



ABB Technikerschule

Technik. Informatik. Wirtschaft. Management →

NDS - Web Engineering

Einführung & Grundlagen 1



Steckbrief Dozent

<i>Key</i>	<i>Value</i>
Name	Hannes Morgenthaler
Alter	Über 40x um die 🍑 gekreist...
Herkunft	Aus dem Westen der CH (Fribourg), Erde 🌎 (<i>angeblich</i>)
Abschluss	BSc. Informatik FH
Job	Lead Software Developer, INGTES AG
Erfahrung	Webseiten & Frontend: ~20 Jahre Fullstack Developer: 12+ Jahre Dozent: ~1 Jahr

Frage: wie schätzt DU dein Vorwissen zu diesem Modul ein?

Skala	Beschreibung	Anzahl Stimmen
1 😳	Webbrowser, was ist das...?	
2 😳 😳	Web-Power-User mit Adblockern und weiteren Extensions	
3 😳 😳 😳	Hab schon mal was von HTML/CSS/JS gehört...	
4 😳 😳 😳 😳	Habe schon selber HTML/CSS/JS geschrieben	
5 😳 😳 😳 😳 😳	Hab schon selber Webseiten und/oder Web-Apps erstellt/gestaltet	

Welche Erwartungen hast **DU** an dieses Modul?

- Diese **Kompetenz / Fähigkeit** möchte ich gerne erlernen
- Dieses **Thema** oder diese **Technologie** interessiert mich

Auftrag

1. Überlege dir **2 bis max. 3 Stichworte / Begriffe**
2. Tritt an die **Wandtafel** und **schreibe** diese auf
3. **Existiert** ein Stichwort schon, mach einen **Strich dahinter**

Zielsetzungen des Dozenten

Ich als Student **kenne die Grundlagen** von

- `HTML` / `CSS` / `JS`
- (sicherer) **Kommunikation** zwischen **Client** und **Server**
- von Responsive Webdesign (*SPA passt sich an Displaygrösse an*)
- von Progressive Web Apps (PWAs) (*SPA kann offline Inhalte speichern und lässt sich installieren*)

Ich als Student **kann selbstständig**

- einen einfachen **Webhost** mit `Ktor` in `Kotlin` programmieren
- eine einfache **Single-Page-App (SPA)** mit `Vue.js` programmieren
- ein **einfaches Design** gemäss einer **Vorlage** in `HTML` / `CSS` umsetzen

Geplantes Kursprogramm

Reihenfolge und Inhalt von Themen wird möglichst den Bedürfnissen angepasst und kann daher ggf. leicht ändern.

1. Grundlagen
 1. Das **World Wide Web (WWW)**
 2. **HTML** & **CSS**
 3. Funktionalität eines **Webbrowsers**
 4. **Document Object Model (DOM)**
 5. **JavaScript** & **TypeScript**
 6. erste **Dynamische Webseite** von 
 7. **HTTP(S), Sessions & Cookies**
 8. Funktionalität eines **Webservers**
2. **Frontend:** Single Page Applications mit **Vue.js**
3. **Backend:** Webserver mit **Ktor** und **Kotlin**
4. **Responsive Web Design**
(Dynamische Anpassung an Bildschirmgrößen)
5. Weiterarbeit am Projekt **Smart Home**
 1. Programmieren eines **Firefighter-Dashboards**
(Front- und Backend)
 2. Anbindung ans Smart-Quartier
 3. Alarme, Events, Aktionen
6. **Progressive Web Apps**
(Offline, Service-Worker, Installation im OS)

Allgemeine Anmerkungen zum Kursinhalt

- Das Thema "Web Engineering" ist sehr umfangreich und vielfältig
- Der zeitlich begrenzte Umfang dieses Moduls wird leider bei Weitem nicht reichen, alle relevanten Aspekte ausreichend zu beleuchten - es musste (notgedrungen) an diversen Stellen erhebliche Kürzungen vorgenommen werden... 😳
- Dieser Kurs fokussiert sich auf die allerwesentlichsten Aspekte, die nach Ansicht des Dozenten zur praktischen Anwendung notwendig sind.
- In diesem Kurs kommen konkrete Techniken, Tools und Frameworks als eine (!) Möglichkeit des Web Engineering zum Einsatz, die vom Dozenten und/oder der Schule bevorzugt werden und sich in der Praxis bewährt haben - aus Zeitgründen können kaum Alternativen betrachtet werden. Es sei jedoch angemerkt, dass es für fast jedes Tool meist dutzende gleichwertige Alternativen gibt → bei Interesse kann der Dozent gerne Auskunft geben/beraten.

Allgemeine Regeln und Grundsätze 1

Ablauf des Unterrichts

- **Qualität vor Quantität:** lieber ein Thema verschieben und dafür den restlichen Stoff richtig verstehen - bitte gebt dem Dozenten Rückmeldung, wenn er es selbst nicht merkt... 🤔
- **Kurstage nach ABB-TS Standard:** jeweils 4 Lektionen à 45 Minuten (jeweils 5' Pause dazwischen)
- In **gemeinsamer** Vereinbarung können situativ Anpassungen vorgenommen werden.

Eigenverantwortung der Studenten

- **Pünktlichkeit:** unsere Zeit ist (leider) knapp bemessen - rechtzeitig starten = rechtzeitig beenden (gilt auch für Pausen)!
- **Keine Präsenzpflicht:** wer fehlt holt Stoff **selbstständig** nach
- **Wenn etwas nicht verstanden wurde:** fragen! (notfalls auch mehrfach, ich erkläre gerne nochmals 🔮)
- **Hausaufgaben:** dienen der **Vertiefung** der Erkenntnisse und sind **unerlässlich**. Das **selbstständige Lösen der Hausaufgaben** wird beim nächsten Kurstag **vorausgesetzt**.

Allgemeine Regeln und Grundsätze 2

Umgang Miteinander

Gegenseitiger Respekt und Toleranz

- **Wir sind alle Erwachsenen:** Pöbeleien, Provokationen, Mobbing, usw. werden **nicht geduldet** und ggf. geandert.
- **Es gibt keine dummen Fragen:** jede Person bringt ihr eigenes Vorwissen mit, lernt anders und im eigenen Tempo - wir **nehmen aufeinander Rücksicht** und geben einander **wertschätzend Feedback**.

Unklarheiten, Fragen und Kritik

- **Verständnisprobleme:**
 - wenn möglich direkt im Unterricht ansprechen
 - Schriftliches Kontaktieren des Dozenten via **MS Teams** oder **E-Mail (Antwort i.d.R. innerhalb von 24h)**
- **Individuelle Lösungen:** falls die eine oder andere Gruppe **stark benachteiligt** oder **stark über- oder unterfordert** ist, werden individuelle Lösungen gesucht wie z.B. **Zusatzunterricht** oder **Gruppenaufträge zum selbständigen Erarbeiten und Präsentieren eines Themas.**

Allgemeine Regeln und Grundsätze 3

Präsenzunterricht ist zum Lernen da

- **Keine Lust oder Zeit:** Wer mal an der Lektion **nicht mitmachen will** oder **nicht kann** z.B. wegen geschäftlicher Verpflichtungen, Stress oder Erschöpfung - kein Problem ( **Eigenverantwortung!**) - bitte einfach den **Unterrichtssaal verlassen** um den Lernfluss der Mitstudenten nicht zu stören!
- **Fragen:** dürfen und sollen jederzeit gestellt werden können - einfach **alles mit Mass:** wenn der Fluss oder Zeitplan des Unterrichts erheblich gestört wird, werden Fragen ggf. auf später vertagt.



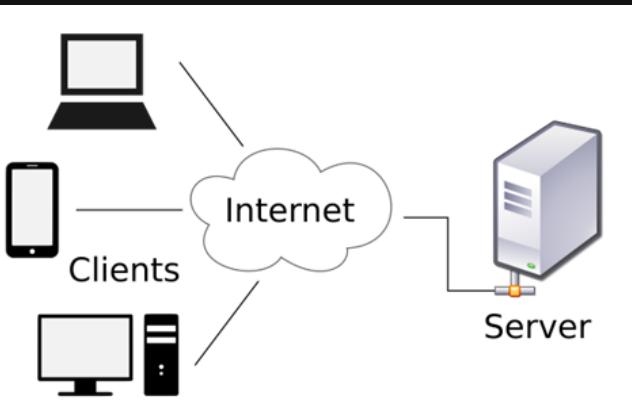
Das World Wide Web

- ENG für "Weltweites Netz", kurz **Web** oder auch **WWW** genannt.
- Ein über das **Internet** abrufbares System von elektronischen **Hypertext-Dokumenten**, sogenannten **Webseiten**
- Umgangssprachlich wird das **WWW** oft mit dem **Internet** gleichgesetzt; das **WWW** ist jedoch **jünger** und stellt **nur eine** von mehreren möglichen **Nutzungen des Internets** dar. Andere Internetdienste wie z.B. **E-Mail** sind **nicht** in das **WWW** integriert.
- Das **WWW** wurde 1989 von **Tim Berners-Lee** und **Robert Cailliau** am **CERN** in **Genf (CH)** entwickelt.

Das World Wide Web 2

- Tim Berners-Lee entwickelte dazu das **HTTP-Netzwerkprotokoll** und **HTML**. Zudem programmierte er den ersten **Webbrowser** und die erste **Webserver-Software**. Er betrieb auch den **ersten Webserver der Welt** auf seinem Entwicklungsrechner 😎💪!
- Das Gesamtkonzept wurde der Öffentlichkeit 1991 unter **Verzicht** auf jegliche **Patentierung** oder **Lizenzzahlungen** zur **freien Verfügung** gestellt, was **erheblich zur heutigen Bedeutung** beitrug.
- Die weltweit erste Webseite info.cern.ch wurde am 6. August 1991 veröffentlicht.
- Das WWW führte zu umfassenden, oft als revolutionär beschriebenen **Umwälzungen in vielen Lebensbereichen**, zur **Entstehung neuer Wirtschaftszweige** und zu einem grundlegenden **Wandel des Kommunikationsverhaltens** und der Mediennutzung. Es wird in seiner **kulturellen Bedeutung**, zusammen mit anderen Internet-Diensten wie E-Mail, teilweise mit der **Erfindung des Buchdrucks** gleichgesetzt

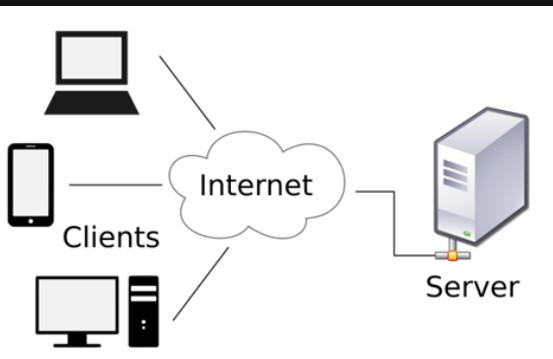
WWW - Funktionsweise



- *Drei Kernstandards*
 - **HTTP** als **Protokoll**, mit dem der Browser Informationen vom Webserver anfordern kann
 - **HTML** als **Auszeichnungssprache**, die festlegt, wie die Information gegliedert ist und wie die Dokumente verknüpft sind (**Hyperlinks**)
 - **Uniform Resource Identifier (URI)** als eindeutige **Bezeichnung** einer Ressource, die in **Hyperlinks** verwendet wird

WWW - Funktionsweise 2

- *Erweiterungen*
 - **Cascading Style Sheets** (CSS) beschreiben das **Aussehen** und die **Andordnung** der Elemente einer Webseite, womit der Inhalt von dessen Darstellung separiert wird
 - **Document Object Model** (DOM) als **Programmierschnittstelle** für externe Programme oder Skriptsprachen (wie JavaScript) von Webbrowsern



Minimale Entwicklungsumgebung einrichten

1. Installieren von *Visual Studio Code (VsCode)*:

<https://code.visualstudio.com>

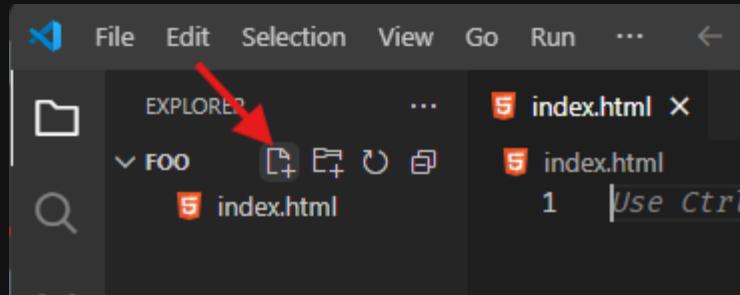
2. Installieren von Erweiterungen:

1. Auto Complete Tag & Live Server

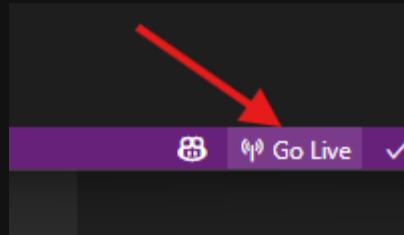


3. Erzeugen sie eine neue Datei mit Namen

index.html :



4. Starten sie den Live-Server:



Allgemeine Anmerkungen

- Viele der nachfolgenden Beispiele basieren auf den exzellenten Tutorials von W3Schools 
- Es lohnt sich IMHO sehr, auf <https://www.w3schools.com/> etwas zu stöbern, es gibt viele gratis Tutorials mit Übungen für viele unterschiedliche Techniken und Tools, die sich in der Praxis bewährt haben!

Was ist HTML?

- HTML steht für Hyper Text Markup Language
 - Hypertext: Wortbildung aus altgriechisch
 - ὑπέρ hyper – zu deutsch ‚über, oberhalb, über ... hinaus‘
 - lateinisch texere – zu deutsch ‚weben, flechten‘
 - Markup Language (*deutsch: Auszeichnungssprache*): maschinenlesbare Sprache für die Gliederung und Formatierung von Texten und anderen Daten
- HTML ist die Standard BeschreibungsSprache für Webseiten
- HTML beschreibt die Struktur einer Webseite
- HTML besteht aus einer Serie von Elementen
- HTML-Elemente sagen dem Webbrowser, welche Inhalte dargestellt werden sollen

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>My first heading</h1>
    <p>My first paragraph</p>
  </body>
</html>
```



Simples HTML-Dokument

- Die `<!DOCTYPE html>`-Deklaration definiert, dass dieses Dokument ein HTML5-Dokument ist.
- Das `<html>`-Element ist das Wurzelement einer HTML-Seite.
- Das `<head>`-Element enthält Metainformationen über die HTML-Seite.
- Das `<title>`-Element gibt einen Titel für die HTML-Seite an (der im Titelbalken des Browsers oder im Tab der Seite angezeigt wird).
- Das `<body>`-Element definiert den Inhalt des Dokuments und ist ein Container für alle sichtbaren Inhalte wie Überschriften, Absätze, Bilder, Hyperlinks, Tabellen, Listen usw.
- Das `<h1>`-Element definiert eine große Überschrift.
- Das `<p>`-Element definiert einen Absatz

Was ist ein HTML-Element?

- Ein **HTML-Element** wird durch ein **Start-Tag**, etwas **Inhalt** und ein **End-Tag** definiert:

```
<tagname>Inhalt ... </tagname>
```

- Das HTML- Element umfasst alles vom Start-Tag bis zum End-Tag:

```
<h1>Meine Überschrift</h1>
<p>Mein Absatz</p>
```

- **Hinweis:** Einige HTML-Elemente haben keinen Inhalt (wie das `
`-Element). Diese Elemente werden leere Elemente genannt. *Leere Elemente* haben **kein End-Tag**.

HTML-Seitenstruktur

- Alle HTML-Dokumente müssen mit einer Dokumenttypdeklaration beginnen: `<!DOCTYPE html>` Sie stellt den Dokumenttyp dar und hilft Webbrowsern, Webseiten korrekt anzuzeigen.
- Das HTML-Dokument selbst beginnt mit `<html>` und endet mit `</html>`.
- Der sichtbare Teil des HTML-Dokuments liegt zwischen `<body>` und `</body>`.

```
<!DOCTYPE html>
<html>
  <head>
    | <title>Ich bin der im Tab/Fenster angezeigte Titel</title>
  </head>

  <body>
    | <h1>Ich bin eine GROSSE Überschrift</h1>
    | <p>Ich bin ein Paragraph</p>
    | <p>Ich bin ein Paragraph mit <strong>starkem</strong> Inhalt 💪 !</p>
  </body>
</html>
```

HTML-Überschriften

- HTML-Überschriften werden mit den `h`-Tags definiert: `<h1>` bis `<h6>`.
- `h1` definiert die wichtigste Überschrift.
- `h6` definiert die unwichtigste Überschrift.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>This is heading 1</h1>
    <h2>This is heading 2</h2>
    <h3>This is heading 3</h3>
    <h4>This is heading 4</h4>
    <h5>This is heading 5</h5>
    <h6>This is heading 6</h6>
  </body>
</html>
```

HTML-Absätze

HTML-Absätze werden mit dem Tag `<p>` definiert.

```
<!DOCTYPE html>
<html>
| <head>
| | <title>Page Title</title>
| </head>
|
| <body>
| | <p>Absatz 1</p>
| | <p>Absatz 2</p>
| | <p>Absatz 3</p>
| </body>
</html>
```

HTML-Links

- HTML-Links werden mit dem `<a>`-Tag definiert (`a` steht für `anchor` ("ANCHOR" zu deutsch)).
 - Das Ziel des Links wird im `href` Attribut angegeben
- ➔ zu **Attributen** gibt es gleich mehr Infos...

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <a href="https://www.abmts.ch/">
      Link zur ABB-TS-Homepage
    </a>
  </body>
</html>
```

HTML-Attribute

- Alle HTML-Elemente können sog. **Attribute** haben.
- Attribute liefern **zusätzliche Informationen** zu den Elementen.
- Attribute werden immer im **Start-Tag** angegeben.
- Attribute kommen normalerweise in Name-Wert-Paaren wie `name="wert"` vor.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <a href="https://www.abmts.ch/">
      Link zur ABB-TS-Homepage
    </a>
    
  </body>
</html>
```

HTML-Bilder 1

- HTML-Bilder werden mit dem ``-Tag definiert.
- Die Quelldatei (`src`), Alternativtext (`alt`), `width` und `height` werden als spezifische Attribute für dieses Element bereitgestellt.
- Beachten sie die effektive Grösse des Bildes (*32px * 32px*) und die effektive Darstellung infolge der Attribute `width` und `height`.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    
  </body>
</html>
```

HTML-Bilder 2

Es gibt zwei Möglichkeiten, die URL anzugeben im `src` :

- **Absolute URL:** Links zu einem externen Bild, das auf einer anderen Webseite gehostet wird; z.B.

`src="https://www.w3schools.com/html/pic_trulli.jpg"` :



- **Relative URL:** Link zu einem Bild, das auf der eigenen Webseite gehostet wird. Hier enthält die URL nicht den Domainnamen.
 - Wenn die URL ohne Schrägstrich (`/`) beginnt, ist sie *relativ* zur *aktuellen Seite*. Bsp. `src="img.jpg"`
 - Wenn die URL mit einem Schrägstrich (`/`) beginnt, ist sie relativ zur aktuellen *Domain*. Bsp. `src="images/img.jpg"` ist relativ zum aktuellen Host- bzw. Domainnamen (z.B. `https://www.abbts.ch`) zu verstehen.

Tipp: Es ist (*fast*) immer am besten, relative URLs zu verwenden.

Frage in die Runde: warum...?

HTML-Anzeige

- Mensch kann nicht absolut sicher sein, wie HTML angezeigt wird.
- Grosse oder kleine Bildschirme und veränderte Fenstergrößen führen zu unterschiedlichen Ergebnissen.
- Bei HTML können Sie die Anzeige **nicht** ändern, indem Sie Ihrem HTML-Code **zusätzliche Leerzeichen** oder *zusätzliche Zeilen hinzufügen.
- Der Webbrowser entfernt **automatisch** alle zusätzlichen Leerzeichen und Zeilen, wenn die Seite angezeigt wird.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <p>
      Dieser Paragraph enthält viele  
Leerschläge im Quellcode, aber  
Der Webbrowser ignoriert diese.
    </p>
    <p>
      Die Anzahl Linien in einem Paragraph hängt von der Grö
    </p>
  </body>
</html>
```

Vorformatierte Texte in HTML

- Das HTML `<pre>`-Element definiert vorformatierten Text.
- Der Text innerhalb eines `pre`-Elements wird in einer Schriftart mit fester Breite (*normalerweise Courier*) angezeigt und behält sowohl Leerzeichen als auch Zeilenumbrüche bei.

```
<!DOCTYPE html>
<html>
| <head>
| | <title>Page Title</title>
| </head>
| <body>
| | <pre>
| | | Dieser Paragraph enthält viele
| | | Leerschläge im Quellcode, und der
| | | Webbrowser berücksicht
| | | diese.
| | </pre>
| </body>
| </html>
```

HTML-Styles

- Das HTML `style` -Attribut wird verwendet, um Stile zu einem Element hinzuzufügen, z.B. Farbe, Schriftart, Grösse und noch viel mehr...!
- Das HTML `style` -Attribut hat die folgende Syntax:

```
<tagname style="eigenschaft: wert;"> ... </tagname>
```

```
<!DOCTYPE html>
<html>
  <body>
    <p>Ich bin normal</p>
    <p style="color: red;">Ich bin rot</p>
    <p style="color: blue;">Ich bin blau</p>
    <p style="font-size: 30px;">Ich bin GROSS! 🤘</p>
  </body>
</html>
```

- Die Eigenschaft ist eine **CSS-Eigenschaft**. Der Wert ist ein **CSS-Wert**.

➔ Zu CSS erfahren wir gleich mehr...

Simple HTML-Seite bauen

Auftrag

- Versuchen sie mit dem *soeben vermittelten Wissen* folgende *HTML-Seite* nachzubauen
- Sie finden die Slides unter <https://teaching-abbts.github.io/nds-web-engineering/day-1/slides>
- Das Bild finden sie unter <https://teaching-abbts.github.io/slides-images/abbts-nds.jpg>. Laden sie dieses herunter und verwenden sie es mit einer *relativen URL*.
- **Achtung:** die farbigen Texte sind *Links*!
 - <https://www.abbts.ch/bildungsgaenge/>
 - <https://www.abbts.ch/nachdiplomstudien/>
 - <https://www.abbts.ch/kurse/>

Höhere Fachschule für Technik, Informatik, Wirtschaft und Management



Die ABB Technikerschule ist eine öffentliche, markt- und leistungsorientierte Bildungsinstitution in der Höheren Berufsbildung und bietet technisch ausgebildeten, ambitionierten Berufsfachleuten berufsbegleitende Bildungsgänge, Nachdiplomstudien und zukunftsgerichtete Weiterbildungsformate an.

«Aus der Praxis - für die Praxis» ist unsere Stärke.

Simple HTML-Seite bauen - Lösungsvorschlag

```
1  <!doctype html>
2  <html>
3  |  <head>
4  |  |  <title></title>
5  |  </head>
6  |  <body>
7  |  |  <h1 style="color: blue">Höhere Fachschule für Technik, Informatik, Wirtschaft und Management</h1>
8  |  |  
9  |  |  <p>
10 |  |  |  Die ABB Technikerschule ist eine öffentliche, markt- und leistungsorientierte Bildungsinstitution in der
11 |  |  |  Höheren
12 |  |  |  Berufsbildung und bietet technisch ausgebildeten, ambitionierten Berufsfachleuten berufsbegleitende
13 |  |  |  <a title="Bildungsgänge HF" href="https://www.abbts.ch/bildungsgaenge/" style="color: red">Bildungsgänge</a>,
14 |  |  |  <a title="Nachdiplomstudien HF" href="https://www.abbts.ch/nachdiplomstudium/" style="color: green">
15 |  |  |  |  Nachdiplomstudien
16 |  |  |  </a>
17 |  |  |  und zukunftsgerichtete
18 |  |  |  <a title="Lehrgänge HFP" href="https://www.abbts.ch/kurse/" style="color: violet">Weiterbildungsformate</a>
19 |  |  |  an.
20 |  |  |  </p>
21 |  |  <strong>«Aus der Praxis – für die Praxis» ist unsere Stärke.</strong>
22 |  </body>
23 </html>
```

Was ist CSS?

- Cascading Style Sheets (CSS) werden zum Formatieren des Layouts einer Webseite verwendet.
- Mit CSS können viele verschiedene Dinge definiert und kontrolliert werden:
 - Farben
 - Schriften
 - Textgrößen
 - Abstände zwischen Elementen
 - Positionierung und Anordnung von Elementen
 - Hintergrundbilder oder Hintergrundfarben
 - unterschiedliche Anzeigen für unterschiedliche Geräte und Bildschirmgrößen
 - Und noch viel mehr...!
- Das Wort Cascading (Kaskadierung) bedeutet, dass ein auf ein übergeordnetes Element angewandter Stil auch für alle untergeordneten Elemente *innerhalb* des übergeordneten Elements gilt. Wenn also die Farbe des Fließtexts auf „Blau“ eingestellt wird, erhalten auch alle Überschriften, Absätze und andere Textelemente innerhalb des Fließtexts die gleiche Farbe (sofern nichts anderes angegeben ist).

Verwendung von CSS

CSS kann auf drei Arten zu HTML-Dokumenten hinzugefügt werden:

- **Inline** – durch Verwendung des `style`-Attributs innerhalb von HTML-Elementen
- **Intern** – durch Verwendung eines `<style>` Elements im `<head>` Abschnitt
- **Extern** – durch Verwendung eines `<link>` Elements zur Verknüpfung mit einer externen CSS-Datei

Inline-CSS (*Repetition*)

- Wird verwendet um einem einzelnen HTML-Element einen eindeutigen Stil zuzuweisen.
- Verwendet das `style`-Attribut eines HTML-Elements

```
<!DOCTYPE html>
<html>
  <body>
    <p>Ich bin normal</p>
    <p style="color: red;">Ich bin rot</p>
    <p style="color: blue;">Ich bin blau</p>
    <p style="font-size: 30px;">Ich bin GROSS! 🤘</p>
  </body>
</html>
```

Internes CSS

- Ein *internes CSS* wird verwendet, um einen **Stil für eine einzelne HTML-Seite** zu definieren.
- Ein *internes CSS* wird im `<head>` Abschnitt einer HTML-Seite innerhalb eines `<style>` Elements definiert.
- Das folgende Beispiel setzt die Textfarbe ALLER `<h1>` Elemente (auf dieser Seite) auf Blau und die Textfarbe ALLER `<p>` Elemente auf Rot. Zusätzlich wird die Seite mit einer `powderblue` Hintergrundfarbe angezeigt.:

```
1  <!DOCTYPE html>
2  <html>
3  |  <head>
4  |  |  <style>
5  |  |  |  body { background-color: powderblue; }
6  |  |  |  h1   { color: blue; }
7  |  |  |  p    { color: red; }
8  |  |  </style>
9  |  </head>
10 |  <body>
11 |  |  <h1>Ich bin blau</h1>
12 |  |  <p>Ich bin rot</p>
13 |  |  <p>Ich bin auch rot...</p>
14 |  |  <h1>Ich bin wiederum blau...</h1>
15 |  </body>
16 </html>
```

Ich bin blau

Ich bin rot

Ich bin auch rot...

Ich bin wiederum blau...

Externes CSS

- Für viele HTML-Seiten wird ein **externes Stylesheet** verwendet, um den Stil zu definieren.
- Um ein externes Stylesheet zu verwenden, fügen Sie im `<head>` Abschnitt jeder HTML-Seite einen **Link** hinzu.
- Mit einem **externen Stylesheet** kann das **Aussehen der ganzen Webseite** durch *eine (!) Datei* verändert werden.

CSS-Farben, Schriftarten und -größen

Einige häufig verwendete CSS-Eigenschaften:

- Die CSS- `color` Eigenschaft definiert die zu verwendende Textfarbe.
- Die CSS- `font-family` Eigenschaft definiert die zu verwendende Schriftart.
- Die CSS- `font-size` Eigenschaft definiert die zu verwendende Textgröße.

CSS-Rahmen

- Die CSS- `border` Eigenschaft definiert einen **Rahmen** um ein **HTML-Element**.
- Für (fast) alle HTML-Elemente können Rahmen definiert werden.

```
<!doctype html>
<html>
  <head>
    <style>
      p {
        border: 2px solid powderblue;
      }
    </style>
  </head>
  <body>
    <h1>Ich bin eine Überschrift</h1>
    <p>Ich bin ein Paragraph</p>
    <p>Ich bin ein Paragraph</p>
    <p>Ich bin ein Paragraph</p>
  </body>
</html>
```

Ich bin eine Überschrift

Ich bin ein Paragraph

Ich bin ein Paragraph

Ich bin ein Paragraph

CSS: Padding und Margin

- Die CSS- `padding` Eigenschaft definiert einen **Abstand zwischen dem Text und dem Rahmen**.
- Die CSS- `margin` Eigenschaft definiert einen **Rand** (Leerraum) *ausserhalb des Rahmens*.
- Mit `.klassenname` und `class="klassenname"` können CSS Regeln kombiniert und sehr flexibel vergeben werden.

```
<!doctype html>
<html>
  <head>
    <style>
      .abstand {
        border: 2px solid powderblue;
      }
      .grosse-abstaende {
        padding: 30px;
        margin: 50px;
      }
    </style>
  </head>
  <body>
    <p class="abstand">Ich bin ein Paragraph</p>
    <p class="abstand grosse-abstaende">Ich bin ein Paragraph</p>
  </body>
</html>
```

Ich bin ein Paragraph

Ich bin ein Paragraph

HTML-DIV

- Das `<div>` Tag definiert eine **Unterteilung** oder einen **Abschnitt** in einem HTML-Dokument.
- Das `<div>` Tag dient als **Container für HTML-Elemente**, die dann gestaltet oder manipuliert werden. Es findet sehr häufig Verwendung zur Kapselung einer Gruppe von Elementen!
- Jede Art von Inhalt kann in das `<div>` Tag eingefügt werden!
- **Hinweis:** Standardmäßig setzen Browser immer einen Zeilenumbruch vor und nach dem `<div>` Element.

Gestalten eines Widgets - Auftrag

- Verwenden sie das nachfolgende **HTML-Dokument** als **Arbeitsgrundlage** *ohne dieses zu verändern.*
- Erzeugen sie die entsprechenden **CSS**-Regeln in der Datei `day-1-widget.css` um das gewünschte Aussehen zu erreichen.
- **Tipps:**
 - Welche **Stile** von Rändern gibt es und kann man die Ranbreite beeinflussen...?
 - Kann Text **aligniert** (rechts/links/mitte) werden...?

```
<!doctype html>
<html>
  <head>
    <link rel="stylesheet" href="day-1-widget.css" />
  </head>
  <body>
    <div class="widget">
      <h1>🔥</h1>
      <h2>Feuer!</h2>
    </div>
  </body>
</html>
```



Hausaufgabe

- Ihr Auftrag ist es, die gelernten Kursinhalte zu **wiederholen**, zu **ergänzen** und zu **vertiefen**, indem sie das HTML Tutorial der W3Schools durchspielen.
- Spielen sie dabei alle Kapitel bis und mit **HTML Id** durch
- Falls sie mit den heutigen Übungen nicht fertig geworden sind, versuchen sie es Zuhause in Ruhe nochmals...



Ende der heutigen Veranstaltung

Vielen herzlichen Dank für eure **Aufmerksamkeit** und **Mitarbeit** ❤️!

Kommt alle gut nach Hause, viel Erfolg bei den Hausaufgaben und eine gute, lehrreiche Woche 🌟

👋 bis nächsten Freitag!