Webseiten-Entwicklung
mit HTML, CSS und JavaScript

Autoren: Luca Rief, Daniel Appenmaier
Version: 1.2

- Einführung
- HTML: Aufbau der Webseite
- CSS: Aussehen der Webseite
- JavaScript: Verhalten der Webseite

Einführung

Vorstellung

Helen Müller

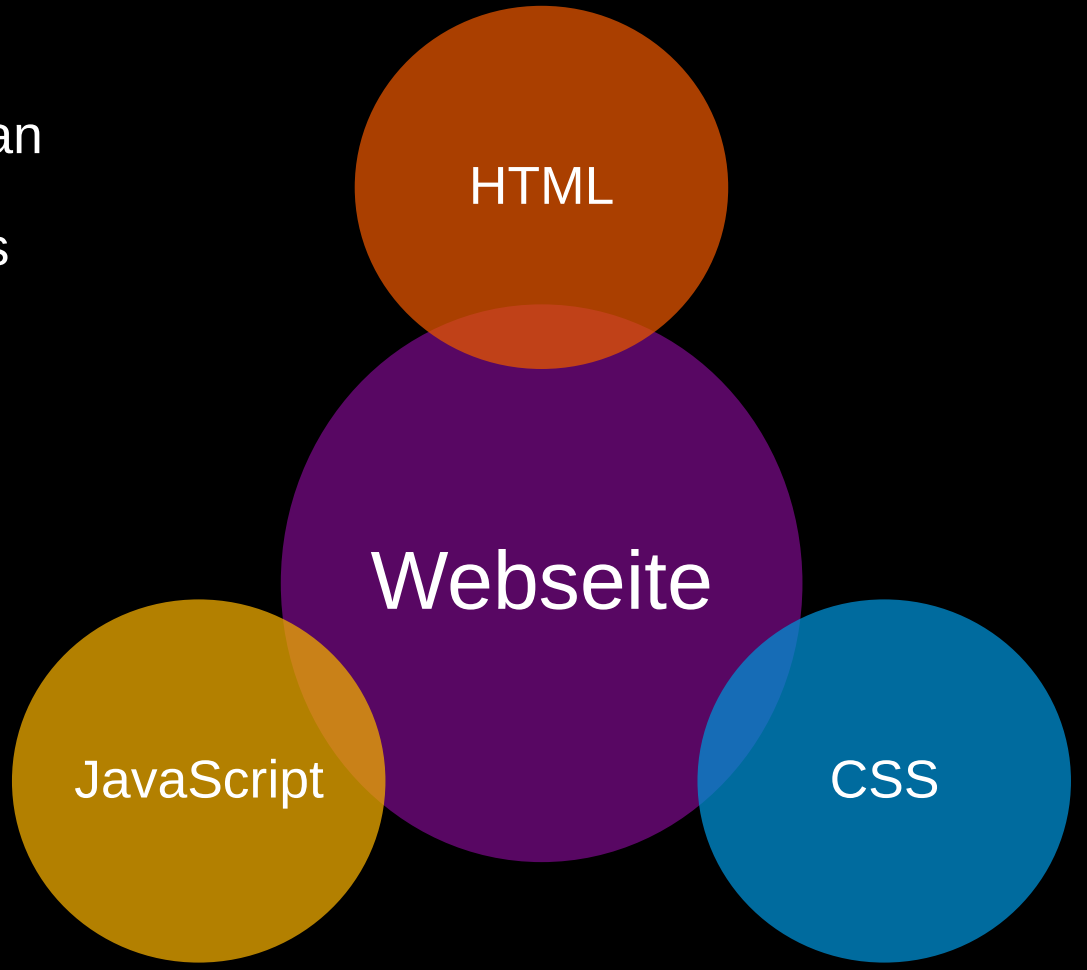
- Wirtschaftsinformatik Business Engineering
- Seit: 2017
- Alter: 21

Lisa Rizzo

- Wirtschaftsinformatik Business Engineering
- Seit: 2019
- Alter: 18

Webseiten

- Webseiten bestehen aus Quellcode
- Quellcode wird vom Browser interpretiert = Bauplan
- 3 Sprachen werden zur Erstellung des Quellcodes benötigt
  - **HTML** strukturiert den Inhalt
  - **CSS** gestaltet die HTML-Elemente
  - **JavaScript** steuert das Verhalten



Tools zur Entwicklung von Webseiten

- Texteditor (z.B. Notepad, Wordpad etc. / z.B. *Notepad++, Visual Studio Code*)
  → Nachteil von einfachen Editoren: keinerlei Unterstützung bei der Entwicklung
  → Vorteil von professionelle Texteditoren: Reihe an Funktionen, die das Entwickeln erleichtern (z.B. Syntax-Highlighting, Code-Vervollständigung etc.)

- Browser (z.B. Google Chrome, Mozilla Firefox etc.)
  → standardmäßig die Entwicklertools über ***F12*** aufrufen
  → Erleichtern das Analysen von Webseiten und das Aufspüren von Fehlern

HTML: Aufbau der Webseite

HTML

- HTML steht für **H**yper**t**ext **M**arkup **L**anguage
- HTML ist kein Programmiersprache!
- HTML = **Auszeichnungssprache**

  → beschreibt den Aufbau von Webseiten
- erste Spezifikation zu HTML wurde 1992 veröffentlicht
- HTML ist eine der Kernsprachen des *World Wide Web*

HTML-Tags

- **Tags** = Anweisungen in spitzen Klammern
- Versorgen die Browser mit Informationen über den Aufbau der Seite
- Legen Struktur über Text und Bilder
- Tags können beliebig ineinander verschachtelt werden
- Öffnende Tags (***<Tag>***) vs. schließende Tags (***</Tag>***)

```
<TagA>
  <TagB>Inhalt 1</TagB>
  <TagC>Inhalt 2</TagC>
</TagA>
```

Attribute

- **Attribute** bringen zusätzliche Informationen in ein Tag (z.B. den Pfad auf eine Bilddatei)
- Attribute und Attributs-Ausprägungen werden innerhalb des öffnenden Tags festgelegt

```
<TagA attributA="Wert 1" attributB="Wert 2">
  <TagB attributC="Wert 3">Inhalt 1</TagB>
  <TagC>Inhalt 2</TagC>
</TagA>
```

Grundgerüst eines HTML-Dokuments

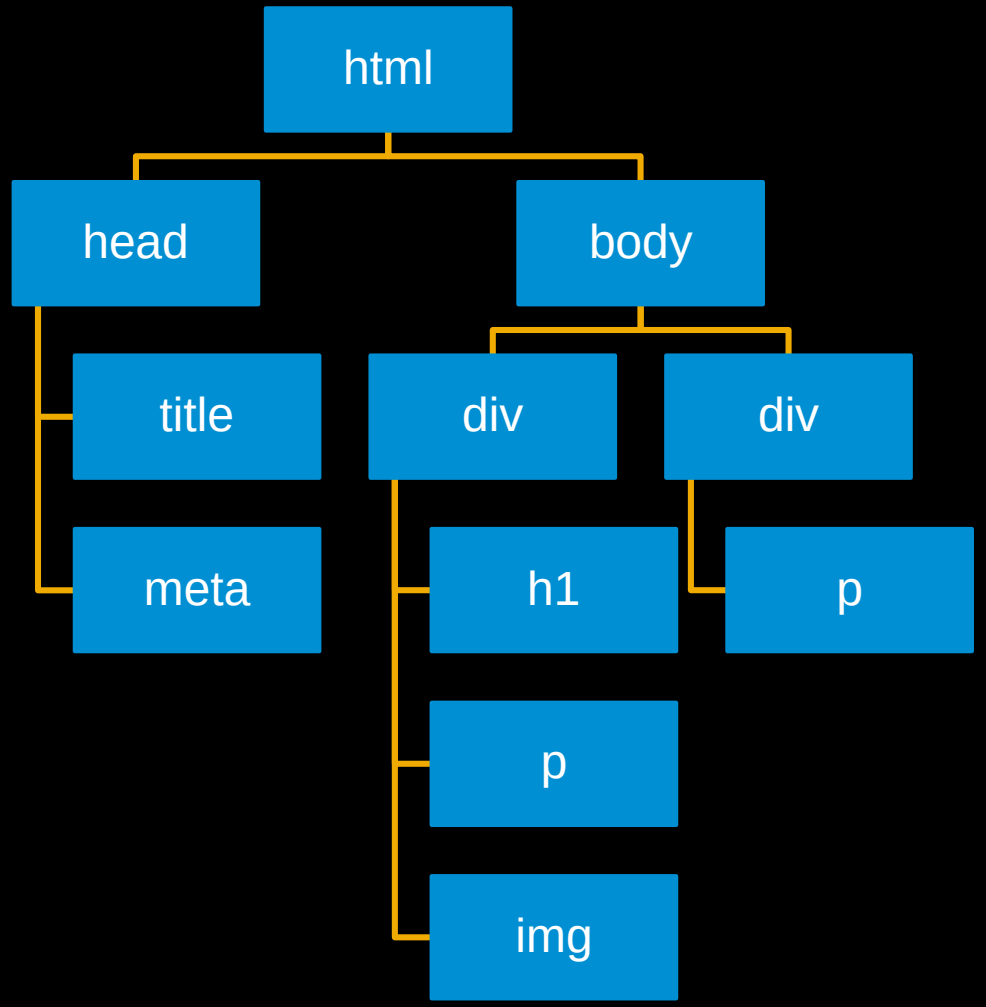
- Grundgerüst eines HTML-Dokuments:
  - → Kopfbereich (Tag ***<head>***)
  - → Inhaltsbereich (Tag ***<body>***)
- **Hinweis**: Kommentare (Tag *<!---->*) werden vom Browser ignoriert

```
<!DOCTYPE html>
<html>                          <!--Beginn des HTML-Dokuments-->
  <head>                        <!--Beginn des Kopfbereichs-->
    <title>Titel</title>        <!--Titel der Webseite-->
  </head>                       <!--Ende des Kopfbereichs-->
  <body>                        <!--Beginn des Inhaltbereichs-->
    Inhalt                      <!--Inhalt der Webseite-->
  </body>                       <!--Ende des Inhaltbereichs-->
</html>                         <!--Ende des HTML-Dokuments-->
```

Document Object Model

- Inhalte eines HTLM-Dokuments werden vom Browser Schritt für Schritt in eine Objektstruktur überführt
- Objektstruktur = DOM (***Document Object Model***)

  → Zugriff auf einzelne Elemente
- **Hinweis**: Attribut ***id*** identifiziert ein HTML-Element
- **Hinweis**: Attribut ***class*** fasst verschiedene HTML-Elemente zu einer gemeinsamen Klasse zusammen



Aufgabe 1

- Erstelle ein HTML-Dokument (*index.html*), welches den Text „Herzlich Willkommen auf der Webseite von *Name*“ anzeigt
- **Optional**: mit dem Titel: „Webseite von *Name“,* da es in unserer Entwicklungsumgebung nicht angezeigt wird

Inhaltsbereich

- Inhaltsbereich (Tag **<*body*>**)
  → sämtliche Inhalte der Webseite werden festgelegt z.B.
  - Texte
  - Aufzählungen
  - Grafiken
  - Verweise

Attribut	Beschreibung	Beispiel
text	Textfarbe	**"white"**
link	Farbe eines Verweises	**"red"**
vlink	Farbe eines besuchten Verweises	**"blue"**
topmargin	Abstand nach oben in Pixel	**50**
leftmargin	Abstand nach links in Pixel	**50**
bgcolor	Hintergrundfarbe	**"black"**
background	Hintergrundbild	**"sap.png"**

Textformatierungen

- Texte können in HTML über vordefinierte Tags angepasst werden
- **Hinweis**: Umlaute und Sonderzeichen müssen in HTML über spezielle Codes abgebildet werden
- **Hilfe**: https://www.adfreak.de/blog/html-umlaute-und-sonderzeichen/

```
<h1>gr&ouml&szligte &Uumlberschrift</h1>
<h6>kleinste &Uumlberschrift</h6>
<b>fetter Text</b>
<i>kursiver Text</i>
<u>unterstrichener Text</u>
<s>durchgestrichener Text</s>
<center>zentrierter Text</center>
<sup>hochgestellter Text</sup>
<sub>tiefgestellter Text</sub>
<marquee>Lauftext</marquee>
```

Trennlinien, Zeilenumbrüche und Absätze

- Tag **<*hr*>** erzeugt eine horizontale Trennlinie (***horizontal row***)
- **Hinweis**: beim Attribut ***align*** sind die Werte:
  - ***left*** (linksbündig)
  - ***right*** (rechtsbündig)
  - ***center*** (zentriert)
  - ***justify*** (Blocksatz) möglich

Attribut	Beschreibung	Beispiel
width	Breite in Pixel	**500**
size	Höhe in Pixel	**5**
align	Ausrichtung	**"center"**
color	Farbe	**"white"**
noshade	Schattierung deaktivieren	

- Tag **<*br*>** erzeugt Zeilenumbrüche (***breaks***)
- Tag **<*p*>** erzeugt Absätze (***paragraphs***)

Attribut	Beschreibung	Beispiel
align	Ausrichtung	**"center"**

Aufzählungen

- geordnete (Tag ***<ol>***) vs. ungeordnete (Tag ***<ul>***)
- Einzelne Einträge werden mit dem Tag ***
*** geöffnet und mit dem Tag ***
*** geschlossen

```
<!--ungeordnete Aufzählung-->
<ul>
  <li>Eintrag 1</li>
  <li>Eintrag 2</li>
  <li>Eintrag 3</li>
</ul>

<!--geordnete Aufzählung-->
<ol>
  <li>Eintrag 1</li>
  <li>Eintrag 2</li>
  <li>Eintrag 3</li>
</ol>
```

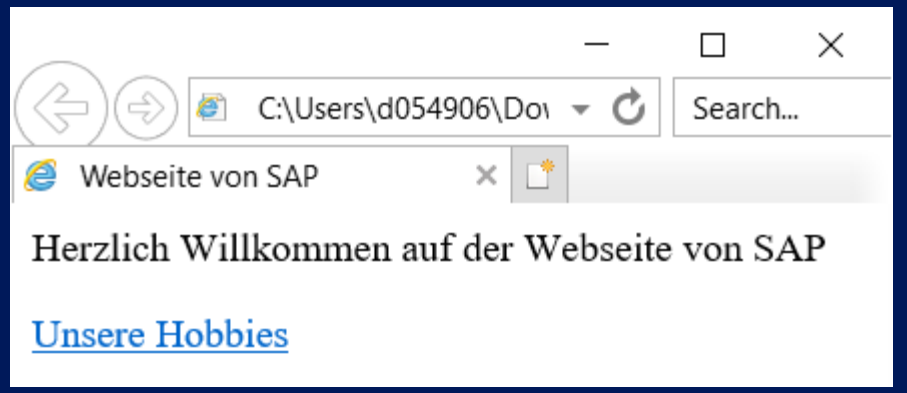
Verweise (*anchors*)

- Navigation innerhalb der Webseite und zu externen Dokumenten
- **Hinweis**: Attribut ***href*** können interne Seiten, externe Seiten sowie Verweise auf HTML-Elemente als Wert zugewiesen werden
- **Hinweis**: Attribut ***target*** der Wert ***_blank*** zugewiesen, wird der Verweis in einem neuen Tab geöffnet

Attribut	Beschreibung	Beispiel
href	Verweis	**"http://www.sap.com"**
title	Tooltip	**"SAP.com"**
target	Ziel	**"_blank"**

Aufgabe 2

- Erstelle ein HTML-Dokument (*hobbies.html*) welches eine Aufzählung Deiner Hobbies (samt passender Überschrift) anzeigt
- Erweitere das HTML-Dokument *index.html* aus Aufgabe 1 um einen Link auf das HTML-Dokument *hobbies.html*



Bilddateien

- Tag ***<img>*** = Bilddateien einbinden
- **Hinweis**: liegen das HTML-Dokument und die Bilddatei nicht im gleichen Verzeichnis muss der Pfad zur Bilddatei ausgehend vom HTML-Dokument angegeben werden

Attribut	Beschreibung	Beispiel
src	Pfad zur Bilddatei	**"image.png"**
alt	Alternativtext	**"Grafik"**
width	Breite in Pixel	**200**
height	Höhe in Pixel	**200**
border	Rahmenstärke in Pixel	**5**
align	Ausrichtung	**"center"**

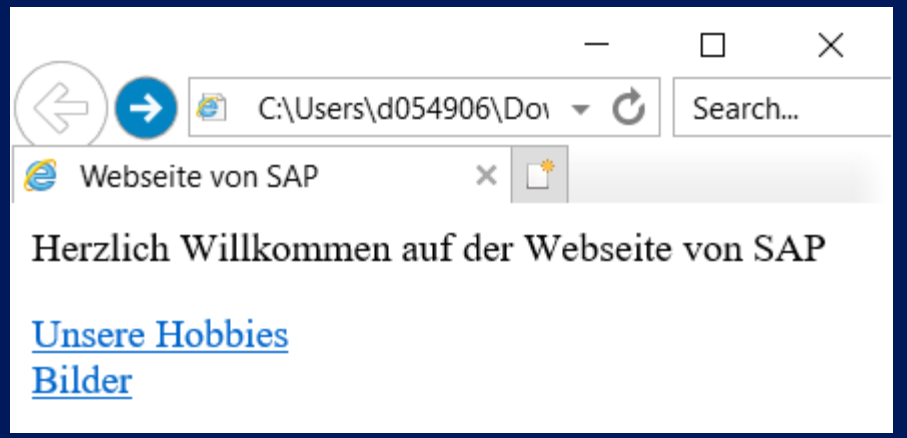
Bereiche (*division*)

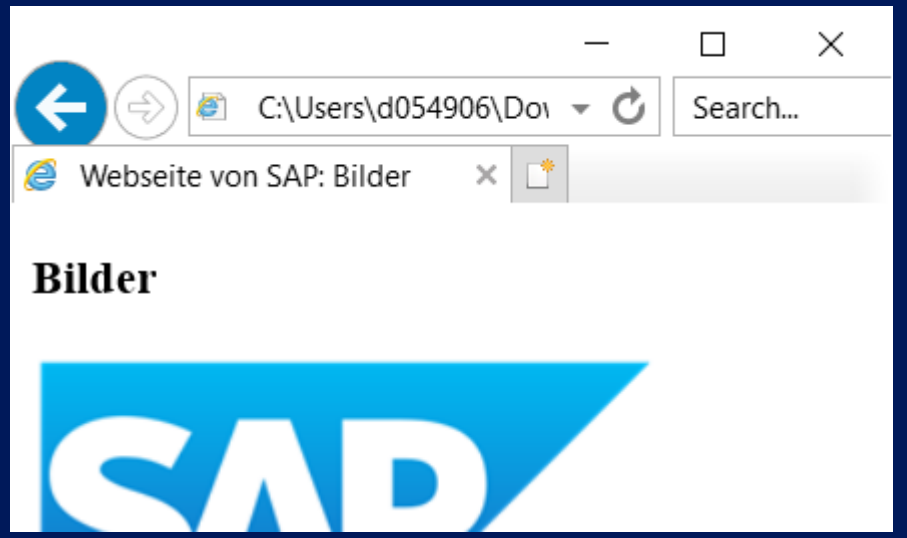
- Tag *** *** = werden mehrere HTLM-Elemente zusammengefasst
- **Hinweis**: standardmäßig ist der Rahmen eines Bereichs nicht sichtbar

```
<div>
  <h1>Überschrift</h1>
  <p>Paragraph</p>
  <ol>
    <li>Eintrag 1</li>
    <li>Eintrag 2</li>
    <li>Eintrag 3</li>
  </ol>
</div>
```

Aufgabe 3

- Erstelle ein HTML-Dokument (*images.html*) welches ein Bild anzeigt
- Erweitere das HTML-Dokument *index.html* aus Aufgabe 2 um einen Link auf das HTML-Dokument *images.html*
- **Achtung**: Da in der Entwicklungsumgebung Bilder nicht direkt eingefügt werden können, werden wir euch später eine Alternative zeigen





CSS: Aussehen der Webseite

CSS

- CSS steht für **C**ascading **S**tyle **S**heets
- CSS ist keine Programmiersprache!
- CSS = **Stylesheet-Sprache**

→ beschreibt das Aussehen von Webseiten

- CSS wurde 1994 veröffentlicht und ist eine Kernsprache des *World Wide Web*

Einbinden von Stylesheets

- CSS-Formatierungen können unterschiedlich eingebunden werden:
  - Inline-Style
  - interne Stylesheets
  - externen Stylesheets
- **Hinweis**: im Rahmen des Workshops wird nur auf das Einbinden externer Stylesheets eingegangen

```css
/* CSS-Datei style.css */
body {
  color: white;
  background-color: black;
}
```

```html
<!--HTML-Dokument index.hmtl-->
<html>
  <head>
    <link rel="stylesheet" href= "style.css"/>
  </head>
  <body>
  </body>
</html>
```

Syntax

- CSS-Formatierungen bestehen aus einem **Selektor**, mehreren **Attributen** sowie den dazugehörigen **Werten**
- **Selektoren** legen fest, welche CSS-Formatierungen für welche Elemente gelten
- **Hinweis**: jede Formatierung muss mit einem Semikolon abgeschlossen werden
- **Hinweis**: Kommentare beginnen in CSS mit /* und enden mit */
- **Hinweis**: Neben selbst festgelegten Klassen gibt es auch eine Reihe vordefinierter Pseudoklassen

```
* { color: white; }          /* Selektor: alle Elemente */
h3 { color: red; }           /* Selektor: HTML-Element */
#id1 { border-color: blue; } /* Selektor: Element-Id */
.classA { color: green; }    /* Selektor: Klasse */
a:visited { color: yellow; } /* Selektor: Pseudoklasse */
h3:hover { color: blue; }    /* Selektor: Pseudoklasse */
```

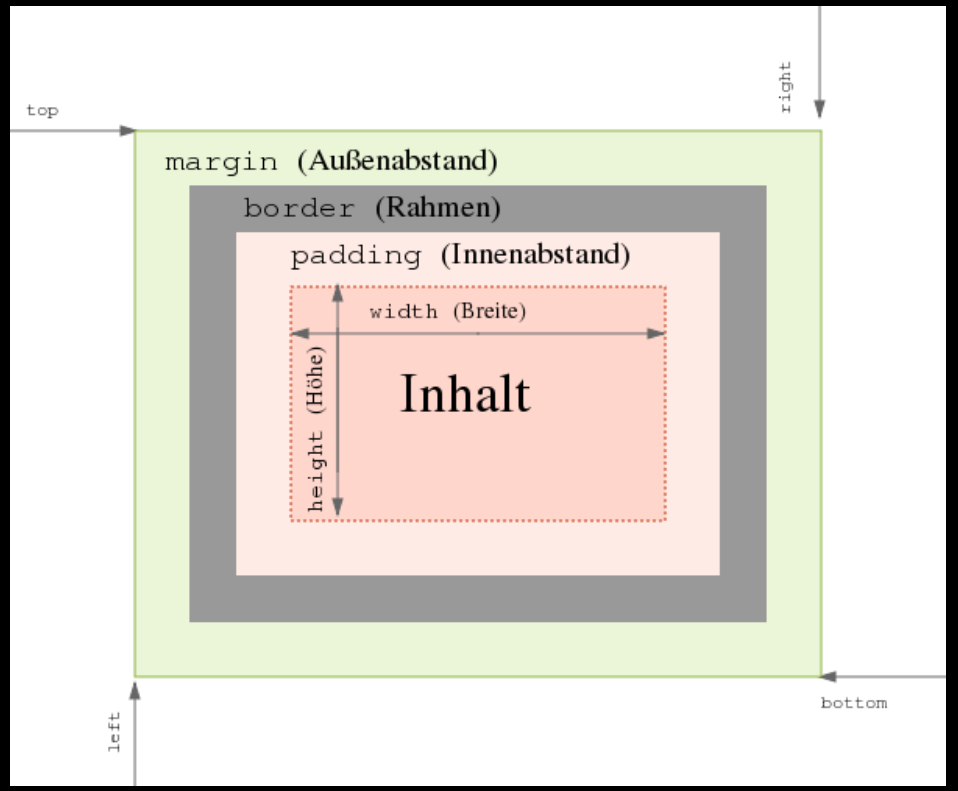
Farbwerte

- Farbwerte können unterschiedlich angegeben werden:
  - Textform (vordefinierte Farben)
  - Hexadezimalzahl
  - RGB- bzw. RGBA-Wert
- **Hinweis**: bei RGB werden Farben durch Angabe der Rot-, Grün- und Blauwerte festgelegt, bei RGBA wird zusätzlich noch der Alphakanal (Transparenz) angegeben

```
color: white;                  /* Text */
color: #ff0000;                /* Hexadezimal */
border-color: rgb(0, 0, 255); /* RGB */
color: rgba(0, 255, 0, 0.5);  /* RGBA */
```

Box-Modell

- HTML-Elementen werden über das Zeichnen von Rechtecken dargestellt
- Das Box-Modell legt dabei die Größe der gezeichneten Rechtecke fest
  - Breite und Höhe des Inhalts (***width***, ***height***)
  - Innenabstand (***padding***)
  - Breite des Rahmens (***border***)
  - Außenabstand (***margin***)



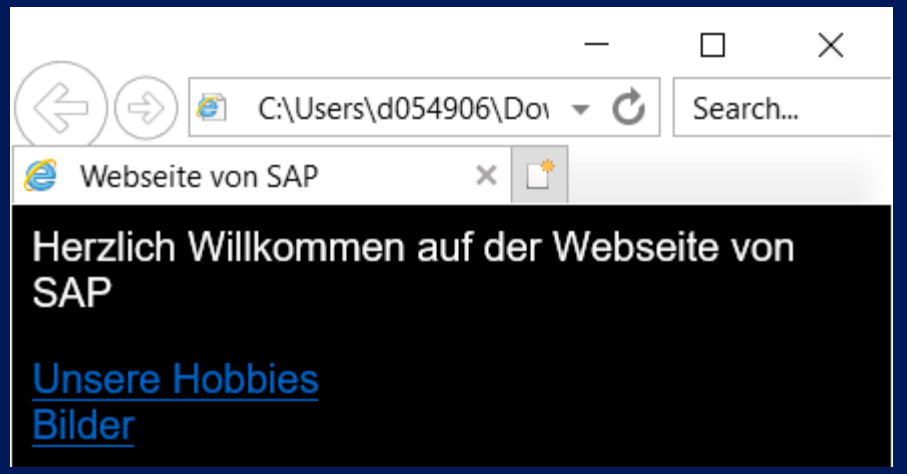
Text-, Rahmen- und Aufzählungsformatierungen

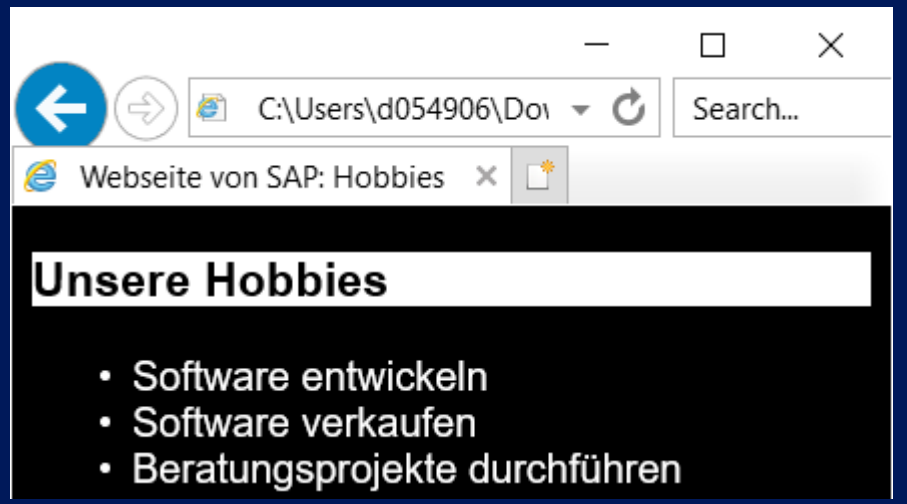
- **Hinweis**: Attribut ***style*** sind beispielsweise die Werte:
  - ***solid*** (ausgefüllt)
  - ***dotted*** (gepunktet)
  - ***groove*** (geritzt)
- **Hinweis**: Attribut ***list-style-type*** sind für ungeordnete Aufzählungen die Werte:
  - ***disc*** (Punkt)
  - ***circle*** (kreis)
  - ***square*** (Quadrat)
- bei geordneten Aufzählungen die Werte:
  - ***decimal*** (Ziffer)
  - ***upper-roman*** (römische Ziffer)
  - ***lower-alpha*** (Kleinbuchstabe) möglich

```
/* Textformatierungen */
color: white;                       /* Schriftfarbe */
font-size: 15px;                    /* Schriftgröße */
font-weight: bold;                  /* Schriftdicke */
font-style: italic;                 /* Schriftstil */
font-family: "Arial", sans-serif; /* Schriftart */
text-decoration: underline;         /* Dekoration */

/* Rahmenformatierungen */
border-color: rgb(0, 0, 255);     /* Linienfarbe */
border-style: solid;                /* Linienstil */
border-radius: 10px;                /* Radius der Ecken */
border-width: 2px;                  /* Linienstärke */

/* Aufzählungsformatierungen */
list-style-type: circle;            /* Aufzählungszeichen */
```

Aufgabe 4

- Erstelle eine CSS-Datei (*style.css*) und lege dort entsprechende Formatierungsregeln für Deine Webseite fest
- Binde die CSS-Datei *style.css* in die HTML-Dokumente aus Aufgabe 3 ein





JavaScript: Verhalten der Webseite

JavaScript

- JavaScript = **Skriptsprache**
- 1995 entwickelt von der Firma Netscape

  → dynamische Webseiten erstellen
- ist **nicht** mit der Programmiersprache Java „verwandt“
- JavaScript = eine der beliebtesten Programmiersprachen

  → Quasi-Standard für die Entwicklung webbasierter Oberflächen

Einbinden von JavaScript-Dateien

- direkt vs. externen JavaScript-Dateien in HTML
- **Hinweis**: im Rahmen des Workshops wird nur auf das Einbinden externer JavaScript-Dateien eingegangen
- **Hinweis**: Kommentare können in JavaScript mit //, Kommentarblöcke mit /* und */ eingefügt werden

```
/* JavaScript-Datei index.js */
console.log('Hello World');
```

```
<!--HTML-Dokument index.hmtl-->
<html>
  <head>
    <script src="index.js"></script>
  </head>
  <body>
  </body>
</html>
```

Grundlegende Sprachelemente in JavaScript

- Variablen ermöglichen das Speichern von Informationen
- Variablen = Schlüsselwort **let** deklariert
- mathematische Operationen:
  - **+** (Addition)
  - **-** (Division)
  - ***** (Multiplikation)
  - **/** (Division)
- **Hinweis**: mit Hilfe von ***console.log()*** können Informationen auf der Konsole ausgegeben werden

```
let result;          // Deklaration einer Variablen
result = 4 * 3;      // Verwenden math. Operationen
console.log(result); // Ausgabe auf der Konsole
```

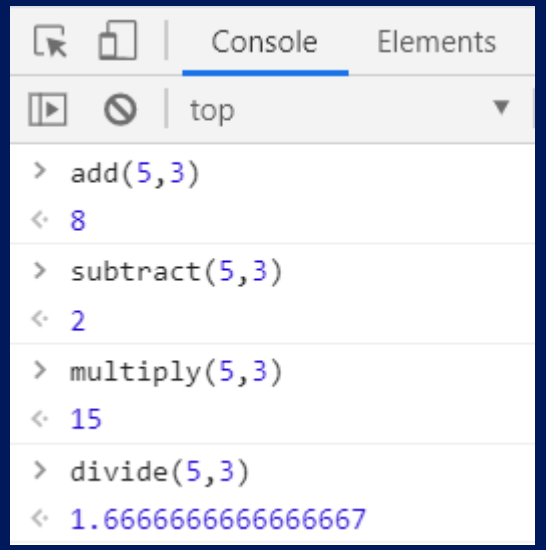
Funktionen

- Funktionen ermöglichen die Kapselung von Quellcode
- **Parameter** ermöglichen es, der Funktion beim Aufrufen Werte mitzugeben
- Schlüsselwort ***return*** ermöglicht die Rückgabe eines Wertes an den Aufrufer der Funktion

```
function multiply(a, b) { // Definition einer Funktion
  let result;
  result = a * b;
  return result;          // Rückgabe eines Wertes
}
```

Aufgabe 5

- Erstellen eine JavaScript-Datei (*calculator.js*) und definiere dort für jede Rechengrundart eine Funktion
- Erstelle ein HTML-Dokument (*calculator.html*) und binde dort die JavaScript-Datei *calculator.js* sowie die CSS-Datei *style.css* ein
- Erweitere das HTML-Dokument *index.html* aus Aufgabe 4 um einen Link auf das HTML-Dokument *calculator.html*



HTML-Formularelemente

- für interaktive Webseiten gibt es verschiedene HTML-Formularelemente:
  - Bezeichner (Tag ***<label>***)
  - Eingabefelder (Tag ***<input>***)
  - Drucktasten (Tag ***<button>***)
- **Hinweis**: Attribut ***type*** (Tag ***<input>***) stehen die Werte zur Verfügung:
  - ***text*** (Textfeld)
  - ***password*** (Passwortfeld)
  - color (Farbwähler)
  - ***radio*** (Auswahlschalter)
  - ***checkbox*** (Kontrollkästchen)

```
<label for="input1">Textfeld</label>
<input type="text" id="input1" maxlength=30/>
<button type="button">Drucktaste</button>
```

Zugriff auf HTML-Elemente

- erfolgt über die Methode ***document.getElementById()***
- Attribut ***value*** auf den Wert eines HTML-Formularelements
- Attribut ***innerHTML*** auf „normale“ HTML-Elemente

```
let input1Value = document.getElementById('input1').value;
let p1Value = document.getElementById('p1').innerHTML;
```

Ereignisbehandlung

- Bei verschiedenen Aktionen (z.B. dem Betätigen einer Drucktaste) werden Ereignisse ausgelöst
- mit Hilfe von JavaScript-Funktionen kann reagiert werden
- jedes Ereignis hat eine dazugehörige Eigenschaft (z.B. die Eigenschaft ***onClick***)

```
/* JavaScript-Datei index.js */
function print() {
  let input1Value = document.getElementById('input1').value;
  document.getElementById('p1').innerHTML = input1Value;
}
```

```
<!--Auszug HTML-Dokument index.hmtl-->
…
<label for="input1">Eingabe</label>
<input type="text" id="input1" maxlength=30/>
<button type="button" onclick="print()">Ausgeben</button>
<p id="p1"/>
…
```

Aufgabe 6

- Erweitere das HTML-Dokument *calculator.html* aus Aufgabe 5 um zwei Eingabefelder samt Bezeichner, vier Drucktasten sowie ein Ausgabefeld
- Passe die JavaScript *calculator.js* so an, dass die Werte zur Berechnung des Ergebnisses aus den Eingabefelder gelesen werden und dass das Ergebnis im Ausgabefeld angezeigt wird
- **Hinweis**: die Funktion *Number()* ermöglicht die Umwandlung einer Zeichenkette in eine Zahl