**How to HTML**

Zu Beginn der Vorlesung haben wir uns mit den Grundlagen von HTML beschäftigt. Grundlegende Tags, wie html, head, body, p(paragraph), h1(und h2,...) und script.  
Tags müssen geöffnet und geschlossen werden. Mit html wird z.B. geöffnet und /html geschlossen.  
Eine HTML Seite, die mit Script Tags JavaScript nutzt, könnte so aussehen:



In manchen Tags, wie script oder img für image stehen wichtige Bestandteile im Öffnungstag dabei.

Der Stil in HTML kann durch den style Tag bestimmt werden. Hier können z.B. Farbe, Schriftgröße und Schriftart festgelegt werden.  
Der Inhalt oder das Layout auf HTML Seiten kann auch verändert werden. Wie durch das Klicken eines Buttons:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Hypertext und Links

In HTML kann man zu verschiedenen Dateien verlinken und diese durch Links aufrufen. Links werden mit dem a Element erstellt und das href Element gibt die Zielseite an.    
Dokumente (html) können in Ordner gegliedert und dadurch organisiert werden. Um auf sie zu verlinken, muss man dann nur im Link den Ordner hinzufügen oder durch ../ aus dem jetzigen Ordner herausgehen ("eins hoch").  
Wie z.B. für die Links der Navigationsleiste dieser Website.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Eingabefelder

Elemente einer Website werden in Formularen mit HTML Code definiert. Sie werden mit Hilfe der Form-Tags definiert. (In JavaScript: Form-Objekt)  
Zugriff auf das Formular bekommt man mit den folgenden Elementen/Formularen:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Um Usereingaben anzunehmen, gibt es die folgenden Eingabefelder:

**Einzeiliges Textfeld**

<input type="text" id="eingabe" name"eingabefeld" size="20" value="Textfeld")

**Mehrzeiliges Textfeld**

<textarea id="eingabe2" name="eingabefeld2" cols="20" rows="5" wrap="virtual"›Textfeld2</textarea>

**Passwort-Objekt**

<input type="password" id="pwd" name="password" size="20"')

**Hidden-Objekt**; Text-Objekt,welches nicht angezeigt wird

<input type="hidden" id="hdn" name="hidden" value"geheim">

**Button-Objekt**

<input type="button" value="Button 1"'>

**Submit-Objekt**

<input type="submit" value="Abschicken">

**Reset-Objekt**

<input type="reset" value="Zurücksetzen">

**Checkbox-Objekt**

<input type="checkbox" id="option1" name="option1" checked=" checked"›Option 1

<input type="checkbox" id="option2" name="option2" checked="checked"›Option 2

// wenn nur eine Option wählbar sein soll, muss das Checked="checked" hinter option2 entfernt werden

**Select-Objekt Einfachauswahl**

<select id="liste" name= liste">

<option selected="selected"›option 1</option>

<option>option 2</option)

<option>option3</option>

</select>

**Select-Objekt Mehrfachauswahl**

<select id="liste" name="liste">

<option selected="selected"›option 1</option) <option selected="selected"›option 2</option>

<option>option3</option>

</select>

**EXKURS: CSS**

Mit css kann man den Style(: die Farbe, Typographie, Ränder und vieles mehr) von Websiten festlegen.  
Es gibt externe css Dateien, die man in der HTML Datei verlinken kann. Außerdem kann man in style Tags im HTML head mit css Code den Style festlegen.

### Verlinkung von css-stylesheets:

Wie oben aufgelistet sieht man hier verschiedene Arten der Nutzung von css.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Beispiele für css-Code:  
@import url('https://fonts.googleapis.com/css2?family=Inter+Tight:wght@600&display=swap');  
body,h1 {font-family:'Inter Tight', sans-serif;} // Für den gesamten body (keine Titel, normaler Text) und h1 wird diese Schriftart gesetzt.  
h1 {letter-spacing: 3px} //Für h1 wird hier der Abstand zwischen Buchstaben festgelegt  
h2,h3{font-family:'Inter Tight', sans-serif; font-weight: 600; color:brown} //Für h2 und h3 werden diese Einstellungen gesetzt  
  
In HTML können Elemente Styles erben, es ist daher einfacher, Styles direkt in den Body zu coden, dann werden sie direkt in die einzelnen Elemente (Bilder etc.) überschrieben.  
Falls man aber etwas anderes für das Element möchte, kann man im Element etwas anderes festlegen. Das gilt dann für das Element, anstatt das im Body bereits festgelegte.

### CSS Klassen

Um Styles den einzelnen HTML Elementen zu vergeben, werden Klassen definiert. im Öffnungs-html Tag steht dann nach z.B. h3 class="beispielclass".  
Diese Klassen können mehrfach und für verschiedene Elemente genutzt werden.

## **DOM und Elementveränderung**

### Document Object Model

Das DOM (Document Object Model) ist der "Baum" der Elemente und Attribute in HTML, durch das DOM kann man die (HTML) Inhalte verändern, manipulieren und analysieren. Also: Das DOM stellt jedes Element in einem Dokument als ein Objekt dar, das von einem Programm aufgerufen und manipuliert werden kann. Zum Beispiel kann ein Programm das DOM verwenden, um den Text in einem Paragraphen-Element zu ändern, ein neues Element in eine Liste einzufügen oder den Inhalt eines Formulars abzurufen. Das DOM wird auch von JS Bibliotheken und Frameworks verwendet, um (die Interaktion mit dem User und) die Veränderung des Inhalts /der Struktur von Webseiten zu ermöglichen.  
Jeder/s Element/Attribut/Text wird als "Knoten" im DOM dargestellt.  
Mit den folgenden Methoden spricht man Knoten im DOM an:

### getElementByID()

Ist zwar ein JavaScript Thema, wird aber hier gebraucht: Die getElementById() Methode ist eine Funktion in JavaScript, die verwendet wird, um ein Element mit einer bestimmten ID im DOM (Document Object Model) zu finden und zu manipulieren. Die ID ist ein Attribut, das jedem Element in einem HTML-Dokument zugewiesen wird und dazu verwendet wird, das Element eindeutig zu identifizieren. Um die getElementById() Methode zu verwenden, gibt man einfach den Namen der ID als Argument an und speichert das Ergebnis in einer Variablen.  
Ein Beispiel dafür sieht zum Beispiel so aus:  
  
let element = document.getElementById('mein-element');  
  
Im Beispiel würde die getElementById() Methode das Element mit der ID mein-element im DOM suchen und es in der Variablen element speichern. Man kann dann auf die Eigenschaften und Methoden des Elements zugreifen, um es zu manipulieren oder Infos abzurufen. Ändern kann man das ganze jetzt durch innerHTML  
  
element.innerHTML = 'Neuer Inhalt';

### getElementsByTagName()

Das funktioniert ähnlich wie ID oben, aber diesmal ist es nicht nur ein Element, sondern Elements, also mehrere. Man kann sich z.B. alle Paragraphen(p), h1 Überschriften oder Links einer Seite ausgeben lassen. Das Ergebnis dieser Methode ist kein einzelner Knoten, sondern eine Liste von Knoten.  
Achtung Beispiel:  
  
let elements = document.getElementsByTagName('p');  
  
elements[0].innerHTML = 'Neuer Inhalt';  
  
Die Methode hier ist besonders nützlich, wenn man alle Elemente eines bestimmten Typs im DOM finden und manipulieren möchte, zum Beispiel alle p-Elemente oder alle image-Elemente.