HTML, CSS und Javascript

Im Zusammenspiel

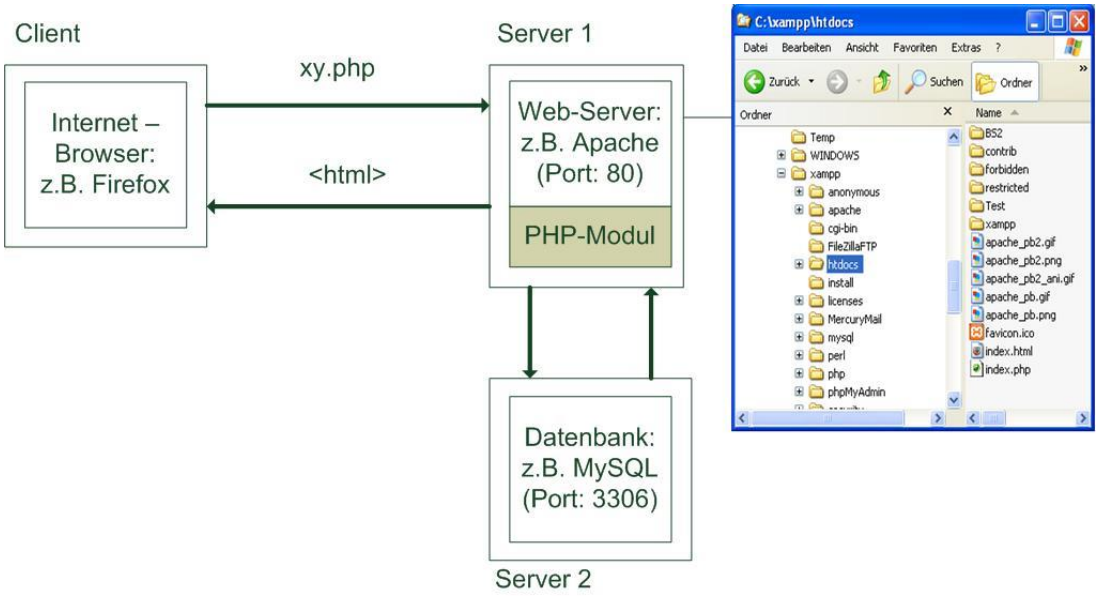
Hinweis: Für diese und weiterführende Technologien gibt es eine sehr gute Online-Dokumentation, welche unter folgenden Link erreichbar ist: https://www.w3schools.com/.

In der **clientseitigen** Webentwicklung sind HMTL, CSS und Javascript eng miteinander verbunden und jedes der drei Elemente stellt andere Ansprüche an den Entwickler. Grundsätzlich kann man von drei völlig unterschiedlichen "Sprachen" sprechen, die sich untereinander ergänzen.

Abgrenzung clientseitige und serverseitige Programmierung

Das Web basiert seit jeher auf einem einfachen Prinzip. Verschiedenste Inhalte werden von Webservern bereitgestellt und können über HTTP oder FTP von Clients abgerufen werden. Bei den Clients handelt es sich um **Browser** wie Mozilla Firefox oder Google Chrome, die auf dem System des Users installiert sind und dort ausgeführt werden. **Webserver** wie Apache oder NGINX hingegen sind Bestandteil von den physischen Webservern. Sie werden in dieser Umgebung auch installiert, ausgeführt und ermöglichen dem jeweiligen Client den Zugriff auf die Inhalte.

Während statische Inhalte wie z. B. Bilder nur übertragen und dargestellt werden, funktionieren **dynamische Inhalte** wie ein Blog-Beitrag, eine Auflistung von Beiträgen, ein Drop-down-Menü oder eine Suchabfrage innerhalb der Website nur **mithilfe von Skripten**. Diese müssen mit der entsprechenden Script-Sprache ausgeführt und interpretiert werden, was sowohl am Server als auch am Client (Browser) geschehen kann. Aus diesem Grund unterscheidet man zwischen serverseitiger und clientseitiger (Web)Programmierung.



Clientseitige Programmierung/Webentwicklung

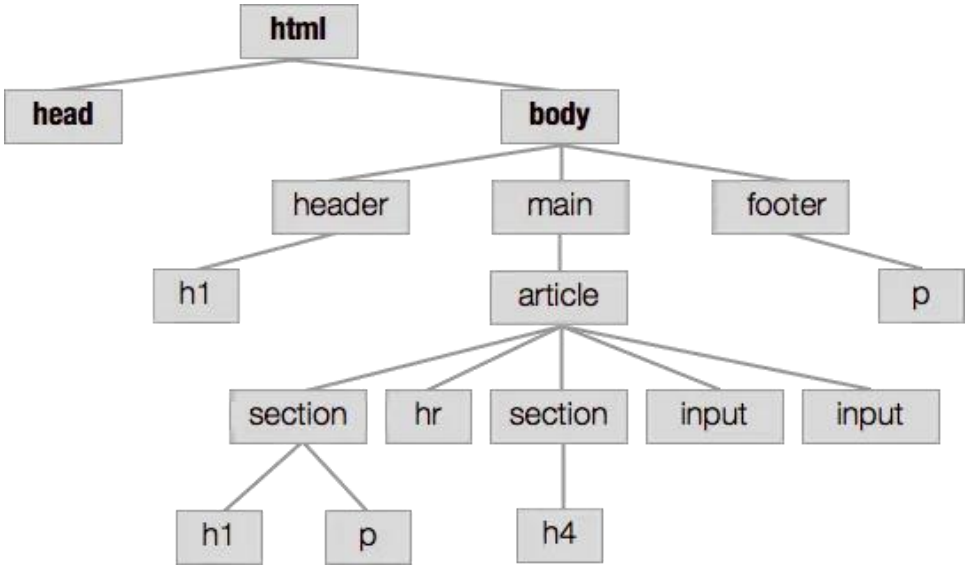
Clientseitige Webentwicklung wird von Webentwicklern dazu verwendet, Projekte mit statischen und dynamischen Inhalten zu realisieren. Anders als bei der Server-Side-Variante werden die programmierten Skripte allerdings nicht vom Server, sondern **vom zugreifenden Client ausgeführt** und verarbeitet. Zu diesem Zweck bettet man die Skripte entweder in das HTML-Dokument ein oder erstellt eine separate Datei, die man im HTML-Dokument integriert. Ruft der User nun eine Webseite oder Webanwendung mit einem solchen Client-Side-Script auf, sendet der Webserver das HTML-Dokument sowie das Skript an den Browser, der dieses ausführt und das Endergebnis präsentiert. Clientseitige Skripte können darüber hinaus **konkrete Instruktionen für den Webbrowser** beinhalten, wie dieser auf bestimmte Aktionen des Nutzers wie z. B. auf einen Button-Klick reagieren soll. Oftmals muss der Client dazu keinen erneuten Kontakt zum Webserver aufbauen.

Da die Skripte im Browser des Nutzers ausgeführt werden, hat er – anders als bei Server-Side-Scripts – die Möglichkeit, den Quellcode einzusehen. Die eingesetzte Scriptsprache muss vom Webbrowser verstanden(interpretierbar) werden und darf von diesem nicht blockiert werden.

Die bedeutendste clientseitige Scriptsprache ist **JavaScript**. Sie wurde vom Mozilla-Vorgänger Netscape entwickelt und bereits 1995 veröffentlicht. Sie fand schnell **Verbreitung** und wurde somit zur **universellen Skriptsprache aller relevanten Webbrowser**.

Drei Technologien – mit vielen Unterschieden

Eine Gemeinsamkeit ist jedoch das DOM (Document Object Model). Das DOM beschreibt HTML-, XML- und SVG-Dokumente. Das DOM ist die Schnittstelle für den Zugriff auf Tags, Attribute und Inhalte von HTML-Seiten, XML-Dokumenten und SVG-Grafiken. Der Webbrowser "parst" das HTML-Dokument und verwaltet es in einem DOM. Die Standards des DOM werden durch W3C definiert und beschrieben. W3C (World Wide Web Consortium) setzt diese Standards, um die Interpretation durch unterschiedliche Web-Browser zu vereinheitlichen.

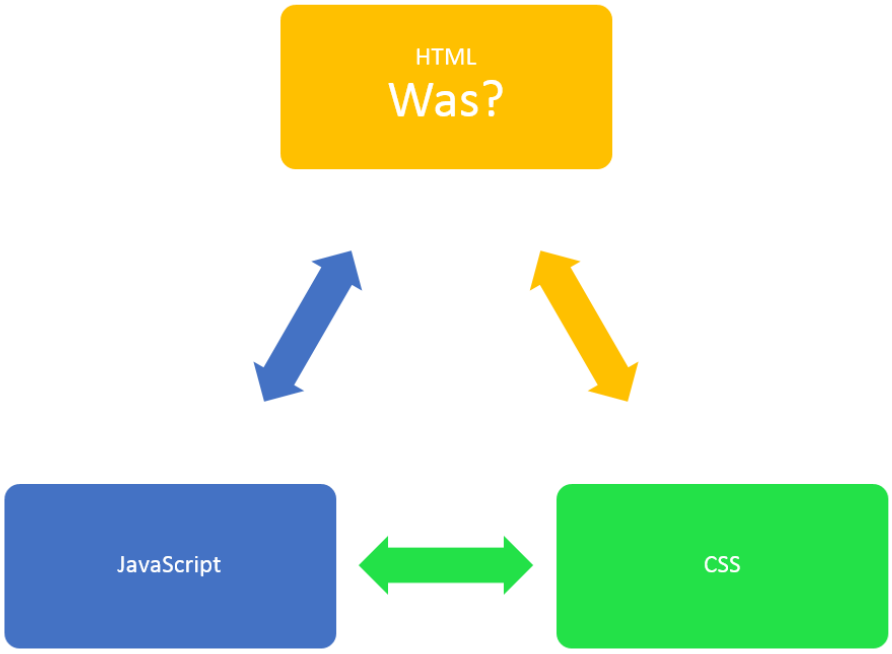


Quelle(Bild):http://lex-learning.de/2016/02/dom/

Die Koexistenz der drei Technologien lässt sich gut in einem Dreieck darstellen und beschreiben. Jede der drei Technologien hat andere Aufgaben und hat eine vollständig konträre Syntax. Jede dieser Sprachen für sich muss man zumindest soweit beherrschen, dass man z. B. aus einem bestehenden Bootstrap-Template eine Website/Webapplikation erstellen kann.

HTML stellt die einzelnen **Elemente** der Seite zur Verfügung, definiert deren **Typ** und die **Struktur,** wie sie zueinander **logisch** angeordnet sind. HTML ist keine Programmiersprache, sondern eine **Auszeichnungssprache** wie die Abkürzung **H**yper**t**ext **M**arkup **L**anguage auch zeigt. Man kann hier also nicht von einer "Programmiersprache" sprechen.

HTML an sich erfüllt die grundlegendste Aufgabe der drei Web-Technologien. Das anzuzeigende Dokument – meist eine Webseite – wird in HTML erstellt.
Mit HTML löst man im Prinzip die Aufgabenstellung: ***Was möchte ich auf meiner Website darstellen? HTML kümmert sich somit primär um das WAS.***



Eine HTML-Beispiel zur Erklärung:

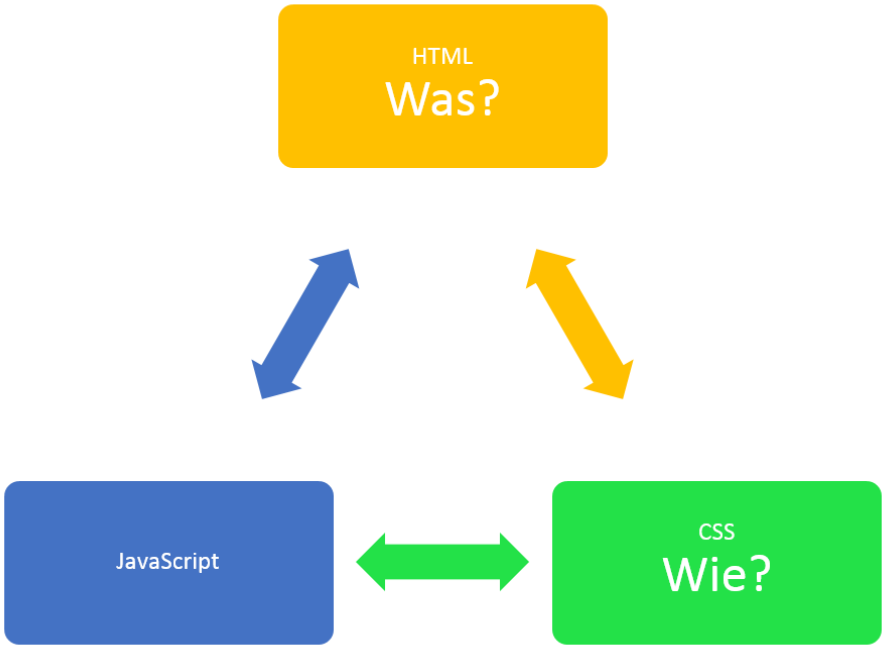
```
<!DOCTYPE html>
<html lang="de">
<head...>
<body>
<h1>Hello, world!</h1>

<div class="row">
    <div class="col-6">
        <p>Dieser Text in einem P-Tag ist ein Blindtext und hat keine Bedeutung. Mittels F
    </div>
    <div class="col-6">
        <div class="actions">
            <button>Text ersetzen</button>
            <button>Buttons einfärben</button>
        </div>
    </div>
</div>
<!-- jQuery (wird für Bootstrap JavaScript-Plugins benötigt) -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<!-- Binde alle kompilierten Plugins zusammen ein (wie hier unten) oder such dir einzelne
<script src="js/bootstrap.min.js"></script>
</body>
</html>
```

HTML stellt die einzelnen Elemente der Seite zur Verfügung, definiert deren Typ und die Struktur wie sie logisch zusammengehören. Es wird der DOM-Baum für das HTML-Dokument aufgebaut. HTML ist

nichts anderes als eine Markup-Language zur Beschreibung von Daten und Elementen.
Eine sehr gute Quelle zum Nachschlagen der einzelnen Tags ist:
https://www.w3schools.com/html/default.asp

CSS-> Styling: gut aussehen soll es auch

CSS steht für Cascading Style Sheets und kümmert sich um die Aufgabenstellung "**Wie sehen die einzelnen Elemente aus"**. Auch mit CSS kann man nicht programmieren. Man "beschreibt" eher das **Aussehen** der einzelnen Elemente und wie sie miteinander ein Layout bilden sollen. Gegenteilig zu HTML, welches die "logische" Formatierung beschreibt, geht es bei CSS um die **Formatierung**.



Gerade im Bereich CSS haben Web-Frameworks enorm zur Reduktion der Komplexität beigetragen. Nicht jeder Software-Entwickler ist gleichzeitig ein guter Web-Designer, und umgekehrt. Mit dem Einsatz von Web-Frameworks wie Bootstrap kann man sich "mehr" auf das wesentliche beschränken und muss nicht ewig viel Zeit mit dem Herumpositionieren von HTML-Elementen verbringen.

Unser HTML-Dokument hat bereits ein paar DIVs, Buttons, eine Überschrift und einen Text. Nun erweitern wir das HTML-Dokument um CSS-Regeln zur Darstellung der Elemente.

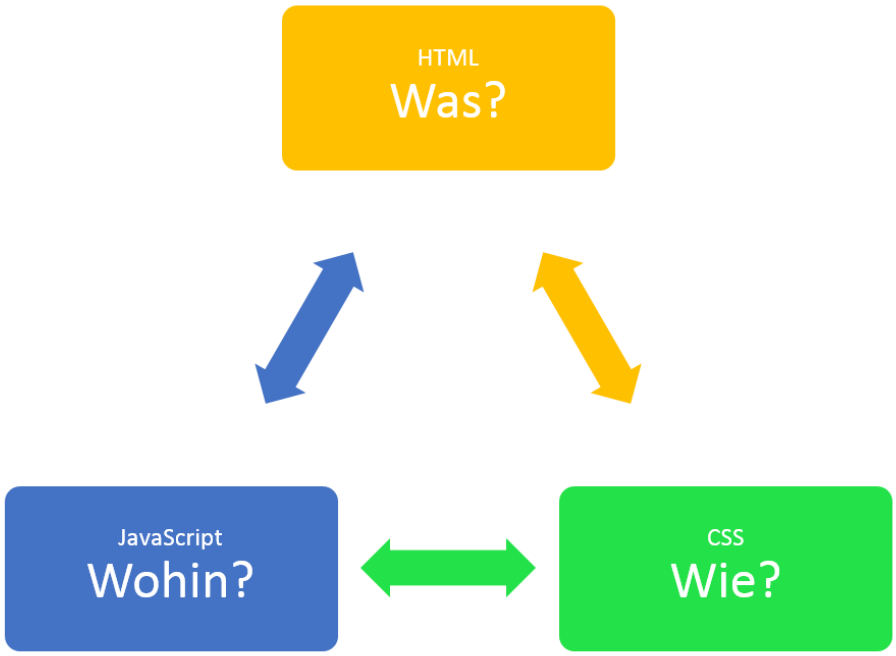
Wir zentrieren die Überschrift und setzen eine Schriftfarbe.

```
1   <!DOCTYPE html>
2   <html lang="de">
3   <head...>
12  <body>
13  <h1 style="text-align: center; color: green;">Hello, world!</h1>
14  <div class="row">
```

Bei CSS-Eigenschaften werden im Wesentlichen die Eigenschaften und Elemente in Bezug auf Aussehen, Position und Sichtbarkeit aufbereitet. CSS bestimmt das Layout und das Design der HTML-Seite.

Javascript > Dynamik & Programmierung

Eine Website, die nur aus HTML und CSS besteht, ist statisch. Es lässt sich nichts verändern. Will man den Inhalt oder das Aussehen von Elementen verändern, kommt Javascript ins Spiel (oder weitere serverseitige Requests). **Wohin sollen sich Elemente bzw. deren Aussehen verändern?**



Mit Javascript kann man sowohl HTML-Eigenschaften als auch CSS-Eigenschaften verändern. Dadurch kann man Einfluss auf den Inhalt, die Struktur und das Aussehen der Seite nehmen. Die Änderung (Manipulation) erfolgt direkt im DOM. Man muss noch definieren, “WANN” das passieren soll.

In unserem Beispiel wollen wir beim Klicken des Buttons alle Buttons, die wir im HTML-Dokument finden können, eine einheitliche Farbe zuweisen. Dazu definieren wir, dass beim Auslösen des “onClick”-Events die Javascript-Funktion „buttonColor“ aufgerufen wird.

```
function buttonColor(color) {
  //Suche im DOM alle Buttons
  var buttons = document.getElementsByTagName('button');
  //iteriere die Buttons
  for (i=0; i<buttons.length; i++) {
    buttons[i].style.color = color;
  }
}
```

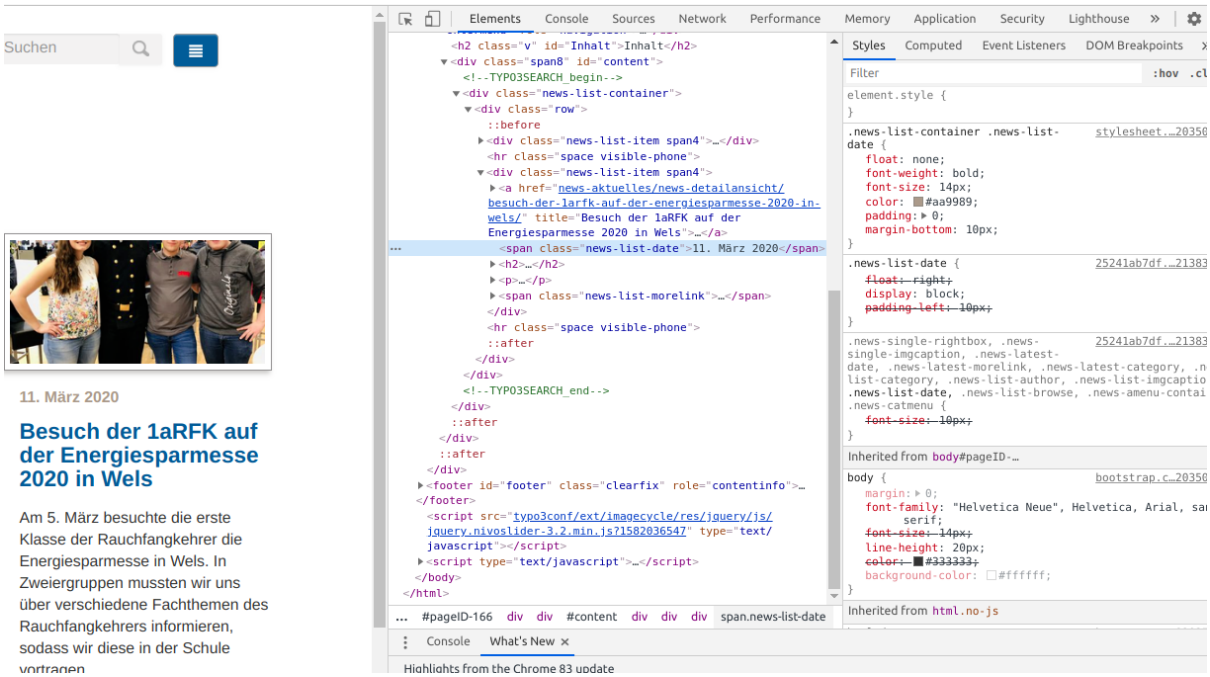
Im Zusammenspiel

Würde der komplette Code so aussehen.

```
<body>
<h1 style="text-align: center; color: green;">Hello, world!</h1>
<div class="row">
    <div class="col-6">
        <p>Dieser Text in einem P-Tag ist ein Blindtext und hat keine Bedeutung. Mittels F12(im Browser kann man üb
    </div>
    <div class="col-6">
        <div class="actions">
            <button>Text ersetzen</button>
            <button onclick="buttonColor('#b3d7ff')">Buttons einfärben</button>
        </div>
    </div>
</div>
<script>
    function buttonColor(color) {
        //Suche im DOM alle Buttons
        var buttons = document.getElementsByTagName('button');
        //iteriere die Buttons
        for (i=0; i<buttons.length; i++) {
            buttons[i].style.color = color;
        }
    }
</script>
```

Der Web-Browser

… hat die Aufgabe Webseiten nach den Definitionen von W3C unter Einhaltung von Sicherheitsrichtlinien darzustellen. Dazu zählt auch die Interpretation von Javascript und CSS. Die bekanntesten Web-Browser Chrome und Mozilla Firefox bieten zur Unterstützung für Web-Entwickler sogenannte Dev-Tools an. Mithilfe dieser Dev-Tools lässt sich der DOM-Baum mit all seinen HTML-Elementen, den gesetzten CSS-Regeln und dem auf der Webseite enthaltenen Javascript-Code nachvollziehen. Die Dev-Tools sind mittels F12-Taste erreichbar. Zusätzlich kann man z. B. HTML-Code und CSS-Regeln verändern und testen, Geräte wie Smartphones und Tablets simulieren, Javascript-Code debuggen, und vieles mehr.



Warum ist es so wichtig, das Zusammenspiel zu verstehen?

Selbst wenn einem heutzutage Frameworks wie Bootstrap einen großen Teil des Codings abnehmen, ist es von Vorteil alle drei Technologien für sich, zu verstehen und auch konzeptionell kombinieren zu können. Immer wieder besteht die Notwendigkeit zumindest Anpassung in Bezug auf Positionierung, Inhalt oder Styling zu machen. Auch für überaus häufig eingesetzte Technologien wie z. B. jQuery ist es notwendig, dass man bestehenden Code versteht bzw. abändern kann.