgen</h3>
  <p>Der Konflikt zwischen Nord- und Südkorea
     ist jetzt entgültig zu Ende.</p>
  <a href="#ueber" title="schließen">schließen</a>
</div>
```

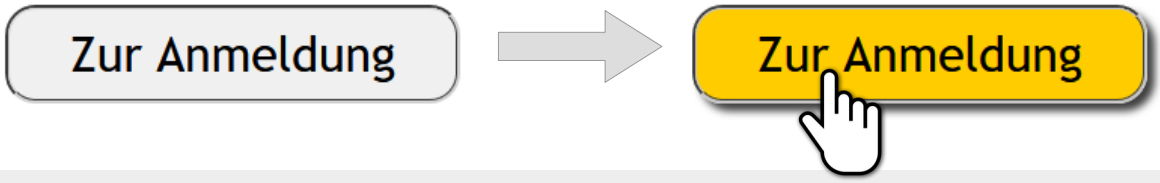
Speichere den Code als **`nachrichten.html`** *ab und öffne das Dokument direkt zum Ankerpunkt d1 mit:* **`nachrichten.html#d1`**

Es gibt noch eine Vielzahl von strukturellen und dynamischen Pseudoklassen wie:

`:first-child`	*Spricht das erste Kindelement an*
`:last-child`	*Spricht das letzte Kindelement an*
`:empty`	*Spricht leere Elemente an (z. B. ohne Text)*

Übung A: Buttons

- ❑ Gestalte eine Button-Klasse (wie unten dargestellt).
- ❑ Verwende die Pseudoklassen **`:link`**, **`:visited`**, **`:any-link`**, **`:hover`**, **`:focus`**, und **`:active`**.



Übung B: Links neu gestalten

- ❑ Öffne das HTML Dokument **`abfalltrennung.html`**
- ❑ Alle Links (a href) sollen folgendes Aussehen haben:
- ❑ Jeder Link: keine Unterstreichung, dunkelgraue Schriftfarbe, Fettschrift
- ❑ Besuchte Links: Dunkelgrüne Schriftfarbe
- ❑ Hover: gepunktete Unterstrichen, blaue Schriftfarbe
- ❑ Fokus: einfach Unterstrichen, rote Schriftfarbe
- ❑ Aktive Links: doppelt Unterstrichen, schwarze Schriftfarbe

Übung C: Navigationsleiste

- ❑ Öffne deine Lösung von **9.2 B: Links neu gestalten** (**`abfalltrennung.html`**).
- ❑ Erstelle eine horizontale Navigationsleiste.
- ❑ Die einzelnen Elemente sollen einen **`:hover`** und einen **`:focus`** Effekt haben.
- ❑ Folgende Navigationspunkte sind vorgesehen:
  Papier | Leichtverpackung | Metall | Bio | Restmüll
- ❑ Definiere für die Navigationspunkte Sprungmarken (bzw. Anker) zu den dazu passenden Müllsorten.
- ❑ **Freie kreative Übung für die Gestaltung und Aussehen!**

Übung D: Ein- und ausblenden

- ❑ Öffne deine Lösung von **9.2 C: Navigationsleiste** (**`abfalltrennung.html`**).
- ❑ Blende alle **`<div class="sorte">`** mit CSS aus.
- ❑ Per Klick auf einen Navigationspunkt soll das dazu passende `<div class="sorte">` wieder eingeblendet werden.

9.3 iFrames

HCL093

*Das **iFrame** Element wird zum Einbinden von HTML-Seiten oder anderen Inhalten (z. B. Bilder) verwendet. Dabei werden fremde Quellen eingebunden (von innerhalb oder außerhalb der Site) und auf der eigenen Webseite angezeigt.*

HTML

<iframe name="" src="" sandbox=""></iframe>

Das **<iframe>** Element hat folgende Attribute:

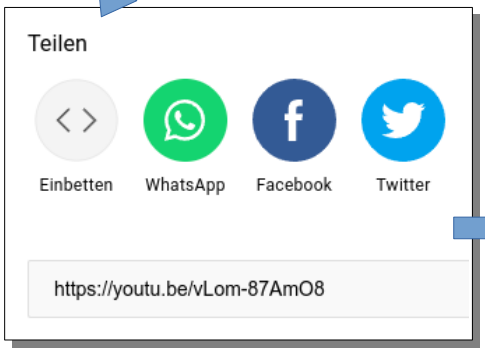
height	Höhe (CSS wird empfohlen)
width	Breite (CSS wird empfohlen)
name	Name des **iframes** (wird benötigt um später über einen Link **a href** mit dem Attribut **target** etwas einzubinden).
src	Quelle die beim Start gezeigt werden soll.
sandbox	Sicherheitseinstellungen:
	allow-forms Formularabsendungen sind erlaubt
	allow-popups erlaubt Popups
	allow-scripts erlaubt Scripting
	...

```
<h1>Österreichische online Tageszeitungen</h1>
<a href="https://www.kurier.at" target="meinIframe">Kurier</a>
<a href="https://www.krone.at" target="meinIframe">KronenZeitung</a>
<a href="https://www.oe24.at" target="meinIframe">oe24.at</a><br>

<iframe height="500" width="300" name="meinIframe"
        src="https://derstandard.at/" sandbox="allow-forms"></iframe>
```

Achtung: Manch fremde Websites verhindert die Darstellung in iFrames mit technischen Hilfsmitteln.

Einbetten eines YouTube Videos!

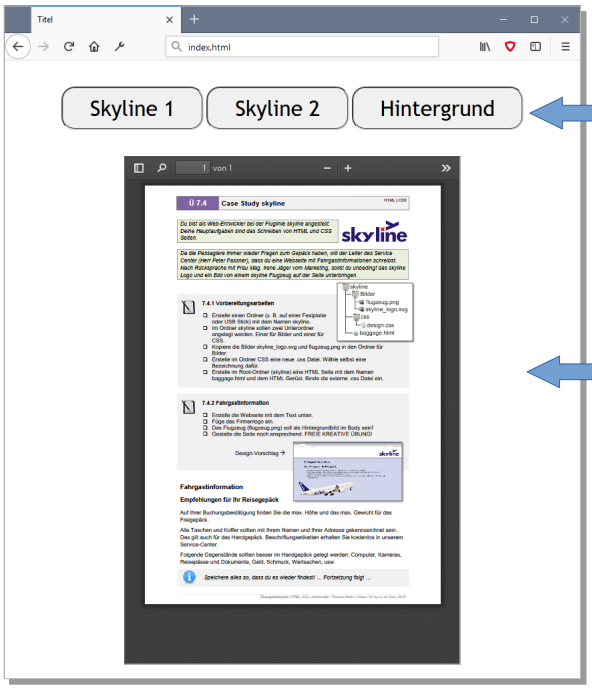
1. YouTube.com besuchen und ein Video auswählen
*2. Auf **TEILEN** klicken*
*3. Auf **EINBETTEN** klicken*
4. Den Quellcode mit dem <iframe> kopieren und in die eigene Webseite einfügen



Ü 9.3 Übungen iFrames

Übung A: PDF darstellen

- ❑ Erstelle ein neues HTML Dokument
- ❑ Füge ein **`iFrame`** mit den ähnlichen Seitenverhältnissen eines DIN-A4 Blattes hinzu.
  TIPP: A4 hat 210 mm Breite und 297 mm Höhe,
  (z. B.: Breite: 500 px und Höhe: 707 px)
- ❑ Erstelle drei Link-Buttons.
- ❑ Verlinke die Buttons mit beliebigen PDF Dokumenten (im Din A4 Format). Die Darstellung soll im `iframe` passieren.



Buttons (Target ist das iframe)

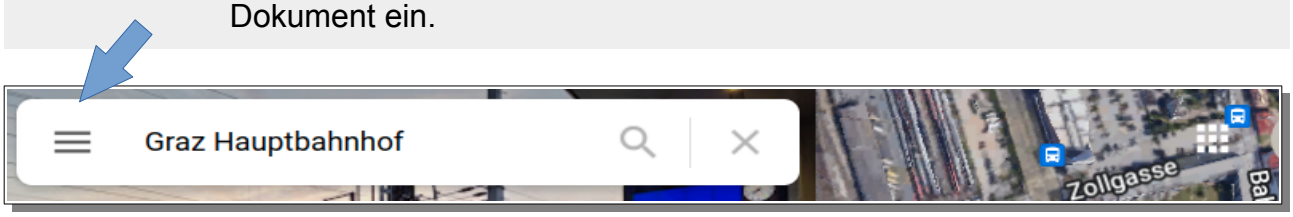
Die meisten modernen Browser können PDFs darstellen. Einige skalieren die Darstellung brauchbar, einige blenden zusätzlich noch eine Navigation mit ein. Es ist also ratsam die PDF Darstellung in einem iframe unbedingt auf verschiedenen Browsern zu testen.

Übung B: YouTube

- ❑ Erstelle ein neues HTML Dokument
- ❑ Suche auf YouTube.com ein cooles Video und bette es in das HTML Dokument ein.

Übung C: Google Maps

- ❑ Erstelle ein neues HTML Dokument
- ❑ Suche auf Google Maps eine Adresse und bette die Karte in dein HTML Dokument ein.



10.2 Pseudoelemente

Pseudoelement können an einfache Selektoren (ID, Class usw.) angehängt werden um damit ein Element zu erweitern. Um Pseudoelemente von Pseudoklassen zu unterscheiden, werden sie mit zwei Doppelpunkten (::) geschrieben.

CSS

`::first-line {…}`

Spricht die erste Text-Zeile eines Elementes an.

```
p::first-line {font-weight:bold;}
```

CSS

`::first-letter {…}`

Spricht das erste Zeichen in einem Element an.

```
p::first-letter {font-size:4em; float:left; padding:10px;}
```

CSS

`::before {content: …}`

Fügt für ein Element zusätzliche Inhalte hinzu.
`span::before {content: ' OK ';}`
Vor jedem **<span>** wird ein OK hinzugefügt.
`p::before {content: url("icon.png");}`
Vor jedem **<p>** wird ein PNG-Bild hinzugefügt.

CSS

`::after {content: …}`

Analog zu **`::before`**. Der Inhalt wird hier nach dem Element angezeigt.

```
a::before {content: "Link: ";}
a::after {content: " OK "}
p::after {content: ""; display: inline-block;
          background: url("icon.png") no-repeat;
          width: 50px; height: 50px;}
```

CSS

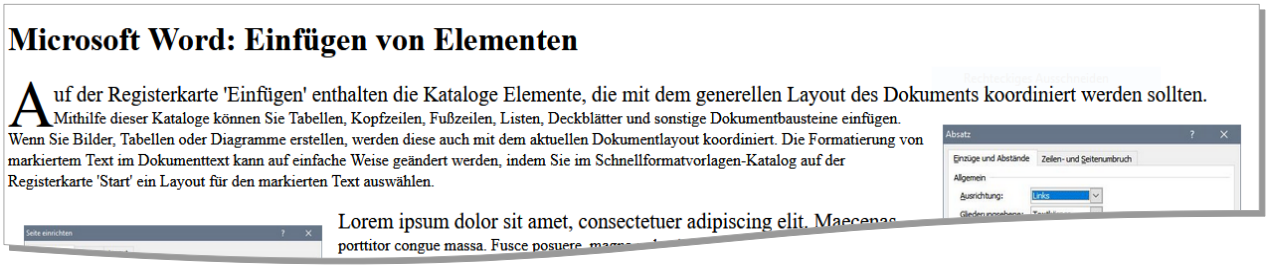
`::selection {…}`

Definiert die Hintergrund- und Schriftfarbe eines markierten Textes.
Das Beispiel zeigt eine Möglichkeit, um das Markieren von Text optisch zu unterbinden:

```
div {color:black; background:gray;}
p::selection {color:inherit; background:inherit;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<div><p>Sehr viel Lauftext</p> ist nicht immer gut!</div>
```

Übung A: Wordelemente

- Öffne die Webseite **Wordelemente.html**
- Erstelle drei beliebige Screenshots von Microsoft Word, LibreOffice oder einem anderen Textverarbeitungsprogramm.
- Binde die drei Bilder an beliebiger Stelle ins HTML Dokument ein.
- Floate zwei Bilder rechts und eines links.
- Der erste Buchstabe jedes Absatzes soll ein Initial sein. (Hervorgehobener Buchstabe).
- **Die erste Zeile jedes Absatzes soll etwas hervorgehoben werden!**



Übung B: Datei Explorer

- Suche im Internet nach Programmsymbolen für
  - Word Dateien (.docx)
  - Excel Dateien (.xlsx)
  - PDF Files (.pdf)
- Erstelle ein HTML Dokument mit einer Tabelle. Die Tabelle soll die Spalten Dateiname, Datum, Typ und Größe haben!
- Schreibe für jeden Dateityp (Word, Excel, PDF) eine eignen CSS-Klasse.
- Vor jeder Klasse soll das dazu passende Icon (Programmsymbol) automatisch hinzugefügt werden. Verwende dafür: **`::before`**
- **Nach jeder Klasse soll die dazu passende Dateierweiterung automatisch hinzugefügt werden. Verwende dafür `::after`**

Dateiname	Datum	Typ	Größe
Planung.docx	2018-06-30	Word-Dokument	30 KiB
Kostenrechnung.xlsx	2018-06-29	Excel-Dokument	78 KiB
Endbericht.pdf	2018-06-28	PDF-Dokument	308 KiB

10.3 Audio

HCL103
Europahymne.mp3

Mit HTML 5 kann man auch Audio- und Videodateien einbinden. Ratsam ist es, mit Multimediaelementen gut überlegt umzugehen. Automatisch abgespielte Soundfiles können den Benutzer entnerven – große Dateien brauchen selbstverständlich mehr Ladezeit.

Browsersupport					
<audio>	4.0	9.0	3.5	4.0	10.5

HTML

<audio> </audio>

Es können Musik und Soundfiles in den Formaten .mp3, .wav und .ogg abgespielt werden. Wobei .mp3 den Vorzug haben sollte, weil dieses Format von den meisten Browsern unterstützt wird.

Folgende Attribute sind für das **<audio>** Element verfügbar:

autoplay	beginnt sofort mit dem Abspielen ohne Wert
controls	zeigt die Steuerungsfunktion an (Lautstärke usw.) ohne Wert
loop	das Abspielen wird endlos wiederholt ohne Wert
muted	zu Beginn stumm schalten ohne Wert
preload	definiert das Ladeverhalten: **auto** ← die gesamte Datei wird geladen **none** ← die Datei wird nicht vorgeladen **metadata** ← es werden nur die Metadaten geladen
src	URL zur Audiodatei

Zusätzlich gibt es noch TimeRange Attribute für JavaScript:

buffered	liest aus, wie viel zwischengespeichert wurde
played	liest aus, wie viel schon gespielt wurde.

Der Text zwischen dem **<audio>** Tag wird angezeigt, wenn das Abspielen nicht möglich ist. z. B. weil es in den Browsereinstellungen deaktiviert wurde.

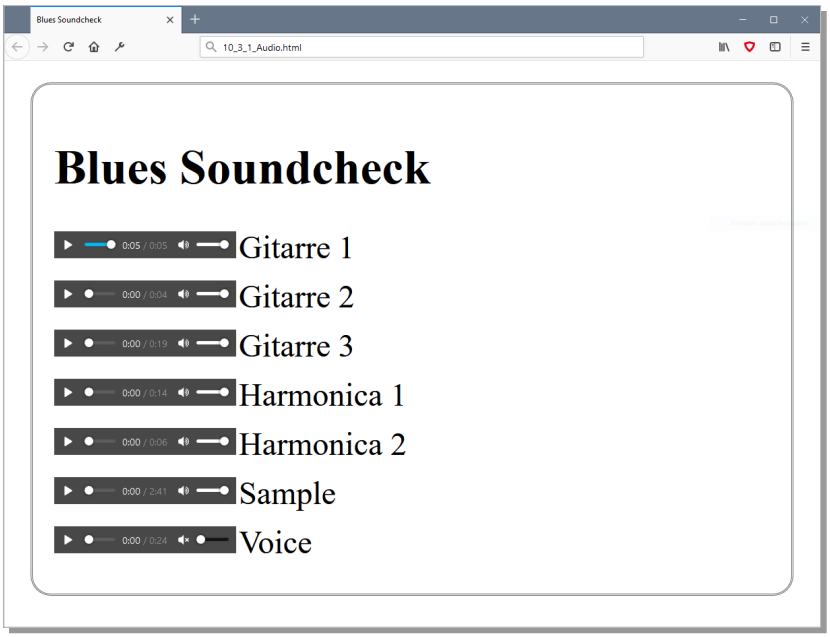
```
<audio controls loop src="Europahymne.mp3">
       Keine Audiowiedergabe möglich!</audio>
```

Ein Fallback ist eine Alternative die abgespielt werden soll, wenn das primäre Sound-Format nicht abgespielt werden kann.
Dafür bekommt das **<audio>** *Element weitere* **<source>** *Tags:*

```
<audio controls autoplay >
   <source src="Europahymne.mp3" type="audio/mp3">
   <source src="Europahymne.ogg" type="audio/ogg">
    Keine Audiowiedergabe möglich!
</audio>
```

Übung A: Blues Soundcheck

- Entzippe die Datei ***Bluesfiles.zip***
- Erstelle ein neues HTML Dokument mit allen Vorarbeiten (Title, Datum usw.)
- Füge die Überschrift "Blues Soundcheck" hinzu.
- Füge alle Soundfiles aus der Datei ***Bluesfiles.zip*** mit dem `<audio>` Tag hinzu.
- Beschrifte die `<audio>` Tags mit einem Hinweistext falls die Darstellung nicht möglich ist und einer lesbaren Bezeichnung des Audioelements.
- Alle `<audio>` Elemente haben Bedienelemente (Play, Lautstärke usw.)
- ***guitar.wav*** soll beim Öffnen der Webseite sofort abgespielt werden.
- Erstelle für ***guitar3.ogg*** ein Fallback auf ***guitar3.wav***.
  Die Idee: ***guitar3.ogg*** hat 311 KiB und soll deshalb auch bevorzugt abgespielt werden. Da aber .ogg Files nicht von allen Browsern abgespielt werden, soll alternativ die ***guitar3.wav*** mit 3369 KiB abgespielt werden.
- ***harmonica.mp3*** soll endlos wiederholt werden.
- ***sample.mp3*** hat eine Größe von 2.5 MiB und soll deshalb automatisch geladen, aber noch nicht automatisch abgespielt werden!
- ***voice.mp3*** soll zum Start stumm geschaltet sein.
- **Teste dein Webdokument auf verschiedenen Browsern!**



Übung B: Blues Designverständnis

- Öffne deine Lösung von **10.3 A: Blues Soundcheck** und verpasse dem Dokument ein cooles Blues Design.
- **Freie kreative Übung!**

10.4 Video

HCL104
paint.mp4

Mit HTML 5 kann man Videodateien einbinden ohne das Plugins oder Add-Ons notwendig sind. Die Nutzung des Video-Elements ist ähnlich wie die, des Audio-Elements. Ein Text innerhalb des <video> Tags wird angezeigt, wenn das Video nicht abspielbar ist.

Browsersupport					
<video>	4.0	9.0	3.5	4.0	10.5

TML

<video> </video>

Es können Videos in den Formaten .mp4, .ogg und .WebM abgespielt werden. Wobei man .mp4 bevorzugen sollte, weil dieses Format von den meisten Browsern unterstützt wird.

Folgende Attribute sind für das **<video>** Element verfügbar:

autoplay	beginnt sofort mit dem Abspielen ohne Wert
controls	zeigt die Steuerungsfunktion an (Play usw.) ohne Wert
loop	das Abspielen wird endlos wiederholt ohne Wert
muted	den Ton auf stumm schalten ohne Wert
preload	definiert das Ladeverhalten: **auto** ← die gesamte Datei wird geladen **none** ← die Datei wird nicht vorgeladen **metadata** ← es werden nur die Metadaten geladen
src	URL zur Videodatei
height, width	Größenangaben (ohne Angaben wird die Größe automatisch erkannt)

```
#myvideo {position:fixed; top:0px; left:0px;
          min-height:100%; min-width:100%;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<video id="myvideo" src="paint.mp4" autoplay controls >
   Das Video kann nicht dargestellt werden!</video>
```

MP4 = MPEG4-Dateien mit H264 video codec + AAC audio codec
WebM = WebM-Dateien mit VP8 oder VP9 video codec + Vorbis audio codec
Ogg = Ogg-Dateien Theora video codec + Vorbis audio codec

Ein Fallback ist eine Alternative die abgespielt werden soll, wenn das primäre Video-Format nicht angezeigt werden kann. Dafür werden im **<video>** *Element weitere* **<source>** *Tags gesetzt. Zusätzlich kann mit dem* **poster** *Attribut ein Vorschaubild definiert werden.*

```
<video controls autoplay poster="vorschaubild.png" >
   <source src="paint.mp4" type="video/mp4">
   <source src="paint.ogg" type="video/ogg">
   Keine Videoanzeige möglich!
</video>
```

Ü 10.4 Übungen Video

HCU104
paint.mp4

Übung A: Video mit Beschriftung

- Erstelle eine neue HTML Seite.
- Binde die Videodatei paint.mp4 in das HTML Dokument ein.
- Das Video soll automatisch starten.
- Das Video soll sich ständig wiederholen.
- Definiere **über dem Video** ein Beschriftungs-Div mit dem Text:
  Actionpainting
  Auf pixabay.com von Piro4D, Auflösung 640 x 360 Pixel
- **Das Beschriftungs-Div soll unsichtbar sein und erst wenn man mit dem Mauszeiger darüberfährt (Hover-Effekt) erscheinen!**

Es gibt mehrere Möglichkeiten, diese Aufgabe zu lösen. Am elegantesten ist wohl die CSS Eigenschaft `opacity`*. Diese bestimmt die Transparenz (Durchsichtigkeit) eines HTML Elements.*

CSS

`opacity: Value;`

Für den `Value` ist ein Zahlenwert zwischen 0 und 1 möglich. Wobei 0 keine Deckkraft und 1 eine 100%ige Deckkraft bedeutet. 0.5 ist halbdurchlässig.

```
#meinDiv {opacity: 0.4;}
```

Browsersupport					
`opacity`	4.0	9.0	3.5	4.0	10.5

11.1 Medienabfrage

Medienabfragen bestimmen die CSS Darstellung für ausgewählte Medientypen (z. B. Bildschirm, Druck) und/oder Medienmerkmalen (z. B. Höhe und Breite des Anzeigebereichs). Die **@media** *Abfrage kann überall im CSS Bereich (z. B. im Head oder in einer externen CSS) eingebunden werden.*

CSS

@media *Medientyp* and (*Medienfeature*) {…};

Folgende ***Medientypen*** sind möglich:

all	alle Ausgabemedien (Standardwert)
print	nur für den Druck
screen	nur für Bildschirme
speech	für die Sprachausgabe

Es gibt eine Vielzahl von Medienmerkmalen. Diese ***Medienfeatures*** gehören zu den wichtigsten:

width	Breite eines Anzeigebereichs, **min**- und **max**- sind erlaubt.
height	Höhe eines Anzeigebereichs, **min**- und **max**- sind erlaubt.
orientation	Seitenformat eines Mediums. Möglich sind **landscape** (Querformat) und **portrait** (Hochformat)
color	Anzahl der Farbbits die ein Ausgabegerät anzeigen kann. Der Wert 0 entspricht einer Anzeige ohne Farben.
resolution	Die Auflösung (Dichte der Bildpunkte) auf dem Ausgabemedium.
scripting	Wenn Scriptsprachen (z. B. JavaScript) verfügbar sind.

Im Beispiel wird die Schriftart der Klasse **.schalter** größer, wenn die Anzeigebreite größer als 1200 Pixel ist.

```
@media screen and (min-width: 1200px) {
        .schalter {font-size:larger;}
   }
```

Wenn man die Webseite für den Ausdruck vorbereiten möchte:

```
@media print {
       #inhalt {font-size: 12pt; width: 18cm; margin:auto;
                background-color: white;}
  }
```

Wenn eine Anzeigefläche (z. B. Browserbreite) kleiner-gleich 600px ist:

```
@media screen and (max-width: 600px) {
        .schalter {display:none;}
   }
```

Ü 11.1 Übungen Medienabfragen

Übung A: Media Queries

- Erstelle ein neues HTML Dokument.
- Füge ein ** ** Element mit folgenden CSS Eigenschaften ein:
  Breite: 80%, Höhe: 300 Pixel
  Zentriert über die Bildschirmbreite
  Rahmen: 10 Pixel Groove in der Farbe Blau
  Hintergrundfarbe: Grün
- Wenn die Browserbreite größer als 1020 Pixel ist, soll das ** ** einen schwarzen Hintergrund haben.
- Wenn die Browserbreite kleiner als 800 Pixel ist, soll das ** ** einen blauen Hintergrund haben mit einem orangen Rahmen.
- Wenn die Browserbreite kleiner als 600 Pixel ist, soll das ** ** keinen Rahmen mehr haben und eine Breite von 100% einnehmen.
  Hintergrundfarbe: Grün.
- **Teste deine Webseite**
  (z. B. mit Firefox → Web-Entwickler-Tools → Bildschirmgrößen testen).

Übung B: Media Queries mit Text

- Öffne deine Lösung von oben (11.1 B: Media Queries).
- Bei einer Browserbreite von mehr als 1020 Pixel soll der Text: "**Größer als 1020 Pixel**" erscheinen.
- **Bei einer Browserbreite von weniger als 800 Pixel soll der Text: "Kleiner als 800 Pixel" erscheinen.**

Übung C: Lebenslauf

- Erstelle einen Lebenslauf (von dir oder einer anderen prominenten Persönlichkeit) mit folgenden Inhalten:
  - Persönlichen Daten (Name, Geburtsdatum, usw.)
  - Berufspraxis (Ferialpraktika udgl.) und Ausbildung
  - Besondere Fähigkeiten (HTML, CSS, Buchhaltung usw.)
  - Hobbies (Webseiten schreiben usw.)
- Optimiere den Lebenslauf für Bildschirmdarstellungen und für den Druck!
- **Freie kreative Übung!**

11.2 Objekte

Bisher haben wir Bilder mit dem **img** *Element eingefügt. Mit dem* **picture** *Element in Verbindung mit* **source** *können wir verschiedene Bildversionen für verschiedene Pixeldichten, Auflösungen und Bildschirmformate einbinden. Im* **picture** *Element werden die Bilder mit dem* **source** *Tag und dem* **img** *Element als Fallback angegeben.*

HTML

<picture> ... </picture>

Im Beispiel wird das Bild **Zug_800.jpg** angezeigt, wenn der Anzeigebereich größer als 800 Pixel ist. Bei einem Viewport zwischen 800 und 600 Pixel wird das Bild **Zug_600.jpg** verwendet – beide wurden mit dem **source** Tag eingebunden. Als Fallback dient das **img** Element mit dem Bild **Zug_400.jpg**. Es wird angezeigt, wenn keines der anderen Bilder angezeigt werden kann, bzw. wenn das **picture** Element vom Browser nicht unterstützt wird.

```
<picture>
   <source media="(min-width: 800px)" srcset="Zug_800.jpg">
   <source media="(min-width: 600px)" srcset="Zug_600.jpg">
   <!---Fallback--->
   <img src="Zug_400.jpg" alt="Ein Railjet der OeBB">
</picture>
```

Das **objekt** *Element definiert eingebunden Objekte innerhalb des HTML Dokuments.* **objekt** *erlaubt das Einbinden von Audio, Video, Java Applets, ActiveX, PDF und Flash Files. Darüber hinaus, kann man auch ein anderes HTML Dokument oder Bilder anzeigen lassen. Mit dem* **param** *Tag übergibt man Parameter an das Plug-In.*

HTML

<object> ... </object>

Folgende Attribute für **object** sind verfügbar:

data	bestimmt den Ort der Quelle.
form	Indentifizierungsname für form Elemente
height	Höhe in Pixel oder %
width	Breite in Pixel oder %
name	Identifizierungsname des Elements
type	der Typ der Quelle

```
<object data="data/test.htm" type="text/html" width="300"
  height="200"> Alternativ: <a href="data/test.htm">test.htm</a>
</object>
```

HTML

<embed> ... </embed>

Zum Einbinden von Anwendungen oder interaktiven Inhalten.

```
<embed src="helloworld.swf" type="application/x-shockwave-flash">
```

Übung A: Smartphone Shop I

- Entzippe die Datei **`Smartphone_Shop.zip`**
- Erstelle ein neues HTML Dokument.
  Der **`<body>`** hat eine schwarze Hintergrundfarbe.
- Erstelle einen Banner mit der Bilddatei **`background.jpg`** als Hintergrund.
- Im Zentrum des Banners steht: "Smartphone Shop Graz"
- Unter dem Banner soll eine horizontale Navigation sein.
- In der Navigation sind folgende Buttons:
  Home | Produkte | Angebote | Webshop | Kontakt
- Gestalte die Webseite anspruchsvoll.
- Speichere die Webseite als `index.html` ab.

Übung B: Smartphone Shop II

- Öffne dein HTML Dokument **`index.html`** aus der
  Übung **11.2 A: Smartphone** Shop.
- Bette die Flashdatei **`flashfile.swf`** ins Dokument ein.
- Setzte eine Sprungmarke/Anker vom Home Button auf die Flashdatei.

Übung C: Smartphone Shop III

- Öffne dein HTML Dokument **`index.html`** aus der
  Übung **11.2 B: Smartphone** Shop.
- Füge ein **`<picture>`** Element ein.
- Erstes Bild: **`phone1.jpg`**
- Zweites Bild: **`phone2.jpg`** (kleiner-gleich 800 Pixel Browserbreite).
- Drittes Bild: **`phone3.jpg`** (kleiner gleich 600 Pixel Browserbreite).
- **`phone3.jpg`** soll auch das Fallback-Bild sein!
- Setzte eine Sprungmarke/Anker vom Produkte Button auf
  das **`<picture>`** Element.

11.3 Viewport

Natürlich ist es unumgänglich, die Webseiten auch für Mobile Devices tauglich zu machen. Die Webseiten sollen auf Smartphones oder Tablets ebenso schön sein, wie bei einer Browserdarstellung. Dabei gibt es zwei Probleme. Erstens, haben Mobile Devices eine beträchtlich größere Pixeldichte (Auflösung) als ein herkömmlicher Bildschirm und zweitens sind ihre Screens viel kleiner. Das führt dazu, das der Mobile Browser die Webseite zwar vollständig darstellen kann, aber Schriften, Bilder usw. sind dann sehr klein. Um dem entgegen zu wirken, kann man den Viewport definieren!

Dieser wird im **<head>** *als* **<meta>** *Tag angegeben. Das Beispiel sagt den mobilen Geräten, dass die Seite mit einer Breite von 1024 Pixel dargestellt werden soll:*

```
<head>
   <meta name="viewport" content="width=1024">
</head>
```

Responsive Web Design

Responsive Web Design (RWD) ist ein Design und Layout-Konzept um Webseiten auf allen Geräten gut darzustellen. Es werden HTML Elemente in seiner Größe verändert, versteckt, ein- oder ausgeblendet und an eine andere Position gestellt. Dabei sollte nur HTML und CSS verwendet werden!

Für Mobile Devices setzen wir den Viewport auf die Screen-Breite des Gerätes:
(Der Meta-Tag sollte auf jeder Website eingetragen werden!)

```
<meta name="viewport" content="width=device-width">
```

Im Attribut **content** *des* **<meta>** *Tags, können auch das Zoom Level (***initial-scale***) bestimmt werden und ob der/die Nutzer/in die Seite zoomen kann (***user-scalable***).*

```
<meta name="viewport"
 content="width=device-width, initial-scale=1.0, user-scalable=no">
```

Der_die User_in soll auf den mobilen Geräten zwar weiterhin vertikal scrollen, jedoch nicht horizontal. Ebenso soll ein Zoomen nicht notwendig sein, um die Inhalten gut zu lesen bzw. betrachten. Dafür ist folgendes zu beachten:

Verwende keine Elemente mit fixen Größenangaben.
Wo immer es möglich ist sollte statt **px** *besser* **%** *verwendet werden. Schriftgrößen sind in* **em** *oder ausgeschrieben (***medium***) empfehlenswerter als in* **pt** *oder* **px***. usw.*

Inhalte sollen nie auf eine bestimmte Darstellungsbreite angewiesen sein, um gut dargestellt zu werden.
Wir müssen davon ausgehen, dass die Webseite sowohl auf kleinen Screens als auch auf extrem großen Screens aufgerufen werden kann. Auch ist nicht sicher, ob ein Browser immer im Vollbildmodus geöffnet ist. Deshalb sollte man die Elemente am besten im Fluss lassen – wo immer es möglich ist.

Verwende Medienabfragen (media queries) um das HTML Dokument für kleine und große Screens zu optimieren.

Übung A: Smartphone Shop IV

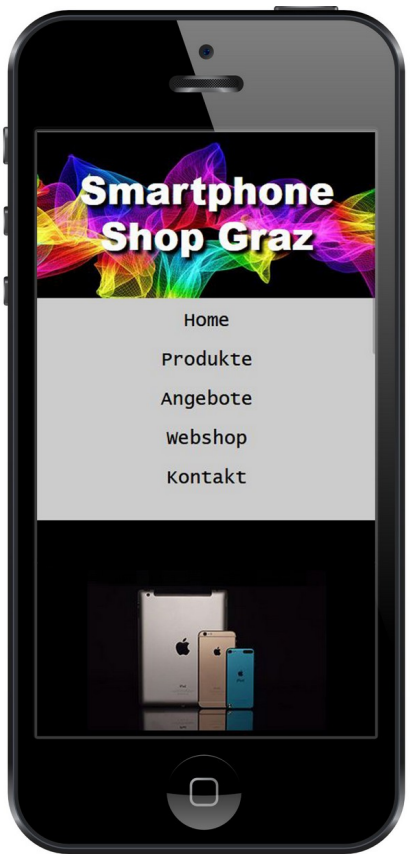
- Öffne dein HTML Dokument **`index.html`** aus der Übung **11.2 A: Smartphone Shop III**.
- Bette das PDF-File **`Aktion.pdf`** ins Dokument ein.
- Setzte eine Sprungmarke/Anker vom **Angebote Button** auf das PDF.
- Füge folgende Adress-Informationen hinzu:

Smartphone Shop Graz
Jakoministraße 15, A-8010 Graz
Telefon: 0316 55587
eMail: shop@smartgraz.at

- Füge noch ein Google Maps `iframe` mit der Adresse oben hinzu und setze eine Sprungmarke vom Kontakt Button auf die Adress-Informationen.

Übung B: Smartphone Shop V

- Öffne dein HTML Dokument **`index.html`** aus der Übung **11.3 A: Smartphone Shop IV**.
- Die Webseite soll für mobile Geräte optimiert werden (Viewport, Medienabfragen).
- **Die horizontale Navigation soll auf mobilen Geräten vertikal dargestellt werden!**



Übung C: Smartphone Shop VI

- Öffne dein HTML Dokument **`index.html`** aus der Übung **11.3 B: Smartphone Shop V.**
- Blende die Navigation für mobile Geräte aus.
- **Füge einen Button hinzu, mit dem man das Navigationsmenü wieder einblenden kann.**

Übung D: Smartphone Shop VII

- Öffne dein HTML Dokument **`index.html`** aus der Übung **11.3 C: Smartphone Shop VI**.
- Analysiere die Probleme, die sich für diese Webseite ergeben.
- Verbessere und viel wichtiger, verschönere die Webseite!
- **Freie kreative Übung!**

11.4 Box Eigenschaften

Schatteneffekte für Boxen

CSS

`box-shadow: [X] [Y] [blur] [spread] [color];`

`[X]`	Horizontaler Versatz des Schattens. `20px` verschiebt den Schatten um 20 Pixel nach rechts. `-10px` verschiebt den Schatten um 10 Pixel nach links.
`[Y]`	Vertikaler Versatz des Schattens `20px` verschiebt den Schatten um 20 Pixel nach unten. `-15px` verschiebt den Schatten um 15 Pixel nach oben.
`[blur]`	Radius der Unschärfe (Weichzeicheneffekt). `0px` entspricht einer scharfen Darstellung.
`[spread]`	Zusätzliche Vergrößerung des Schattens. Bei `0px` hat der Schatten die gleiche Größe wie sein Element.
`[color]`	Farbangabe

Weitere Attribute von box-shadow:

`inset`	wirft den Schatten ins Innere der Box. `Inset` darf nur am Anfang oder am Ende der Attribute gesetzt werden. ohne Wert
`none`	kein Schatten

```
.bilder {box-shadow: 10px 10px 5px 0px gray;}
    div {box-shadow: inset -10px -10px 20px 10px black;}
```

Anzeigen von Inhalten: `overflow`

Die HTML Elemente sollten so gestaltet werden, dass sie problemlos in der Eltern-Box Platz finden. Sollte es aber vorkommen, dass der Inhalt größer ist als die Eltern-Box, kann man mit der `overflow` Eigenschaft das Verhalten näher bestimmen.

CSS

`overflow: [value];`

Für `value` stehen folgende Attribute zur Verfügung:

`auto`	Browserabhängig
`scroll`	Scrollbalken werden angezeigt
`visible`	Inhalte sind sichtbar und ragen über das Elternelement hinaus.
`hidden`	Inhalte werden an den Rändern unsichtbar (abgeschnitten).

Zusätzlich gibt es die CSS Eigenschaften `overflow-x` und `overflow-y` für horizontale und vertikale Inhalte die über die Box hinausragen.

```
div {overflow: hidden;}
```

Mit der CSS Eigenschaft `resize` erlaubt man dem_der Nutzer_in das Verändern der Größen eines Elements.
Folgende Werte sind möglich: `both; horizontal; vertical; none;`

```
div {resize: both;}
```

12.1 Formulare

Zahlreiche Anwendungen lassen sich mit Formularen in HTML verwirklichen (z. B. Mail-Formulare). Formulare bieten die Möglichkeit zur Eingabe von Informationen – die Verarbeitung dieser Informationen erfolgt dann durch eine clientseitige Scriptsprache (z. B. JavaScript) oder einer serverseitigen-Sprache (z. B. PHP). Die Formularelemente werden in einem **<form>** *Tag platziert und mit einem* **<label>** *Tag beschriftet.*

HTML

<form action="…" method="…" name="…"> ... </form>

action	Im **action** Attribut wird eine URL eingetragen. In den meisten Fällen ist das die Adresse zu einem Script. **action="mailscript.php"** Wird das **action** Attribut weggelassen, dann werden die Daten an das aktuelle Dokument gesendet.
method	Legt die Methode fest, wie die Daten versendet werden sollen. Es gibt zwei mögliche Werte: **get** Die Daten werden sichtbar über die URL-Zeile des Browsers weitergegeben. **post** Die Übertragung erfolgt unsichtbar im Hintergrund.
name	Definiert einen Namen für das Formular. Dieser ist meist notwendig für die weitere Verarbeitung mit z. B. JavaScript oder PHP.

Weitere Attribute:

target	Zielfenster
accept-charset	Angabe einer Zeichenkodierung
enctype	Angabe der Datenkodierung, Standardwert ist text/plain
autocomplete	Mit den Werten **on** oder **off** kann die Autovervollständigung des Browsers aktiviert, bzw. deaktiviert werden.
novalidate	Das Formular soll nicht auf Vollständigkeit geprüft werden.

```
<form action="mailversenden.php" method="get" > ... </form>
```

HTML

<label > ... </label>

Das **<label>** Element hat zwei Vorteile. Erstens erleichtert es einen Screenreader das assoziierte Formelement vorzulesen. Zweitens wird bei einem Mouse-Click auf den Text innerhalb eines **<label>** Tags ein Focus auf das Formelement gelegt.

Mit dem Attribut **for=""** bezieht sich das **<label>** Element auf ein bestimmtes Formularelement (z. B. input) mit gleicher id.

```
<form action="senden.php" name="person" id="meinForm"
   autocomplete="off" method="post">
  <label for="vorname">Vorname</label>
  <input type="text" name="vorname" id="vorname" maxlength="30">
  <button type="submit">Eingaben absenden</button>
</form>
```

12.2 Textarea

*Über das **<textarea>** Element kann der Benutzer mehrzeilige Texteingaben tätigen.*

HTML

<textarea name="" rows="" cols=""> ... </textarea>

name	Eine Bezeichnung für die weitere serverseitige Verarbeitung.
rows	Anzahl der Zeilen (bestimmt damit auch die Höhe)
cols	Anzahl der Spalten bzw. die Anzahl der Zeichen pro Zeile

```
<form action="feedback.php" method="post" name="FB1" id="FB1">
  <h1>Rückmeldung</h1>
  <label for="TF1">Haben Sie noch weitere Vorschläge?</label><br>
  <textarea name="TF1" id="TF1"
            rows="5" cols="50">Ihr Feedback...</textarea>
</form>
```

Weitere Attribute sind:

wrap	Bestimmt den Textumbruch. Mögliche Werte sind: **hard** oder **soft**.
readonly	Das Eingabefeld kann nur gelesen werden. Eine Eingabe ist also nicht möglich. ohne Wert
required	Eine Eingabe muss erfolgen um das Formular absenden zu können. ohne Wert
placeholder	Es kann ein Text bestimmt werden, welcher in der Textarea als Hinweis erscheint.
autofocus	Legt den Focus nach dem Laden der Seite auf das Textfeld. ohne Wert
maxlength	Maximale Länge der Eingabe (Anzahl der maximalen Buchstaben).

```
<textarea name="tear2" id="tear2"
 rows="4" cols="60" placeholder="Ihr Feedback ..."
 required autofocus maxlength="220" ></textarea>
```

*Da das **<textarea>** Element kompromisslos alle Sonderzeichen, Leerzeichen und Zeilenschaltungen darstellt, kann es auch für die Darstellung von HTML Code verwendet werden.*

```
<textarea rows="4" cols="100" readonly >
  <h1>Technisches Service</h1>
  <p>Hier finden Sie alle ...</p>
</textarea>
```

*Das gleiche Ergebnis innerhalb eines **<pre>** Tags sieht dann so aus:*

```
<pre>
  &lt;h1&gt;Technisches Service&lt;/h1&gt;
  &lt;p&gt;Hier finden Sie alle ...&lt;/p&gt;
</pre>
```

12.3 Input

Mit dem **<input>** *Element können einzeilige Eingaben getätigt werden. Es ist sehr umfangreich und wird durch das* **type** *Attribut näher bestimmt. Das* **<input>** *Element ist ein Standalone-Tag und muss also nicht geschlossen werden.*

HTML

<input type="" name="" value="">

type	Beschreibt die Art des Eingabefelds. Als Standard wird **type="text"** angenommen.
name	Der Name des Eingabefelds wird für die spätere Verarbeitung in einer Script- oder Programmiersprache verwendet.
value	Ein vordefinierter Wert.

```
<form>
  <label for="Vorname" >Ihren Vorname?</label><br>
  <input type="text" name="Vorname" id="Vorname" value="Kevin">
</form>
```

Weitere generelle Attribute

size	Definiert die Anzahl der Zeichen, die eingeben werden können. z. B. bei **size="15"** hat das Feld eine Breite von 15 Zeichen.
readonly	Das Eingabefeld kann nur gelesen werden. Eine Eingabe ist nicht möglich. ohne Wert
min	Der Minimum-Wert bei numerischen oder Datumseingabefeldern. z. B. **min="200"** bedeutet eine Eingabe von mehr als 200.
max	Der Maximum-Wert bei numerischen oder Datumseingabefeldern. z. B. **max="1300"** bedeutet eine Eingabe von weniger als 1300.
step	Die Schrittweite der Eingabe bei numerischen oder bei Datumsfeldern. **step="any"** hebt die Schrittweite auf.
multiple	Erlaubt Mehrfacheingaben für Datei-Uploads oder Email-Eingaben. ohne Wert

Attribute zur Validierung und Gestaltung

minlength	Mindestlänge der Eingabe (in Zeichen) z. B. **minlength="5"** verlangt mindestens fünf Zeichen.
maxlength	Maximallänge der Eingabe (in Zeichen).
required	Pflichtfeld. Eine Eingabe muss erfolgen. ohne Wert
placeholder	Sichtbarer Hinweis im Inputfeld. Es verschwindet nach der Eingabe eines Zeichens.
autofocus	Nach dem Laden der Webseite, wird er Focus auf das Inputfeld gelegt. ohne Wert
autocomplete	Automatisches Vervollständigen von Formularfeldern durch den Browser. Als Werte sind **on** oder **off** möglich.

```
<input type="text" name="Nachname"
       size="20" maxlength="20" minlength="3"
       placeholder="Bitte Nachnamen eingeben!" required
       autocomplete="off" autofocus >
```

Ü 12.3 Übungen Input

Übung A: `get` und `post`

- Recherchiere im Internet den Unterschied zwischen **`get`** und **`post`** als Methode für ein Formular.
- Wie funktioniert **`get`**? Wie funktioniert **`post`**?
- Welche Vor- und Nachteile haben **`get`** und **`post`**?
- **Finde jeweils mind. drei Beispiele wo `get` bzw. `post` Sinn ergibt.**

Übung B: Kommentar-Formular

- Erstelle ein neues HTML-Dokument mit einem Formular für Kommentare. Das Formular hat ein einzeiliges Texteingabefeld für den Vor- und Nachnamen und ein mehrzeiliges Textfeld für den Kommentar.
- **`<form>` Eigenschaften:**

  Das Formular wird im **`comment.php`** Script verarbeitet. Eine passende Methode ist selbst zu wählen. Beschrifte das Formular (**`<label>`**) mit "**Ihr Feedback**"
- **Vor- und Nachnamen (einzeilige Texteingabe)**

  Die Attribute für den Namen und Größe sind selbst zu wählen. Es ist ein Pflichtfeld mit einem Platzhaltertext "**Hier bitte den Namen eingeben**". Die Beschriftung lautet: "**Vor- und Nachname**".
- **Kommentarfeld (mehrzeilige Texteingabe)**

  Die Attribute für den Namen und Größe sind selbst zu wählen. Die maximale Länge der Eingabe ist auf 600 Zeichen beschränkt. Es ist ein Pflichtfeld. Die Beschriftung lautet: "**Kommentar**".
- CSS Anpassung für die Beschriftungen:

  Schriftart: Arial, Kursiv.
- CSS Anpassung für die Textfelder:

  Schriftart: Courier New | Hintergrundfarbe: hellblau | Rahmen: 2 Pixel Dunkelgrau. | Wenn die Maus über das Textfeld fährt, soll sich die Hintergrundfarbe in weiß ändern

12.4 Input Typen

Texteingabe

HTML

```
<input type="text">
```

Passwörter

HTML

```
<input type="password">
```

Im Eingabefeld werden Platzhalter anstatt der Zeichen angezeigt.
(Browserabhängig, meist aber Sterne *)

eMail-Adressen

HTML

```
<input type="email">
```

Eine Eingabe ist nur valide, wenn auch ein @ Zeichen darin vorkommt.

Web-Adressen

HTML

```
<input type="url">
```

Validiert die URL. Eine vollständige Eingabe mit http ist notwendig:
(z. B. http://www.css4.at) Mobile Geräte blenden z. B. .com automatisch in der Tastatur ein.

Für die Typen `text`, `password`, `email` *und* `url` *gibt es das Attribut* `inputmode`. *Es hilft den mobilen Geräten die passende Tastatur zu wählen.*
Werte sind z. B. `verbatim` *(alphanumerisch),* `latin` *(Texteingabe mit Texteingabehilfe) usw.*

Eingabe von Zahlen

HTML

```
<input type="number">
```

Es können nur Zahlen eingegeben werden.
Weitere Attribute sind `max`, `min`, `step` und `value`.

Eingabe von Zahlen mit einem Schieberegler

HTML

```
<input type="range">
```

Der Schieberegler (Slider) wird für die Eingabe von Zahlen verwendet.
Weitere Attribute sind `max`, `min`, `step` und `value`.

Mit dem `orient` Attribut kann man den Slider auch vertikal darstellen.
z. B. `<input type="range" orient="vertical">`

Telefonnummern

HTML

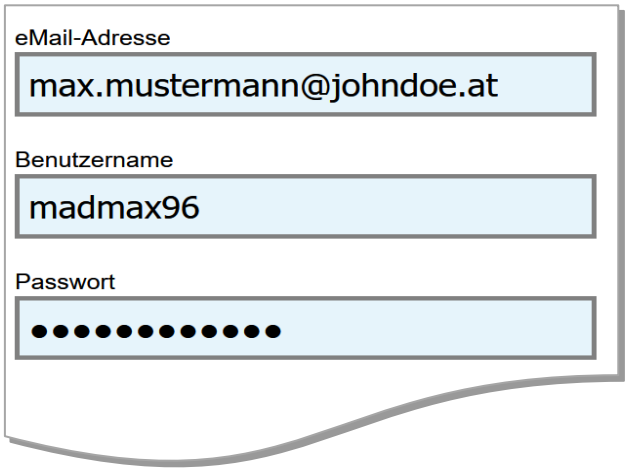
```
<input type="tel">
```

Mobile Geräte blenden eine angepasste Tastatur ein.

Ü 12.4 Übungen Input Typen

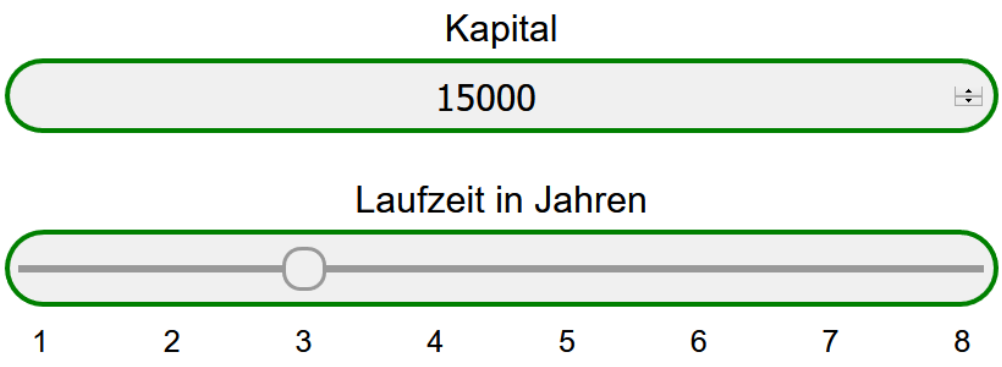
Übung A: Anmeldeformular

- Erstelle ein typisches Anmeldeformular mit den Feldern eMail, Benutzername und Passwort.
- Beschrifte alle Felder mit dem **<label>** Element.
- **Verschönere das Aussehen mit CSS. Freie kreative Übung.**



Übung B: Online Sparen

- Erstelle ein Formular zur Berechnung von Sparzinsen für ein Online-Sparbuch.
- Zwei Eingabefelder
- Feld 1: Kapital
  Nummernfeld, mindestens 1000 und maximal 50000. Schrittweite 500. Beschriftung durch ein **<label>** Element.
- Feld 2: Laufzeit in Jahren
  Schieberegler, mindestens 1 und maximal 8, Schrittweite 1. Beschriftung durch ein **<label>** Element.
- **Verändere das Aussehen mit CSS.**



12.5 Form Button

HCL125

Für Formulare gibt es das **<button>** *Element, welches anklickbare Schaltflächen erzeugt. Diese können ein Formular absenden, das Formular zurücksetzten oder eine clientseitige Aktion/Funktion (z. B. für JavaScript) auslösen.*

HTML

<button type="" > ... </button>

Zwischen dem Start und Endtag wird die Beschriftung des Buttons eingetragen.
Das Aussehen kann selbstverständlich mit CSS angepasst werden.
Für das **type** Attribut gibt es folgende Werte:

button	Auslösen einer clientseitigen Aktion
submit	Absenden eines Formulars, Standardwert
reset	Zurücksetzen des Formulars.

Passwort eingeben: [ ] Komplexität prüfen | Zurücksetzen | Absenden

```
<form>
   <label for="eingabetext">Passwort eingeben:</label>
   <input type="password" name="eingabetext" id="eingabetext">
   <button type="button">Komplexität prüfen</button>
   <button type="reset">Zurücksetzen</button>
   <button type="submit">Absenden</button>
</form>
```

Weitere Attribute:

form	Wenn der Button außerhalb des Formulars ist, kann mit dem form Attribut der Name des Formulars angegeben werden.
formaction	Im Button kann die Aktion (URL) des Formulars festgelegt werden. Sollte es im **<form>** Element eine abweichende Methode geben, dann gilt die des **<buttons>**
formmethod	Im Button kann die Methode **(Get, Post)** definiert werden. Sollte es im **<form>** Element eine abweichende Methode geben, dann gilt die des **<buttons>**
formenctype	Codierung über den Button
name	Zur weiteren Verwendung in einem Script
value	Rückgabewert eines Buttons.

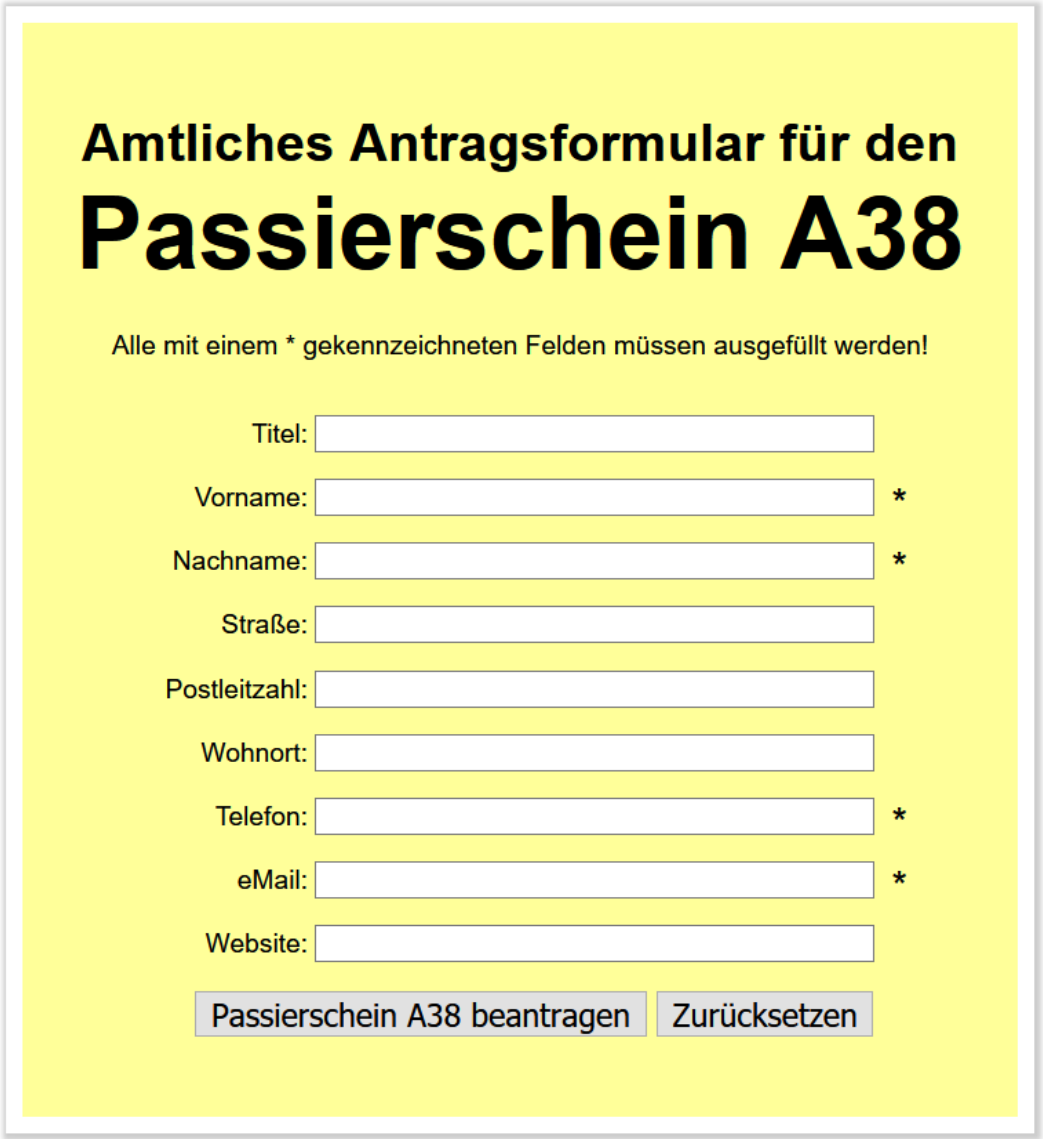
```
<form name="meinForm" id="meinForm">
   <input type="text" name="Feld01" id="Feld01">
</form>

<button form="meinForm" formaction="standard.php"
        formmethod="get" type="submit">Standard</button>
<button form="meinForm" formaction="premium.php"
        formmethod="post" type="submit">Premium</button>
```

Mit dem Attribut **formnovalidate** *kann die Überprüfung von Pflichtfeldern (mit* **require** *gekennzeichnet) unterbunden werden. Es ist analog zum* **novalidate** *Attribut im* **<form>** *Element.*

Ü 12.5 Übungen Formular Buttons

Übung A: Passierschein A38

- Erstelle ein Formular wie unten dargestellt.
- Wähle zwischen zwei Layoutlösungen für das Formular:
  - mit Tabelle (einfach, aber mehr Schreibarbeit)
  - ohne Tabelle (knifflig, aber dafür weniger Schreibarbeit)
- Versuche mit CSS das gleiche Erscheinungsbild wie unten zu erzeugen.
- Die Felder Vorname, Nachname, Telefon und eMail sind Pflichtfelder.
- **Zwei Buttons zum Absenden und zum Zurücksetzten des Formulars.**



12.6 Dialogeingaben

Datei-Upload

HTML

<input type="file" >

Der Datei-Upload funktioniert nur mit der **Post**-Methode (im **<form>** oder im **<button>**). Zusätzlich muss im **<form>** Element noch das Attribut **enctype="multipart/form-data"** hinzugefügt werden, denn sonst wird nur der Name der Datei übertragen. Weitere Attribute:

accept	Definiert die möglichen Dateitypen (MIME-Typen) für den Upload. z. B. nur Textdateien: **<input type="file" accept="text/*">** nur JPG Bilder: **<input type="file" accept="image/jpeg">**
multiple	Mehrfachauswahl ohne Wert

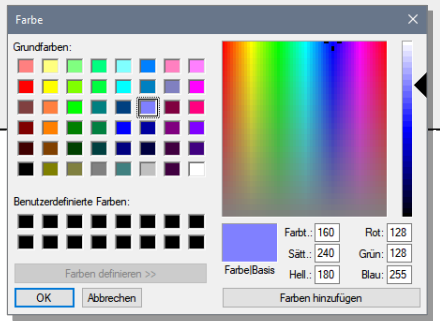
```
<form action="upload.php" enctype="multipart/form-data" method="post">
 <label for="datei" >Bilder hochladen</label>
 <input type="file" accept="image/*" name="datei" id="datei" multiple>
</form>
```

Farbauswahl

HTML

<input type="color" >

Der Rückgabewert der ausgewählten Farbe ist ein HEX-Wert.




```
<form method="get">
  <label for="farbe" >Lieblingsfarbe</label>
  <input type="color" value="#ffffff"
         id="farbe" name="farbe" >
</form>
```

Datum und Uhrzeit

HTML

<input type="date" >

Mit dem **type="date"** Attribut werden Datumseingaben abgefragt. In der Regel stellt der Browser ein Datumsfeld oder ein Kalender-Dialog zur Verfügung.

Artverwandt sind:

Wochenangaben:	**<input type="week" >**
Monatsangaben:	**<input type="month" >**
Uhrzeiten:	**<input type="time" >**
Kombination:	**<input type="datetime-local" >**

Versteckte Elemente

HTML

<input type="hidden" value="" >

Das Eingabefeld ist für den Benutzer unsichtbar. Der Wert wird aber beim Absenden übergeben.

Übung A: Farbauswahl

- Erstelle ein neues HTML Dokument.
- Füge ein Eingabefeld für Farben hinzu.
- Beschrifte das Eingabefeld mit "Farbe der Mannschaft". `<label>`
- **Verwende als Methode `GET` mit dem `Report_GET.html` über den Submit-Button und nicht als Attribut im `<form>` Element.**

Übung B: Online Bewerbung

- Erstelle ein neues HTML Dokument mit einem Formular zur Online Bewerbung.
- Folgende Eingabefelder sollten eingebunden sein:
  Vor- und Nachname, E-Mail-Adresse, Geburtsdatum, Dateiupload für den Lebenslauf (nur PDF) und ein mehrzeiliges Textfeld für das Motivationsschreiben.
- **Gestalte das Formular mit CSS anspruchsvoll. Freie kreative Übung.**

12.7 Auswahleingaben

Radio-Buttons

HTML

`<input type="radio" name="" value="">`

Radio-Buttons sind eine Gruppe von Auswahlbuttons in der nur eine Eingabe erlaubt ist. Wird also ein Button angeklickt, wird ein anderer abgewählt. Folgende Attribute sind möglich:

`name`	Alle Radio-Buttons mit dem gleichen **`name`** Attributwert, gehören zu einer Gruppe.
`value`	Definiert den Wert, der beim Absenden des Formulars übergeben wird wenn das Element aktiviert wurde.
`checked`	Ein bestimmter Radio-Button wird vorselektiert. ohne Wert
`disabled`	Zum Deaktivieren. Es wird ausgegraut dargestellt. ohne Wert

```
<form>
  <input type="radio" name="Zahlung" value="Nachnahme" id="nn">
  <label for="nn"> per Nachnahme </label>
  <input type="radio" name="Zahlung" value="Vorauskassa" id="vk">
  <label for="vk"> per Vorauskassa </label>
</form>
```

Checkbox

HTML

`<input type="checkbox" name="">`

Eine Checkbox ist ein anklickbares Kontrollfeld. Per Klick erscheint in der Regel ein Häkchen bei den ausgewählten Feldern.

`name`	Jedes Checkbox Element sollte einen eigenen Namen bekommen.
`checked`	Zum Vorselektieren von Checkboxen. ohne Wert
`disabled`	Zum Deaktivieren. Es wird ausgegraut dargestellt. ohne Wert

```
<form method="get">
    <input type="checkbox" name="abo1" id="gabo">
    <label for="gabo"> Gratis Ausgabe bestellen</label><br>
    <input type="checkbox" name="newsletter" id="nwsl" checked>
    <label for="nwsl"> Newsletter bestellen</label>
    <button type="submit">Absenden</button>
</form>
```

Fieldset

HTML

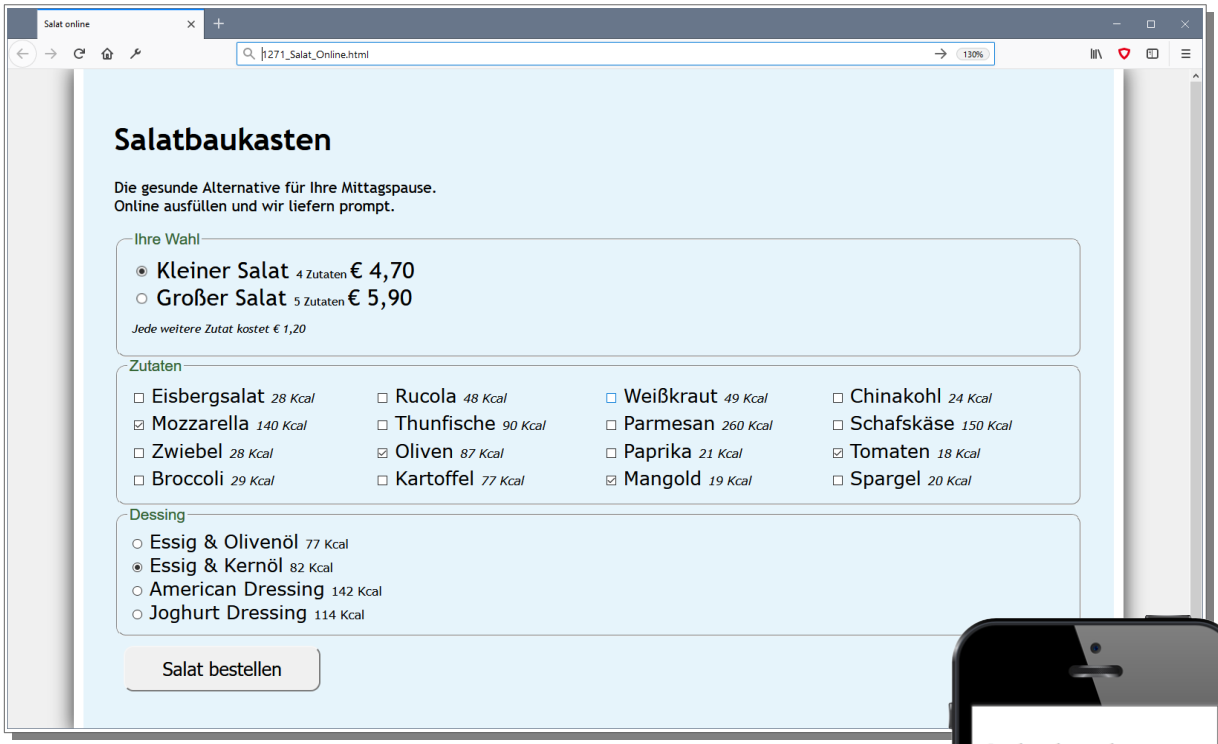
`<fieldset> <legend> … </legend> … </fieldset>`

Mit dem **`<fieldset>`** Element kann man Formelemente **innerhalb** eines Formulars gruppieren. Es sind die Attribute **`name`**, **`disabled`** und **`form`** verfügbar. Das **`<legend>`** Element wird zur Beschriftung des **`<fieldset>`** verwendet.

```
<fieldset name="Verpackung">
   <legend>Welche Verpackung möchten Sie?</legend>
   <input type="checkbox" name="Oeko"> Öko <br>
   <input type="checkbox" name="Premium"> Premium
</fieldset>
```

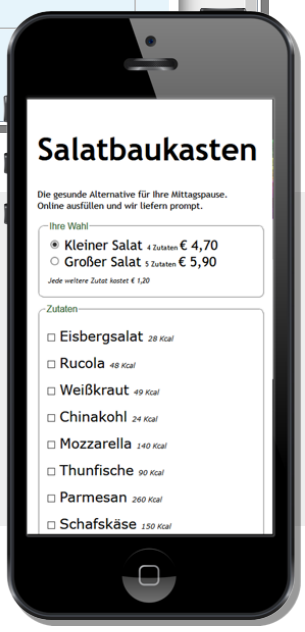
Übung A: Salat Online

- Erstelle ein Bestellformular für einen Salat Lieferbetrieb.
- Auswahl zwischen einem großen und einem kleinen Salat.
- Mindestens 12 Zutaten mit Kalorienangaben (Mehrfachauswahl).
- Mindestens 4 Dressings mit Kalorienangaben (nur eine Auswahl möglich).
- Verwende **`<fieldset>`** Elemente zur optischen Trennung.
- Beschrifte alle Felder mit **`<label>`**
- Gestalte die Seite mit CSS – freie kreative Übung.
- **Teste die Seite mit der `Report_GET.html`**



Übung B: Salat Online Mobil

- Öffne deine Lösung von **Übung 12.7 A: Salat Online.**
- Optimiere die Seite für mobile Geräte (Smartphone, Tablet usw.)
- **Teste die Seite auf einem Smartphone!**



Drop-Down-Liste

HTML

<select name="" > <option> ... </option> </select>

Das **<select>** Element stellt eine Auswahlliste, bzw. ein Drop-Down-Feld zur Verfügung. Innerhalb des **<select>** Elements werden über den **<option>** Tag die Einträge festgelegt.

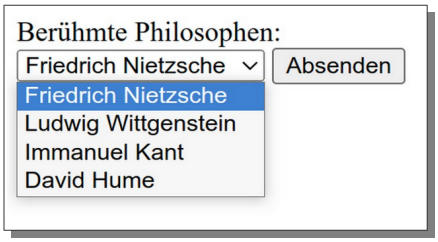
Attribute für das <select> Element.

name	Name des Elements (Variablenname).
size	Anzahl der sichtbaren Zeilen. Ohne Size Attribut wird ein Drop-Down-Feld angezeigt.
multiple	Ermöglicht eine Mehrfachauswahl ohne Wert
required	Die Eingabe ist erforderlich (Pflichtfeld) ohne Wert

Attribute für das <option> Element.

value	Übergabewert, wenn es angeklickt wird. Ohne Wert, wird der Inhalt des **<option>** Tag übermittelt.
selected	Eine Option wird vorausgewählt. ohne Wert

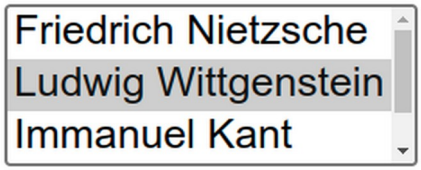
```
<form action="Report_GET.html" method="get">
   <label for="philos">Berühmte Philosophen:</label><br>
    <select name="philos" id="philos">
      <option>Friedrich Nietzsche</option>
      <option>Ludwig Wittgenstein</option>
      <option>Immanuel Kant</option>
      <option>David Hume</option>
    </select>
  <button type="submit">Absenden</button>
</form>
```



Auswahlliste (sichtbare Auswahl)

Hier eine Auswahlliste mit drei Zeilen und **value** *Attribute für das* **<option>** *Element. Das Auswahlfeld* **value="2typ"** *wurde vorselektiert.*

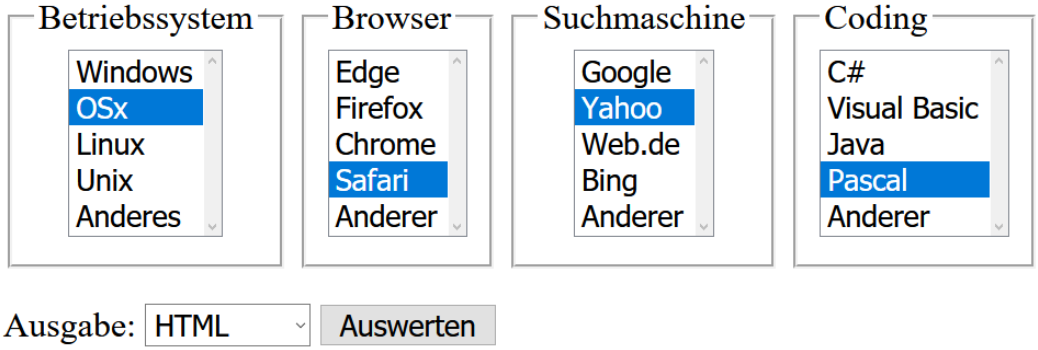
```
  <select name="philos" id="philos" size="3">
    <option value="1typ">Friedrich Nietzsche</option>
    <option value="2typ" selected>Ludwig Wittgenstein</option>
    <option value="3typ">Immanuel Kant</option>
    <option value="4typ">David Hume</option>
  </select>
```



Eine Mehrfachauswahl (mit dem **multiple** *Attribut) ist für den Anwender nicht immer klar ersichtlich (z. B. mit gedrückter STRG-Taste selektieren). Checkboxen eigenen sich besser für eine Mehrfachauswahl.*

Übung A: Softwareliste

- Erstelle ein Formular wie unten dargestellt.
- Vier Auswahllisten.
- Fieldsets als Beschriftung
- **Ein Drop-Down-Feld (Ausgabe: HTML, XML, Plain Text).**



Übung B: Gewinnspiel

- Erstelle eine Webseite für ein Gewinnspiel
- Mindestens drei Radio-Buttons
- Drei Textfelder für den Namen, eMail-Adresse und Telefon.
- Ein Drop-Down-Feld
- Mindestens zwei Checkboxen.
- **Freie kreative Übung.**

13.1 Animationen

Mit CSS ist es möglich, Animationen ganz ohne Flash oder JavaScript zu erstellen. Dabei werden die Start-Eigenschaften eines Elements bis zu den End-Eigenschaften verändert. Der Übergang verläuft flüssig. Für die Animation wird zuerst eine **`@keyframe`** *Regel erstellt.*

CSS

`@keyframes` `meineAnimation{ ... }`

Im folgenden Beispiel ist **`meineAnimation`** der Name der Animation. Natürlich kann auch ein anderer Name gewählt werden. Innerhalb der **`@keyframes`** Regel wird die Start-Eigenschaft mit **`from {…}`** und die End-Eigenschaften mit **`to {…}`** festgelegt. Die Farbe ändert sich von Blau nach Rot!

```
@keyframes meineAnimation {
   from {background-color:blue;}
   to   {background-color:red;}
     }
```

Nachdem die Animation über die **`@keyframes`** *Regel erstellt wurde, muss sie über einen Selektor (*`ID, Class`*, usw) dem Element zugewiesen werden.*
Dafür wird mit **`animation-name`** *der Name der Animation angegeben und mit* **`animation-duration`** *die Dauer der Animation festgelegt. Nach der Animation nimmt das Element wieder seinen originalen Style an.*

CSS

`animation-name: [Name];`

Für **`[Name]`** wird der Namen der **`@keyframes`** Regel eingetragen.
In diesem Beispiel: **`animation-name: meineAnimation;`**

SS

`animation-duration: [Value];`

Für **`[Value]`** wird die Dauer der Animation in Sekunden eingetragen.
In diesem Beispiel vier Sekunden: **`animation-duration: 4s;`**

```
div {width: 500px; height: 500px;
     background-color:red;
     animation-name: meineAnimation;
     animation-duration: 4s;}
```

Das Beispiel verwandelt alle **`<div>`** *Elemente vom Quadrat in einen Kreis. Dauer der Animation ist 5 Sekunden.*

```
@keyframes rundeEcken {
     from {border-radius:0px;}
     to   {border-radius: 500px;}
           }
     div {width: 200px; height: 200px; margin:auto;
          border: 20px solid red; border-radius: 500px;
          animation-name: rundeEcken;
          animation-duration: 5s;}
```

13.2 Animationen II

*Die **@keyframes** Regel ist nicht nur auf ein **from** und **to** beschränkt. Die einzelnen Animatinsschritte können auch mit Prozentangaben definiert werden.*
(von 0% bis 100%).

CSS **@keyframes animationsName { 0 - 100%{…} }**

```
@keyframes bewegung {
      0%  {top: 0px; left: 0px; background: red;}
     25%  {top: 0px; left: 200px; background: blue;}
     50%  {top: 200px; left: 200px; background: yellow;}
     75%  {top: 200px; left: 0px; background: green;}
    100%  {top: 0px; left: 0px; background: red;}
          }
```

Verzögerung

CSS **animation-delay: [Wert];**

Die Eigenschaft bestimmt die Verzögerung des Animationsstarts.
`animation-delay: 3s;` wartet drei Sekunden bis die Animation startet.

```
#dasDIV {width:200px; height:200px; background:green;
         position:fixed;
         animation-name:bewegung; animation-duration:3s;
         animation-delay:5s;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  <div id="dasDIV">Warte 5 Sekunden</div>
```

Wiederholungen

CSS **animation-iteration-count: [Wert];**

Die Eigenschaft bestimmt die Wiederholungen der Animation.
`animation-iteration-count: 4;` wiederholt die Animation viermal.
`animation-iteration-count: infinite;` wiederholt ohne Ende.

```
#dasDIV {width:200px; height:200px; background:green;
         position:fixed;
         animation-name:bewegung; animation-duration:3s;
         animation-iteration-count:3;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  <div id="dasDIV">Drei Wiederholungen</div>
```

Zustand

CSS **animation-play-state: ;**

Bestimmt ob eine Animation pausiert oder läuft.
`animation-play-state: paused;` Die Animation pausiert.
`animation-play-state: running;` Die Animation läuft (Standardwert)

```
         #dasDIV:hover {animation-play-state:paused;}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
         <div id="dasDIV">Stoppe mich mit der Maus</div>
```

13.3 Animationen III

Richtung der Animation

CSS

`animation-direction: [Wert];`

`[Wert]` Wechselt bei jedem Durchgang (Zyklus) seine Richtung.

`normal`	Die Animation wird immer vorwärts abgespielt.
`alternate`	Die Animation wechselt nach jedem Durchlauf die Richtung.
`reverse`	Die Animation wird immer rückwärts abgespielt.

Zustand der Animation

CSS

`animation-fill-mode: [Wert];`

`[Wert]` Definiert den Zustand nach der Animation.

`forwards`	Das Element übernimmt die Werte des letzten Animationsschrittes.
`backwards`	Das Element übernimmt die Werte des ersten Animationsschrittes
`both`	Verbindet die Werte von forwards und backwards.

Dauer zwischen den Schlüsselbildern

CSS

`animation-timing-function: [Wert];`

`[Wert]` Die Timingfunktion bezieht sich auf die Dauer zwischen den Schlüsselbildern. **`.ease, ease-in, ease-out, ease-in-out, linear, step-start, step-end, cubic-bezier(), steps() usw.`**

Shorthand Propertie animation (Kurzschreibweise)

CSS

`animation: ... ;`

Die einfache CSS Eigenschaft **`animation`** fasst alle anderen animation-Eigenschaften zusammen.

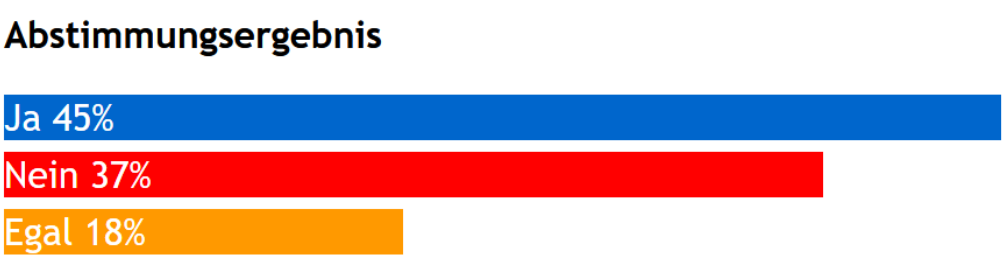
Das Beispiel lässt ein Element aus der Bildschirmmitte hineinfliegen!

```
        #meinDIV {position:fixed; background-color:#CCC;
                  animation: ausZentrum 3s;
                  animation-fill-mode: forwards;}
        @keyframes ausZentrum {
               from {height:1%; width:1%; top:48%; right:48%;}
               to   {height:80%; width:80%; top:10%; right:10%;}
                        }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        <div id="meinDIV">Hinweis</div>
```

Ü 13.3 Übungen Animationen

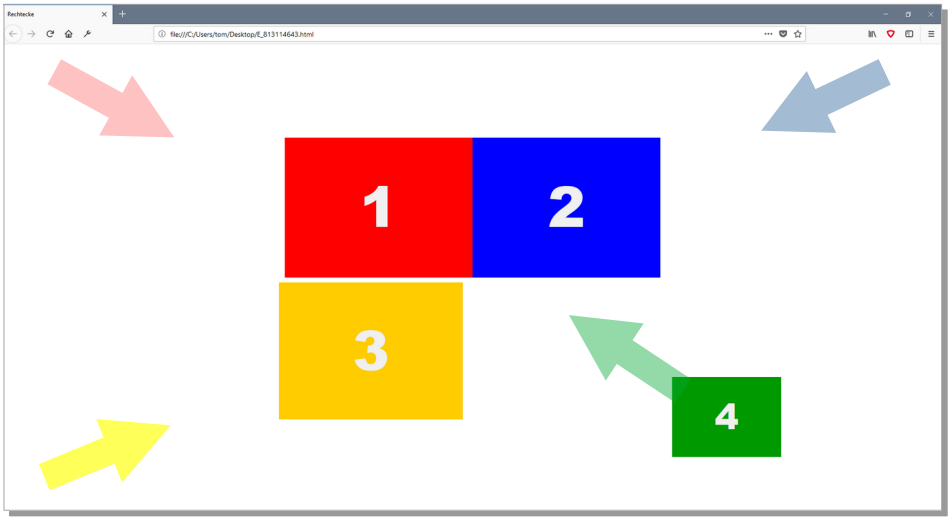
Übung A: Abstimmungsergebnis

- Erstelle ein animiertes Balkendiagramm.
- Die Balken (Ja 45%, Nein 37%, Egal 18%) haben eine unterschiedliche Farbe
- Die Balken wachsen von links nach rechts.
- Die Balken erscheinen hintereinander (Verzögerung).
- **Verschönere das Diagramm mit CSS.**



Übung B: Rechtecke ins Zentrum

- Vier rechteckige Div's sollen im Zentrum angeordnet werden.
- Die Div's dürfen sich nicht überlappen.
- Jedes Div hat eine eigene unterschiedliche Farbe.
- Die Div' fliegen von der Ecke ins Zentrum
  1: von links oben
  2: von rechts oben
  3: von links unten
  4: von rechts unten
- Die Div's sollen auch in Höhe und Breite animiert werden (von klein nach groß).
- **Jedes Div hat eine unterschiedliche Timing-Funktion.**



13.4 Transformieren

Mit der CSS Eigenschaft **`transform`** *lassen sich 2D und 3D Veränderungen bewerkstelligen. Wir werden uns die 2D Veränderungen ansehen, weil damit die meisten 3D Veränderungen auch visualisiert werden können.*

Skalieren

CSS

`transform: scale(x,y);`

Verändert die Größe eines Elements über die x-Achse (Breite) und die y-Achse (Höhe). **`scale(1,1)`** ist die Originalgröße, **`scale(0.5,0.5)`** ist die halbe Größe und **`scale(2,2)`** vergrößert das Element um das doppelte.

Die x und y Achse lässt sich auch direkt ansprechen:
`transform: scaleX(2); transform: scaleX(1.4);`

Für die **`scale(x,y)`** *Funktion sind positive und negative Zahlen (Dezimal und Ganzzahlen) möglich. Bei* **`transform: scale(1,-1)`** *wird das Element gespiegelt.*

Das Beispiel zeigt eine animierte Unterstreichung von der Mitte heraus.

```
   .mitUL {position:relative; display:inline-block;}
   .mitUL:before {content: ""; position: absolute; width: 100%;
                  border-bottom: 3px solid black; bottom: 0; left: 0;
                  animation: meineAniStart 2s;}
    @keyframes meineAniStart {
          from {transform: scaleX(0);}
          to   {transform: scaleX(1);} }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    <h1 class="mitUL">Unsere Abteilung als Profit-Center</h1>
```

Drehen

CSS

`transform: rotate(winkel);`

Dreht ein Element im Uhrzeigersinn. Bei einem negativen Winkel gegen den Uhrzeigersinn.

Der Winkel-Wert wird als Zahl plus **`deg`** angegeben:
`transform: rotate(45deg);`

Andere Winkelmaße sind **`grad`** (Gon), **`rad`** (Radiant), **`turn`** (Vollwinkel).
Bei **`deg`** entspricht eine Kreisumrundung 360°.

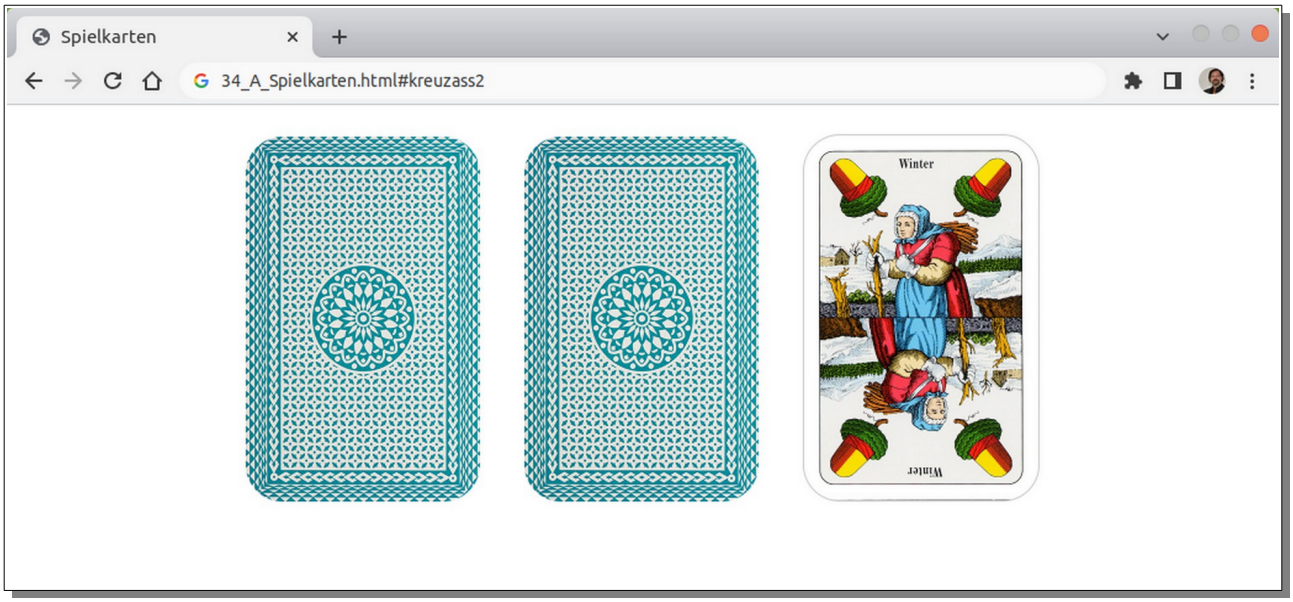
Das Beispiel simuliert einen analogen Sekundenzeiger.

```
        #drehe {animation: meineDrehung infinite 60s linear ;
                display: inline-block; margin: 20%;}
        @keyframes meineDrehung {
            from {transform: rotate(-90deg);}
            to   {transform: rotate(270deg);}    }
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        <div id="drehe">Sekunden &rArr;</div>
```

Ü 13.4 Übungen Transformieren

Übung A: Spielkarten

- Erstelle eine Webseite mit drei Spielkarten.
  (Bilder dazu sind sicher im Internet zu finden).
- Die Spielkarten sind zum Start verdeckt aufgelegt.
- Durch einen Klick auf eine Karte wird diese umgedreht.
- Es darf immer nur eine Spielkarte aufgedeckt sein.
- Das Umdrehen soll animiert sein.
- **Freie kreative Übung.**



Übung B: Polaroid

- Suche im Internet nach einem Selbstportrait eines berühmten Malers.
- Erstelle eine Webseite mit dem Bild in einer Polaroid Optik.
- Füge einen Schatten hinzu.
- Drehe das Polaroid mit CSS.
- **Beschrifte es mit dem Namen des Künstlers.**

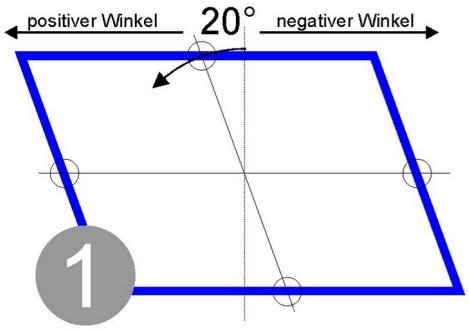
13.5 Transformieren II

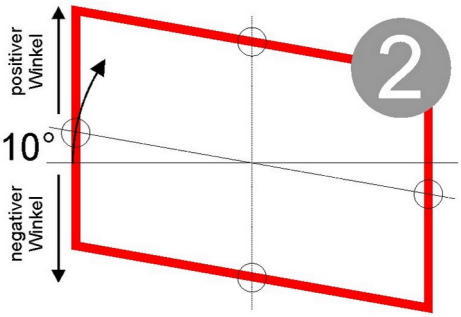
Verzerren

CSS

```
transform: skew(x,y);
```

Verzerrt ein Element über einen Winkel für die x oder y Achse.
Die x und y Achse lässt sich auch direkt ansprechen:
Form 1: `transform: skewX(20deg);`
Form 2: `transform: skewY(10deg);`





Verschieben

CSS

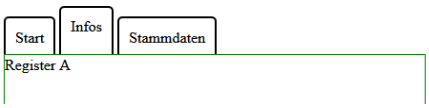
```
transform: translate(x,y);
```

Verschiebt ein Element über die X-Achse und Y-Achse.
Als Werte können Pixel, cm, %, em usw. gesetzt werden. Negative Werte sind ebenfalls möglich.
Die Eigenschaft kann auch direkt angesprochen werden:

`transform: translateX(2cm);` Zwei Zentimeter nach rechts

`transform: translateY(-35px);` 35 Pixel nach oben

Ein einfacher Tab-Reiter mit **`transform: translate`**




```
  .register {height:2em; display:inline-block;
             border-radius:5px 5px 0px 0px; padding:10px;
             border: 2px solid black;}
  .register:hover {transform: translateY(-0.7em);}
 #inhalt {background-color:#FFF; border:1px solid green; height:10em;
          transform:translateY(-1em);}
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      <nav><span class="register">Start</span>
           <span class="register">Infos</span>
           <span class="register">Stammdaten</span></nav>
      <div id="inhalt">Register A</div>
```

Komplexere Verformungen von Elementen lassen sich mit der **`matrix()`** *Funktion realisieren. z. B.* **`transform: matrix(1, 2, -1, 1, 80, 80);`**

13.6 Transitions

Auch ohne Animationen oder Keyframe Regeln, lassen sich Übergänge mit CSS definieren. Die **transiton** *Eigenschaft kommt immer dann zu tragen, wenn sich ein Element verändert (z. B. durch* **:hover**), *beim Start und in Verbindung mit einer Transformation. Um einen weichen Übergang zu erzeugen, muss man den Namen der CSS Eigenschaft und die Dauer angeben. Zusätzlich kann man noch eine Wartezeit und die Geschwindigkeit angeben.*

CSS

transition-property: [Wert];

Als **[Wert]** gibt man entweder die CSS Eigenschaft an, oder **all** um allen Eigenschaften einen weichen Übergang zu geben.

transition-property: background-color;
Der Übergang wird nur auf die Hintergrundfarbe des Selektors angewandt.

transition-property: all;
Alle Eigenschaften des Selektors bekommen einen Übergang.

CSS

transition-duration: [Wert];

Die Dauer des Überganges. Als **[Wert]** gibt man eine Zahl in Sekunden an.

transition-duration: 2s;	2 Sekunden

CSS

transition-timing-function: [Wert];

Die Geschwindigkeit des Überganges.
Als **[Wert]** sind unter anderen folgende möglich:

ease	langsamer Start, dann schneller und gegen Ende wieder langsam
linear	Gleiche Geschwindigkeit
ease-in	langsamer Start
ease-out	langsames Ende

Darüber hinaus gibt es noch **steps(int,start|end)**, **step-start**, **step-end**, **ease-in-out** und **cubic-bezir(n,n,n,n)**.

CSS

transition-delay: [Wert];

Verzögerung. Als **[Wert]** gibt man eine Zahl in Sekunden an.

transition-delay: 3s;	3 Sekunden Wartezeit bis zum Start.

```
.schalter {background:blue; color:white; padding:20px;
           transition-property:all; transition-duration:3s;
           transition-timing-function:ease;
           transition-delay:0.5s;}
.schalter:hover {background:yellow; color:black;}
```

CSS

transition: [property] [duration] [timing-function] [delay];

Shorthand – man kann alle vier Eigenschaften mit einer zusammenfassen.

```
.schalter {transition: all 3s ease 0.5s;}
```

Ü 13.6 Übungen Transitions

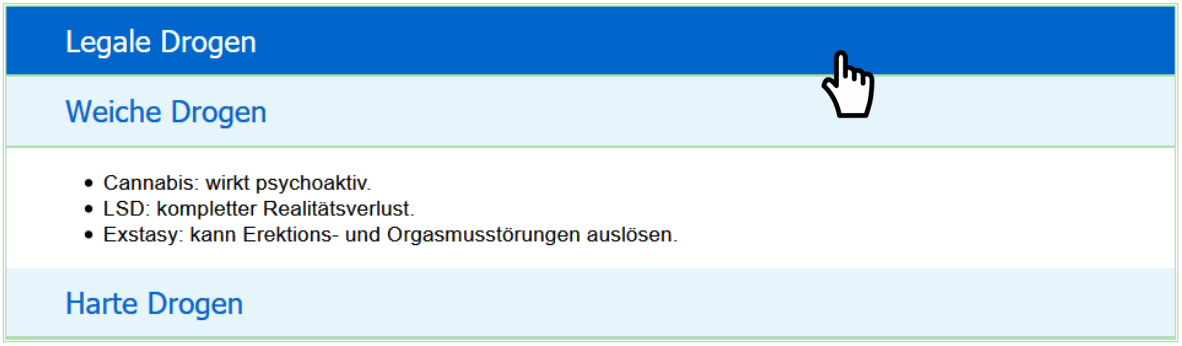
Übung A: Button mit Übergang

- Erstelle ein neues HTML Dokument.
- Gestalte zwei Buttons mit weichem Übergang.
- **Bei einem Hover soll ein 3D Effekt entstehen.**

3D Effekte lassen einfach sich durch einen veränderten Box-Shadow realisieren.

Übung B: Accordion

- Erstelle ein neues HTML Dokument mit einem Accordion
- Die Kategorien sind:
  4 Legale Drogen
  5 Sanfte Drogen
  6 Harte Drogen
- Suche im Internet nach Beispielen für die Kategorien
  (z. B: Harte Drogen, Heroin, Opium usw.) mit Wirkung und Gefahren.
- Durch einen Klick auf die Kategorie soll der Inhalt mit den Beispielen angezeigt werden – die anderen schließen sich.
- Das Öffnen und Schließen soll animiert sein.
- **Gestalte die Webseite anspruchsvoll mit CSS.**



Übung C: Mülltrennung

- Erstelle eine Webseite zum Thema "Mülltrennung"
- Gestalte es mit einem Tab-Reiter.
- **Freie kreative Übung.**

14.1 CSS Variablen

Mit benutzerdefinierten Eigenschaften (custom properties) oder CSS Variablen kann man zu Beginn Werte festlegen, die später bei einem Redesign nur einmal geändert werden müssen.

Browsersupport					
var()	49.0	15.0	31.0	9.1	36.0

Variable festlegen

CSS

:root {--variablenname: WERT;}

Die Variablen sollten zu Beginn der CSS Anweisungen definiert werden. Der Variablenname ist frei wählbar, man muss aber Groß- und Kleinschreibung beachten, weil diese Case-Sensitive sind.

```
:root {--akzent: #009;}
```

Variable abrufen

CSS

```
css-selector {css-eigenschaft: var(--variablenname);}
```

Die festgelegte Variable kann nun im gesamten CSS Bereich mit **var()** abgerufen werden.

```
h1 {color: var(--akzent);}
```

Um sicher zu gehen, kann man einen Fallback definieren der dann angewandt wird, wenn der Browser die Variablenfunktion nicht versteht. Dafür wird ein Beistrich gesetzt und der Fallback-Wert hinzugefügt.

```
h1 {color: var(--akzent, blue);}
```

Beispiel zur Nutzung von unterschiedlichen Variablentypen

```
:root {--schriftfarbe: #333;
       --schriftart: Arial, Times;
       --schriftgr: 2em; }

  h1 {color:var(--schriftfarbe, blue);
      font-family:var(--schriftart);
      font-size:var(--schriftgr, 3em);}

   p {color:var(--schriftfarbe, black);
      font-family:var(--schriftart);}
```

```
<h1>Ökologisch Einkaufen!</h1>
<p>Gut und günstig - die Umwelt freuts!</p>
```

14.2 calc() Funktion

Mit der **`calc()`** *Funktion können einfache Berechnungen in die CSS Anweisungen eingebunden werden. Diese sind jedoch nur bei Eigenschaften möglich, die ein nummerisches Maß erwarten (z. B:* **`width, height, font-size, margin, padding`** *usw.). Dabei können Berechnungen zwischen unterschiedlichen Maßen angestellt werden z. B.* **`width: calc(10em - 30px);`**

Browsersupport					
`calc()`	26.0	9.0	16.0	7.0	15.0

CSS

`CSS-Eigenschaft:` **`calc();`**

Mögliche Operationen sind

- **Addition (+)**
- **Subtraktion (-)**
- **Multiplikation (*)**
- **Division (/)**

sowie Klammern. Vor und nach dem Rechenzeichen muss ein Leerzeichen gesetzt werden (zumindest bei Plus und Minus).
Unmögliche Rechenergebnisse (z. B: Division durch Null, oder eine negative Zahl für **`width`**) werden ignoriert.

```
font-size: calc(12pt + 1em);
width:     calc(80% - 2 * 3px);
min-width: calc((60% + 8px) / 2 ));
```

Natürlich kann man die **`calc()`** *Funktion mit CSS Variablen* **`var()`** *kombinieren.*

```
:root {--breite:300px;
       --schrift:12pt;}
#eineBox {height:    200px; border: 3px solid red;
          width:     calc(var(--breite) + 100px);
          font-size: calc(var(--schrift) * 2);}
```

```
<div id="eineBox">Drinnen ist es dunkel</div>
```

Ein Fallback setzt man über die CSS Reihenfolge!

```
#eineBox {height:    200px; border: 3px solid red;
          width:     400px;
          width:     calc(var(--breite) + 100px);
          font-size: calc(var(--schrift) * 2);}
```

14.3 counter() Funktion

Neben dem **<ol>** *Element, kann man auch mit CSS die Anzahl der Verwendungen durchnummerieren lassen. z. B. jede Überschrift* **<h1>** *durchnummerieren.*

Dafür muss man zuerst den Zähler mit der CSS Eigenschaft **counter-reset** *auf Null setzen und dem Zähler einen Namen geben. Vorzugsweise im Selektor für den* **<body>**, *damit der Zähler für das gesamte Dokument gilt.*

CSS `counter-reset: NAME;`

```
body {counter-reset: abschnitt;}
```

Im Anschluss wird im Selektor für das gewünschte Element der Zähler aktiviert. Als Selektor verwenden wir ein **h2** *Element mit einem* **::before** *Pseudoelement. Damit wird bei jeder Überschrift der zweiten Ebene vor dem Text der Zähler hinzugefügt.*

CSS `counter-increment: NAME;`

```
h2::before {counter-increment: abschnitt;}
```

Mit der **content** *Eigenschaft und der* **counter()** *Funktion bestimmen wir das Aussehen.*

CSS `content: counter(NAME);`

```
body {counter-reset: abschnitt;}
h2::before {counter-increment: abschnitt;
            content: "Kapitel " counter(abschnitt) ": ";}
```

Zähler verschachteln (mehrere Ebenen durchnummerieren)

```
body {counter-reset: ersteEbene;}
h1 {counter-reset: zweiteEbene;}
h1::before {counter-increment: ersteEbene;
            content: "Kapitel " counter(ersteEbene) ": ";}
h2::before {
    counter-increment: zweiteEbene;
    content: counter(ersteEbene) "." counter(zweiteEbene) " ";}
```

```
<h1>Unternehmensführung</h1>
   <h2>Die Planung</h2>
   <h2>Die Kontrolle</h2>
<h1>Personalmanagement</h1>
   <h2>Arbeitnehmer haben Rechte</h2>
   <h2>Motivierte Mitarbeiter</h2>
<h1>Ökomanagement</h1>
   <h2>Grundlagen</h2>
   <h2>Instrumente</h2>
```

Kapitel 1: Unternehmensführung

1.1 Die Planung

1.2 Die Kontrolle

Kapitel 2: Personalmanagement

2.1 Arbeitnehmer haben Rechte

2.2 Motivierte Mitarbeiter

Kapitel 3: Ökomanagement

3.1 Grundlagen

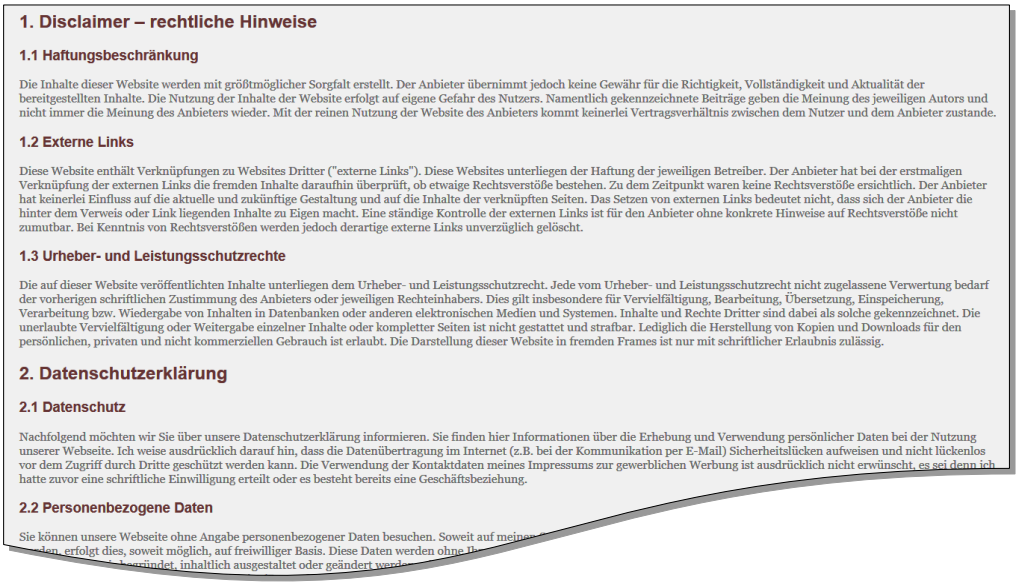
3.2 Instrumente

Übung A: Farbpalette

- Erstelle ein neues CSS Stylesheet
- Definiere mit CSS Variablen eine eigene Farbpalette im externen Stylesheet.
  `--meinBlau, --meinRot, --Akzent1` usw.
- Binde das CSS Stylesheet in ein beliebiges HTML Dokument ein.
  TIPP: `<link rel="stylesheet" ...`
- Teste deine Farbpalette.
- **Freie kreative Übung.**

Übung B: Impressum

- Öffne das HTML Dokument **`impressum.html`**
- Definiere die Schriftarten mit CSS Variablen.
  - Schriftfamilie für Überschriften
  - Schriftfamilie für Lauftext
- Definiere eine CSS Variable für die Schriftgröße mit 12pt.
- Die Schriftgrößen sollen mit **calc()** und der CSS Variable für die Schriftgröße berechnet werden.
  - h1 = doppelt so groß wie die CSS Variable
  - h2 = plus 0.5em zur CSS Variable
  - h3 = plus 2pt zur CSS Variable
  - p = die CSS Variable
- **Erstelle einen verschachtelten `counter()` für `<h2>` und `<h3>` (Nummerierung für die Überschriften)**



1. Disclaimer – rechtliche Hinweise

1.1 Haftungsbeschränkung

Die Inhalte dieser Website werden mit größtmöglicher Sorgfalt erstellt. Der Anbieter übernimmt jedoch keine Gewähr für die Richtigkeit, Vollständigkeit und Aktualität der bereitgestellten Inhalte. Die Nutzung der Inhalte der Website erfolgt auf eigene Gefahr des Nutzers. Namentlich gekennzeichnete Beiträge geben die Meinung des jeweiligen Autors und nicht immer die Meinung des Anbieters wieder. Mit der reinen Nutzung der Website des Anbieters kommt keinerlei Vertragsverhältnis zwischen dem Nutzer und dem Anbieter zustande.

1.2 Externe Links

Diese Website enthält Verknüpfungen zu Websites Dritter ("externe Links"). Diese Websites unterliegen der Haftung der jeweiligen Betreiber. Der Anbieter hat bei der erstmaligen Verknüpfung der externen Links die fremden Inhalte daraufhin überprüft, ob etwaige Rechtsverstöße bestehen. Zu dem Zeitpunkt waren keine Rechtsverstöße ersichtlich. Der Anbieter hat keinerlei Einfluss auf die aktuelle und zukünftige Gestaltung und auf die Inhalte der verknüpften Seiten. Das Setzen von externen Links bedeutet nicht, dass sich der Anbieter die hinter dem Verweis oder Link liegenden Inhalte zu Eigen macht. Eine ständige Kontrolle der externen Links ist für den Anbieter ohne konkrete Hinweise auf Rechtsverstöße nicht zumutbar. Bei Kenntnis von Rechtsverstößen werden jedoch derartige externe Links unverzüglich gelöscht.

1.3 Urheber- und Leistungsschutzrechte

Die auf dieser Website veröffentlichten Inhalte unterliegen dem Urheber- und Leistungsschutzrecht. Jede vom Urheber- und Leistungsschutzrecht nicht zugelassene Verwertung bedarf der vorherigen schriftlichen Zustimmung des Anbieters oder jeweiligen Rechteinhabers. Dies gilt insbesondere für Vervielfältigung, Bearbeitung, Übersetzung, Einspeicherung, Verarbeitung bzw. Wiedergabe von Inhalten in Datenbanken oder anderen elektronischen Medien und Systemen. Inhalte und Rechte Dritter sind dabei als solche gekennzeichnet. Die unerlaubte Vervielfältigung oder Weitergabe einzelner Inhalte oder kompletter Seiten ist nicht gestattet und strafbar. Lediglich die Herstellung von Kopien und Downloads für den persönlichen, privaten und nicht kommerziellen Gebrauch ist erlaubt. Die Darstellung dieser Website in fremden Frames ist nur mit schriftlicher Erlaubnis zulässig.

2. Datenschutzerklärung

2.1 Datenschutz

Nachfolgend möchten wir Sie über unsere Datenschutzerklärung informieren. Sie finden hier Informationen über die Erhebung und Verwendung persönlicher Daten bei der Nutzung unserer Webseite. Ich weise ausdrücklich darauf hin, dass die Datenübertragung im Internet (z.B. bei der Kommunikation per E-Mail) Sicherheitslücken aufweisen und nicht lückenlos vor dem Zugriff durch Dritte geschützt werden kann. Die Verwendung der Kontaktdaten meines Impressums zur gewerblichen Werbung ist ausdrücklich nicht erwünscht, es sei denn ich hatte zuvor eine schriftliche Einwilligung erteilt oder es besteht bereits eine Geschäftsbeziehung.

2.2 Personenbezogene Daten

Sie können unsere Webseite ohne Angabe personenbezogener Daten besuchen. Soweit auf meine[illegible] erfolgt dies, soweit möglich, auf freiwilliger Basis. Diese Daten werden ohne [illegible] gegründet, inhaltlich ausgestaltet oder geändert werden.

14.4 Attributselektoren

Mit dem einfachen Attributselektor können Elemente angesprochen, die ein bestimmtes Attribut besitzen. Zur Erinnerung: in `<p title="Absatz1">` *ist* `title` *das Attribut. Angesprochen wird das ganze Element indem der Attributselektor mit eckigen Klammern im CSS Bereich angegeben wird. z. B:* `[title] {color:red;}`

CSS

`[Attribut]: {CSS Eigenschaft;}`

Im Beispiel werden alle Pflichtfelder blau eingefärbt und alle Beschriftungen die ein Title Attribut haben eine rote Schriftfarbe zugewiesen.

```
[required] {background-color:blue;}
[title] {color:red;}
```

```
<label for="vorname" title="Vorname">Vorname: </label>
<input name="vorname" id="vorname" type="text" required ><br>
<label for="nachname" title="Nachname">Nachname: </label>
<input name="nachname" id="nachname" type="text" required >
```

CSS

`[Attribut=Attributwert]: {CSS Eigenschaft;}`

Gibt man dem Attribut zusätzlich (durch ein = Zeichen) einen Wert mit, so werden nur jene Elemente angesprochen, die sowohl Attribut als auch den passenden Wert haben. Im Beispiel wird nur der Reset-Button durchgestrichen.

```
[type=reset] {text-decoration: line-through;}
```

```
<button type="submit">Absenden</button>
<button type="reset">Zurücksetzen</button>
```

Es gibt drei Teilübereinstimmungen für die Attributwerte.

`[Attribut^="abc"]` ← Zeichenkette zu Beginn des Attributwertes
`[Attribut$="xyz"]` ← Zeichenkette am Schluss des Attributwertes
`[Attribut*="klm"]` ← Zeichenkette innerhalb des Attributwertes.

```
[href^="http://"] {font-weight: bold }
[href$=".com"]    {color: green }
[title*="Graz"]   {font-style:italic;}
```

```
<a href="http://www.css4.at">CSS4 Lernplattform</a>
<a href="http://www.wordpress.com">Blog Anbieter</a>
<h1 title="Für Graz und Graz-Umgebung">In ihrer Region</h1>
```

Attributselektoren können mit anderen Selektoren verbunden werden:

Universalselektor: `*[ Attribut]` → `*[title="Wichtig"] {…;}`
Typselektor: `element[Attribut]` → `a[href$=".net"] {…;}`
Klassenselektor: `.klasse[Attribut]` → `.bild[src*="png"] {…;}`

15.1 Image Map

HCL151
Austria.png

Mit einer Image Map kann der Anwender per Maus auf ein Detail in einem Bild klicken um damit einen Verweis (Link) auszuführen. z. B. Es wird eine Österreichkarte angezeigt und er User kann ein Bundesland auswählen. Zuerst wird ein beliebiges Bild mit dem Attribut **usemap** *eingefügt (mit* **#***). Dann ein* **<map>** *Element mit dem Attribut* **name***, welches gleich sein muss (ohne* **#***) wie das* **usemap** *Attribut im Bild. Innerhalb des* **<map>** *Tags wird über das* **<area>** *Element der anklickbare Bereich festgelegt.*

HTML

```
<img src="" … usemap="#mapname">
<map name="#mapname">
    <area shape="" coords="" href="" target="">
</map>
```

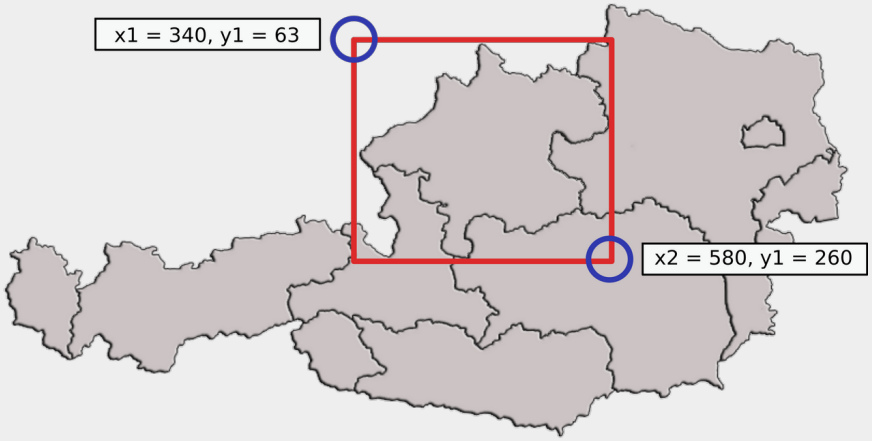
Es können beliebig viele <area> Tags innerhalb des <map> Elements gesetzt werden. Folgende Attribute sind möglich:

shape bestimmt die Form

- **shape="rect"** ← eine rechteckige Fläche
- **shape="circle"** ← ein Kreis
- **shape="poly"** ← ein Polygon

coords bestimmt die Koordinaten der Form

Für **shape="rect"** gilt **coords="x1, y1, x2, y2"**



Für **shape="circle"** gilt **coords="x, y, r"**

- **x** = Mittelpunkt, Anzahl der Pixel von links
- **y** = Mittelpunkt, Anzahl der Pixel von oben
- **r** = Radius in Pixel

Für **shape="poly"** gilt **coords="x1, y1, x2, y2 … xn, yn"**

href und *target* benötigt man für den Verweis.

```
<img src="Austria.png" width="800" heigth="450"
     usemap="#AustriaKarte">
  <map name="#AustriaKarte">
    <area shape="rect" coords="340, 63, 580, 260"
          href="http://www.ooe.gv.at"
          alt="Oberoesterreich" target="_blank" >
  </map>
```

Ü 15.1 Übungen Image Map

Übung A: Fehlersuchbild

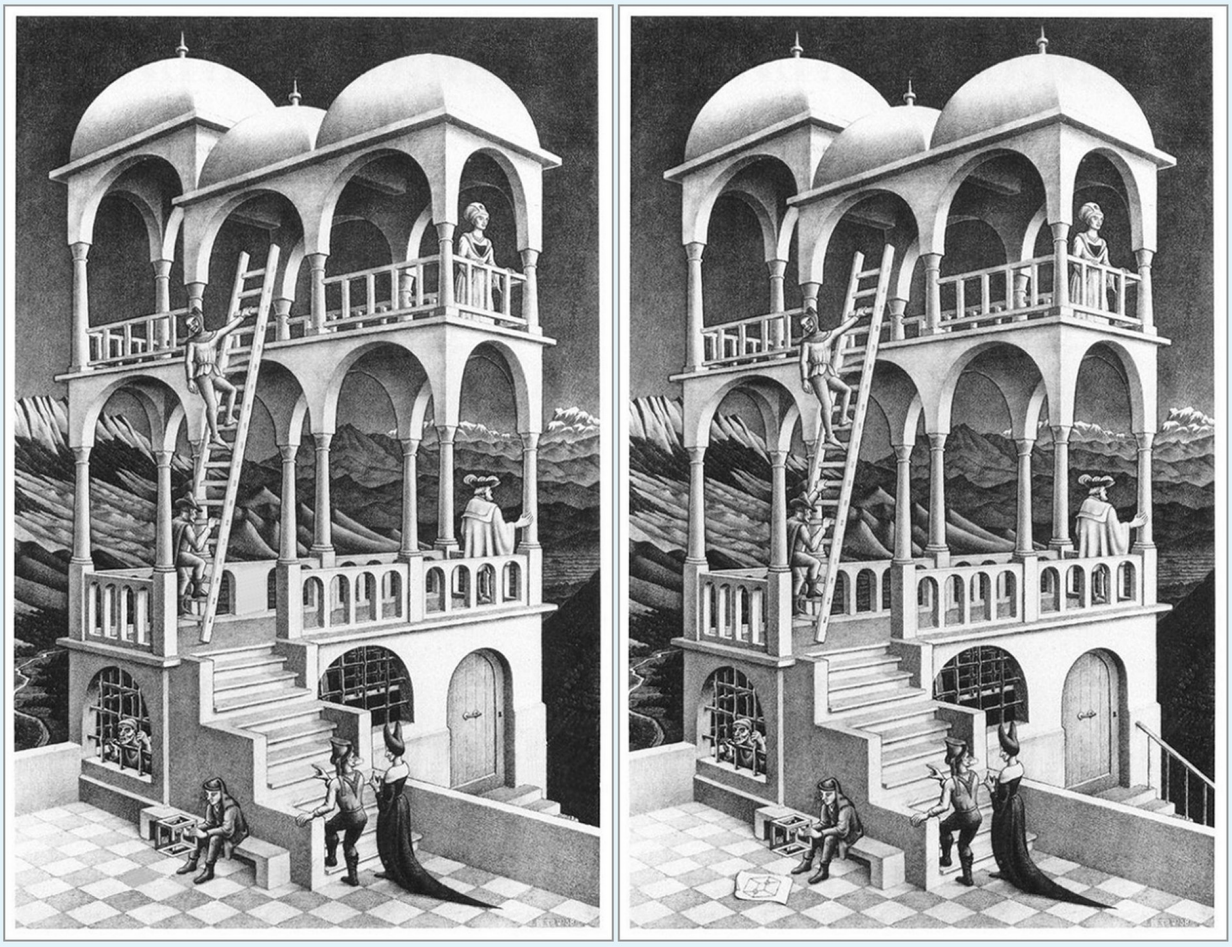
- Öffne mit einem Bildbearbeitungsprogramm ein beliebiges Bild
- Retuschiere in dem Bild fünf Fehler
- Erstelle ein neues HTML Dokument.
- Die fünf Fehler sollen über eine Image Map anklickbar sein.
- Baue ein Feedback ein
  (wenn man auf einen Fehler klickt, soll eine Rückmeldung kommen!)
- **Gestalte die Seite mit CSS anspruchsvoll. Freie kreative Übung.**

Im Internet findest du zahlreiche Image-Map Generatoren
z. B. auf `https://www.image-map.net`

Mit der CSS Eigenschaft `cursor` kann man den Mauszeiger verändern.
z. B. `#meinbild:hover {cursor:pointer;}`

Fehlersuchbild

Finde die 5 Fehler im linken Bild



Internetquellen und online Tools

Background Image Generator
http://bg.siteorigin.com/

css4.at Lernplattform
https://www.css4.at

dict.cc Wörterbücher
https://www.dict.cc/

Duden onine
https://www.duden.de/woerterbuch

ExtractMetaData – Meta Daten auslesen
https://www.extractmetadata.com/de.html

flickr Fotocommunitiy
https://www.flickr.com/

Fundamentum – pädagogisches Forum
http://fundamentum.xobor.de/

Google (nahezu alle Dienste)
https://www.google.at

MDN web docs – Mozilla für Entwickler
https://developer.mozilla.org/de/

Mediaevent – CSS, HTML und JS mit {stil}
https://www.mediaevent.de/

openstreetmap
https://www.openstreetmap.org

Peter Kropff Tutorials
https://www.peterkropff.de/

pixabay Bilderplattform
https://pixabay.com/de/

php.net – Benutzerhandbuch
https://www.php.net

PHP-Einfach.de Tutorials
https://www.php-einfach.de

selfhtml.org – Die Energie des Verstehens
https://selfhtml.org/

stackoverflow – developers empowering
https://stackoverflow.com/

W3C CSS Validation Service
http://jigsaw.w3.org/css-validator

W3C Markup Validation Service
https://validator.w3.org

w3schools.com – Web Developer Site
https://www.w3schools.com/

Wikimedia Commons
https://commons.wikimedia.org

Wikipedia – die freie Enzyklopädie
https://de.wikipedia.org