INT - Internet

IT-Ausbildung
Wintersemester 2022/23

von:
Roman Schmal

Inhaltsverzeichnis

1 Einführung ins Internet

„Das Internet ist für uns alle Neuland! “
Angela Merkel, am 19.06.2013

1.1 Entstehung des Internets

1.1.1 ARPA-Net

Die Defense Advanced Research Projects Agency (ARPA), eine seit 1958 bestehende wissenschaftliche Einrichtung, deren Forschungsergebnisse in militärische Zwecke einflossen, entschloss sich 1966 zur Vernetzung der ARPA-eigenen Großrechner. Dabei wurde die Idee des "dezentralen Netzwerks"wieder aufgegriffen. Ende 1969 waren die ersten vier, drei Jahre später bereits 40 Rechner an das ARPA-Net angeschlossen. Aus ihm sollte später das Internet entwachsen.

→ Idee eines dezentralen Netzwerkes, dass miteinander kommuniziert bzw. Daten austauscht. Falls ein Rechner ausfällt, sind die Daten auf den anderen noch vorhanden.

1.1.2 Wissenschaftliche Einrichtungen

→ Für die Wissenschaft war der Austausch neuer Daten interessant.

Wegen der offenen Architektur des ARPA-Net stand einer solchen Verwendung nichts im Wege. Wissenschaftler konnten von den frühen 70er Jahren an Forschungsergebnisse anderer Institute über das ARPA-Net abrufen oder anderen angeschlossenen Instituten eigene Daten zur Verfügung stellen. Da die angeschlossenen Rechner verschiedene Typen hatten, also Großrechner, Unix-Rechner oder auch PC's, musste ein unabhängiges Datenübertragungsprotokoll für das Netz entwickelt werden.
Das **TCP/IP**-Protokoll wurde dann 1973/74 entwickelt und wurde dann am 01.01.1983 vollständig im Arpanet etabliert. Erst als ein einheitliches Netzwerkstandart durch Netzwerkprotokolle (OSI-Modell) erarbeitet wurde, konnten Verbindungen zwischen Host unterschiedlicher lokaler Netze sichergestellt werden.

1.1.3 Das World Wide Web (WWW)

Der Informatiker Tim Berners-Lee, der in den 80er und 90er Jahren am CERN gearbeitet hat, schrieb ein **Hypertext**-Programm, mit dem man Textdateien editieren konnte. Diese waren in Knoten unterteilt und es gab zu Knoten eine zugehörige Liste mit Links zu anderen Knoten. Man konnte auf jegliche Art von Querbeziehung linken, die man kannte oder fand. Links auf Ziele innerhalb einer Datei wurden vom Enquire-Programm automatisch bidirektional dargestellt – das heißt, auch wenn der Link nur von A nach B gesetzt war, fand man bei B in der Liste den Rückverweis auf A.

→ Hypertext bedeutet in etwa so viel wie Übertext. Gemeint ist jedoch das Hypermedium als das den konkreten Text und seine Medien übergreifende Medium. Hypertext bleibt dabei ein abstraktes Konzept, da es keine konkrete Summe aller involvierten Texte

in Hypertext geben kann.

Im Herbst des Jahres 1990 schrieb Berners-Lee eigenhändig die ersten Versionen der drei Säulen seines Konzepts:

- 1. die Spezifikation für die Kommunikation zwischen Web-Clients und Web-Servern – das so genannte HTTP-Protokoll (HTTP = Hypertext Transfer Protocol)
- 2. die Spezifikation für die Adressierung beliebiger Dateien und Datenquellen im Web und im übrigen Internet – das Schema der sogenannten URIs (URI = Universal Resource Identifier, universeller Quellenbezeichner).
- 3. die Spezifikation einer Auszeichnungssprache für Web-Dokumente, der Berners-Lee den Namen HTML gab (Hypertext Markup Language, Hypertext Auszeichnungssprache).

Berners-Lee schrieb auch die erste Web-Server-Software die unter `info.cern.ch` erreichbar war.

1.1.4 Webbrowser

Wirklich benutzbar wurde das Internet erst durch die Entwicklung von Web-Browsern, mit denen man Webseiten ohne komplizierte Befehle laden und anschauen konnte. Da aber unterschiedliche Unternehmen, wie Microsoft oder Netscape immer wieder neue Features einführten, die die Konkurrenz nicht unterstütze, erarbeiteten das World Wide Web Consortium (W3C) gemeinsame Standards. Da diese aber zu langsam waren, gründeten Apple, Google und Mozilla 2004 die Web Hypertext Application Technology Working Group (WHATWG), die neue Standards unter dem Namen **HTML 5** entwickeln sollten. Derzeit beherrscht Google mit seinem Chrom-Browser den Markt. (vgl. Abbildung 1)

1.2 Aufbau des Internets und Adressierung

Das Internet ist ein weltumspannendes, dezentrales Daten-Netz aus Millionen verbundenen Computern und Computernetzwerken, die miteinander kommunizieren können. Prinzipiell kann jeder Computer, ob Smartphone, Tablet oder PC, ans Internet angeschlossen werden. Um einen Computer mit dem Internet zu verbinden, muss man eine Funk, Daten- oder Telefonverbindung zu einem Internetdienstanbieter (engl. Internet Service Provider) aufbauen. Dieser Provider ist über Datenleitungen mit anderen Providern und Internet-Knoten verbunden, über die man auf Webseiten aus der ganzen Welt zugreifen kann.

Auf z.B. `https://global-internet-map-2021.telegeography.com/` findet sich eine sog. Internetkarte mit den unterschiedlichen Knotenpunkten.

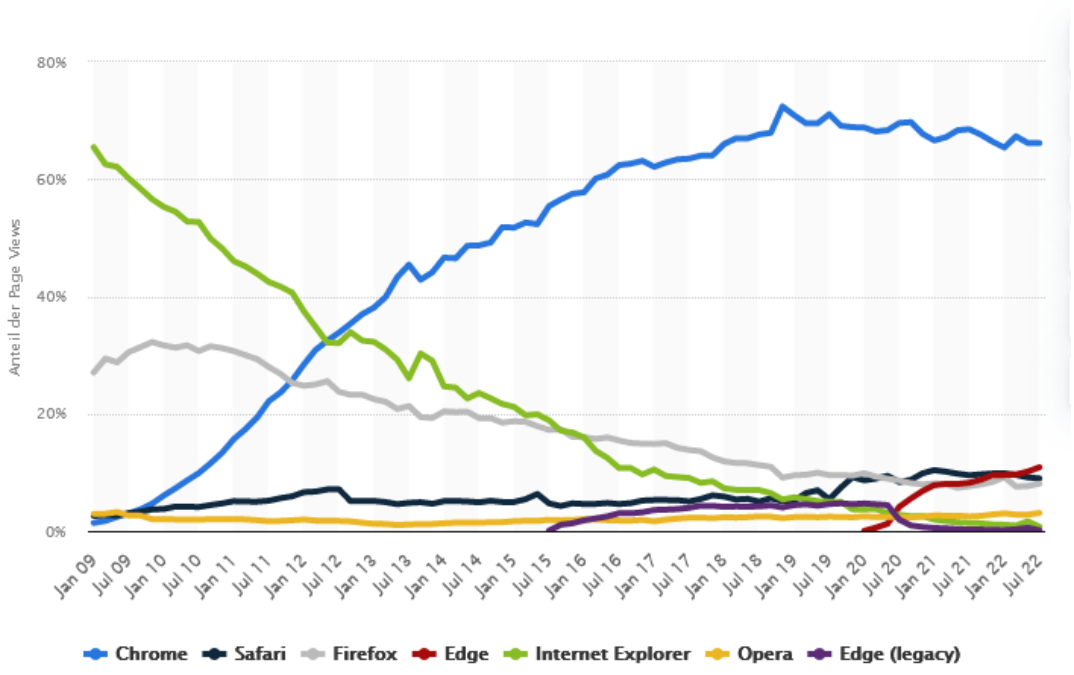

Abbildung 1: Marktanteile der führenden Webbrowsern von 2009 bis 2018.

1.2.1 Internet-Provider

Der Internet Service Provider, den man im Alltag als Internet Provider bezeichnet, ist der Anbieter, der dem Nutzer das Internet bereitstellt. Ein solcher Provider stellt die Verbindungstechnik bereit, zu der in den Augen der Normalnutzer vor allem der Router gehört. Diese Provider kümmern sich aber auch um die Leitungen bzw. um die Funktechnik. Man unterscheidet zwischen Tier-1-, Tier-2- und Tier-3-ISP, wobei die Klassifizierung eine Aussage über die Netzwerkgröße zulässt. Tier-1-Provider sind die lokalen Anbieter, während die Tier-3-ISPs auf der globalen Ebene agieren.

1.2.2 Client-Server-Technologie

Für die einzelnen Internetdienste wie WWW, E-Mail, FTP usw. muss auf einem Host-Rechner, der anderen Rechnern diese Dienste anbieten will, eine entsprechende Server-Software laufen. Ein Host-Rechner kann einen Internet-Dienst nur anbieten, wenn eine entsprechende Server-Software auf dem Rechner aktiv ist, wenn der Rechner önlineïst und wenn keine schützende Software (Firewall) den Zugriff von außen verhindert bzw. einschränkt.

Server sind Programme, die permanent darauf warten, dass eine Anfrage eintrifft, die ihren Dienst betreffen. So wartet etwa ein Web-Server darauf, dass Anfragen eintreffen, die Web-Seiten auf dem Server-Rechner abrufen wollen. Clients sind dagegen Software-Programme, die typischerweise Daten von Servern anfordern. Ihr Browser ist

beispielsweise ein Client. Wenn Sie etwa auf einen Verweis klicken, der zu einer HTTP-Adresse führt, startet der Browser, also der Client, eine Anfrage an den entsprechenden Server auf dem entfernten Host-Rechner. Der Server wertet die Anfrage aus und sendet die gewünschten Daten. Um die Kommunikation zwischen Clients und Servern zu regeln, gibt es entsprechende Protokolle.

→ Internetdienste müssen von Host-Rechnern 'angeboten' werden. Dies geschieht durch Server-Software, die darauf warten bis eine Anfrage durch einen Client gestellt wird. Die Kommunikation zwischen Client und Servern regeln entsprechende Protokolle:

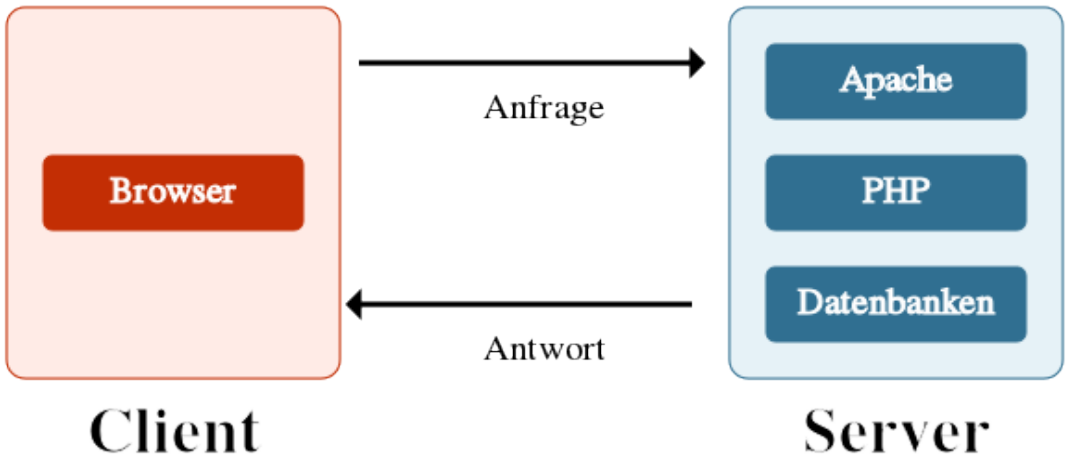

Abbildung 2: Client-Server-Modell

1.3 Protokolle

Protokolle wurden entwickelt, damit es einheitliche Verfahrens- und Ablaufvorschriften gibt. Dies gewährleistet die Kommunikation zwischen unterschiedlichen Computern und Netzwerksystemen.

1.3.1 Internet Protocol (IP)

Jeder Rechner im Internet besitzt eine IP-Adresse und ist dadurch eindeutig identifizierbar. Eine solche Adresse besteht bei IPV4 aus 32 Bit, die in vier durch Punkte getrennte Zahlen zwischen 0 und 255 dargestellt werden (gepunktete Dezimalform), oder bei IPV6 durch acht Vierergruppen von hexadezimalen Digits, also $8 \times 16 = 128$ Bit. Die einzelnen Gruppen werden hier durch Doppelpunkte getrennt.

1.3.2 Domain Name System (DNS)

Ein Domainname besteht aus mehreren, durch Punkt getrennten Labels. Er wird immer von rechts nach links gelesen und ausgewertet. Am wichtigsten ist die Top Level Domain, ein Buchstabenkürzel, das das Land (oder eine weltweite Organisation) kennzeichnet.

Host/Dienst	Second-Level-Domain	Top-Level-Domain
www	youtube	de

→ Es gibt in Inhaltsverzeichnis um das Ziel von hinten aufzuschlüsseln. Dieses wird auf Root-Servern gespeichert, von denen es 13 auf der Welt gibt.

Iterative-Suche: Bei einer Anfrage wird nun das Inhaltsverzeichnis nach dem zuständigen Nameserver für die TLD gefunden. Dieses ermitteln nun den zuständigen SLD-NS.

Rekursive Suche: Bei einer Anfrage wird der nächste erreichbare NS angefragt, des Internet-Anbieters. Falls dieser keinen Eintrag hat, fragt er den NS, der für Ihn zuständig ist, das geht dann bis zum Root-Server und dann startet die iterative.

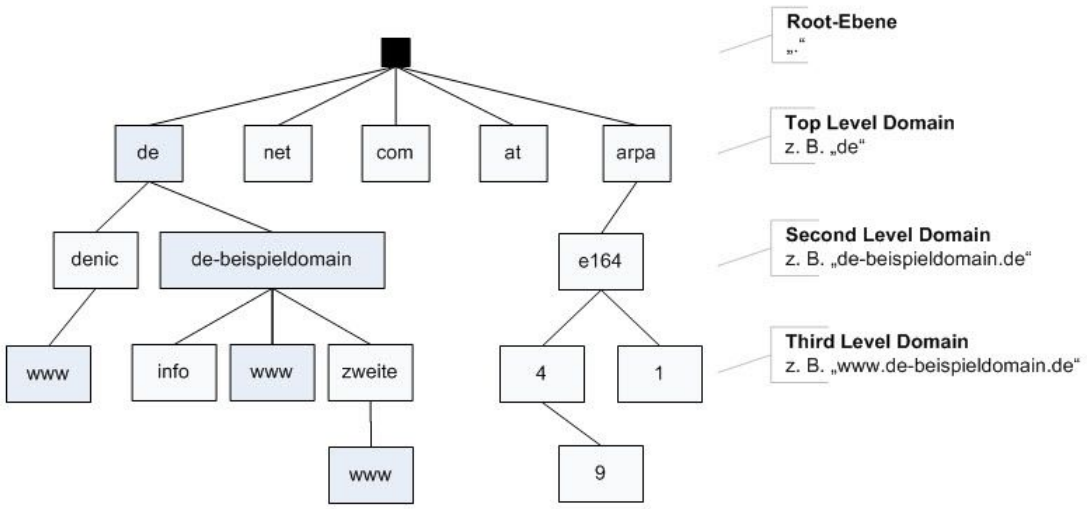

Abbildung 3: Schematischer aufbau einer Namensauflösung.

1.3.3 Uniform Resource Locator (URL)

Auf Deutsch: einheitlicher Ressourcenzeiger. Im allgemeinen Sprachgebrauch auch als Internetadressen oder Webadressen bezeichnet.

→ identifiziert und lokalisiert eine Ressource, z.B. eine Webseite oder Datei.
`https://www.youtube.com/watch?v=dQw4w9WgXcQ`

- Übertragungsprotokoll: https:
- Adresse: www.youtube.com
- evtl. Pfad: watch?v=dQw4w9WgXcQ

1.3.4 TCP/IP

Da Texte, Bilder und Videos oft aus zu großen Dateien bestehen, werden sie für den Transport mit dem TCP/IP-Protokoll in Datenpakete zerlegt und dann verschickt. Dabei nehmen die Pakete nicht den direkten Weg, sondern werden je nach Verfügbarkeit zwischen den einzelnen Internet-Knoten hin und her geroutet (deshalb der engl. Begriff: Router). So läuft ein Großteil „innerdeutscher“ E-Mails und Seiten-, bzw. Suchanfragen über Router, die im Ausland stehen.

Damit die Datenpakete wieder auffindbar sind, bestehen sie immer aus zwei Teilen, dem Header und der Nutzlast (payload). Die Nutzlast enthält die zu übertragenden Daten, die wiederum Protokollinformationen der Anwendungsschicht wie HTTP oder FTP entsprechen können. Der Header enthält für die Kommunikation erforderliche Daten sowie die Dateiformat-beschreibende Information.

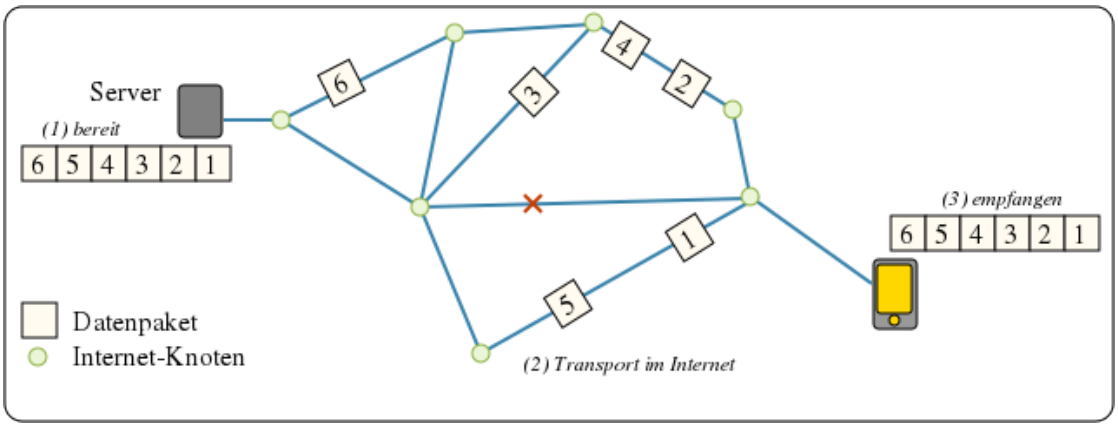

Abbildung 4: Beispiel-schema für ein Datentransport durch das Internet, mit Hilfe des TCP/IP Protokolls.

→ Daten werden in kleinere Pakete getrennt und werden bis zum Zielort hin und her geroutet. Die Datenpakete bestehen aus dem **Header**, der die Informationen für die Kommunikation beinhaltet und der **Nutzlast**, die wiederum die übertragenden Daten hat.

1.4 Organisationen des Internets

1.4.1 W3C (World Wide Web Consortium):

Diese Organisation überwacht die Entwicklung von Standards im World Wide Web und dokumentiert diese. Das W3C arbeitet mit der ISOC zusammen, ist ihr aber nicht untergeordnet.

1.4.2 IGF (Internet Governance Forum):

Dieses Beratungsgremium wurde von der UN-Organisation für wirtschaftliche und soziale Angelegenheiten (UN-DESA) 2004 ins Leben gerufen. Das IGF berät die Gremien der ISOC und versucht, den Einfluss der Vereinten Nationen auf das Internet sicherzustellen.

1.4.3 ICANN (Internet Coroporation for Assigned Names and Numbers):

Diese Non-Profit-Organisation koordiniert die Vergabe von einmaligen Namen und Adressen im Internet. Dazu gehört die Koordination des Domain Name Systems und die Zuteilung von IP-Adressen. Die ICANN hat ihren Hauptsitz in Los Angeles und wurde 1998 gegründet.

1.4.4 IANA (Internet Assigned Numbers Administration):

Die Vergabe von Internet-Adressen unterstand bis 1998 einem einzelnen Mann: Jon Postel. Nach dem Tode des Informatikers wurde die Kontrolle über das Domain Name System an die ICANN übergeben.

1.4.5 IRTF (Internet Research Task Force):

Diese Gruppe widmet sich der Forschung an neuen Internet-Technologien. Sie wurde 1986 gegründet, um die Forschung und Entwicklung im Bereich der Netzwerke und deren Techniken zu fördern. Ihre Arbeit wird koordiniert von der IRSG (Internet Research Steering Group).

1.4.6 IETF (Internet Engineering Task Force):

Diese Organisation ist eine Vereinigung von Informatikern und Fachleuten, die Standards wie das Internet Protocol und Übertragungsstandards wie TCP oder UDP entwickeln und dokumentieren. Seine Arbeit wird gesteuert von der IESG (Internet Engineering Steering Group).

1.4.7 IAB (Internet Architecture Board):

Das IAB ist ein Komitee, das den Überblick über die Arbeit der einzelnen ISOC-Unterorganisationen hält und mit dem RFC Editor die Dokumentation von Internet-Standards organisiert. → RFC: Request for Comments - Dokumentation.

1.4.8 ISOC (Internet Society):

Die Nichtregierungsorganisation mit Sitz in Genf und Reston/USA überwacht die Entwicklung der Ressourcen und Standards des Internets. Die ISOC wurde 1994 gegründet und hat etwa 6.000 Mitglieder.

2 Grundaufbau und Inhalte von Web-Seiten

Während Webseiten immer komplexer werden, bleiben die zugrunde liegenden Strukturen bemerkenswert einfach. Die Struktur für eine Webseite wird durch HTML vorgegeben. Die Browser wiederum stellen die in HTML gefassten Inhalte für die Anwender dar. Eine Webseite besteht primär aus drei Komponenten:

- Textinhalt : der reine Text, der auf der Seite erscheint, um die Besucher über Ihre Firma, Familienferien, Produkte – oder was sonst der Schwerpunkt Ihrer Seite ist – zu informieren.
- Verweise auf andere Dateien : Diese Links laden Objekte wie Bilder oder Audio-, Video- und SVG-Dateien und verknüpfen mit anderen HTML-Seiten oder weiteren relevanten Inhalten und auch Stylesheets (Formatvorlagen, die das Layout der Seite steuern) und JavaScript- Dateien, die die Seite um bestimmte Effekte ergänzen.
- Markup: Diese HTML-Elemente (auch Auszeichnungen genannt) beschreiben Ihren Textinhalt und lassen die Querverweise funktionieren (das M in HTML steht für Markup , also Auszeichnung).

Bemerkenswerterweise bestehen alle diese Komponenten einer Webseite ausschließlich aus Text. Diese wesentliche Eigenschaft bedeutet, dass Webseiten in reinem Textformat gespeichert werden und praktisch mit jedem Browser auf jeder beliebigen Plattform (egal ob Desktop, Tablet-PC, Smartphone etc.) betrachtet werden können. Das garantiert die Universalität des Webs. Eine Seite sieht vielleicht etwas anders aus, wenn man sie auf zwei Geräten nebeneinander aufruft und vergleicht, aber das macht nichts. Wichtig ist hier als erster Schritt, dass die Inhalte für alle User zugänglich sind, und HTML leistet das. Neben den drei Komponenten, aus denen eine Webseite hauptsächlich besteht, enthält sie auch HTML mit Informationen über die Seite selbst. Das Meiste davon werden Ihre User nicht explizit zu Gesicht bekommen, sondern es ist prinzipiell für Browser und Suchmaschinen gedacht. Das können beispielsweise Informationen über die primäre Sprache der Inhalte (Deutsch, Englisch usw.), die Zeichenkodierung (üblicherweise UTF-8) und anderes mehr sein.

2.0.1 Semantische Struktur

HTML5 betont die HTML-Semantik und überlässt die gesamte visuelle Gestaltung dem CSS. Das war bei früheren Versionen von HTML nicht immer der Fall. Als das Web entstand, gab es noch eine korrekte Weise, um Seiten zu formatieren. HTML gab es schon ein paar Jahre, als im Dezember 1996 CSS1 formell eingeführt wurde. Um diese Lücke zwischenzeitlich zu füllen, wurden bei HTML eine Handvoll Darstellungselemente aufgenommen, deren Zweck es war, eine einfache Formatierung des Texts zu erlauben, also ihn z.B. fett, kursiv oder in anderer Größe als die umgebende Schrift darzustellen.

Diese Elemente waren damals für ihre Zwecke ganz dienlich, aber während sich die

empfohlenen Vorgehensweisen im Web weiterentwickelten, fielen sie berechtigterweise in Ungnade. Im Kern dieser Denkweise stand (und steht immer noch) das Konzept, dass HTML nur für die Beschreibung der Bedeutung der Inhalte gedacht ist und nicht deren Darstellung. Die darstellungsbezogenen HTML-Elemente durchbrachen diese bewährte Vorgehensweise. Somit bezeichnete man deren Verwendung in HTML 4 als veraltet und empfahl den Autoren, Text und andere Seitenelemente mit CSS zu formatieren.

HTML5 geht noch weiter: Es eliminiert bestimmte darstellungsbezogene Elemente und definiert andere neu, damit sie nur einen semantischen Wert enthalten und nicht die Darstellung diktieren. Das Element small ist ein solches Beispiel. Anfänglich war es dazu gedacht, Text kleiner als Normaltext zu machen. Doch in HTML5 steht **small** für das „Kleingedruckte“, z.B. rechtliche Hinweise. Sie können diesem Text mit CSS die größte Schrift auf der Seite geben, wenn Sie wollen, aber das ändert die inhaltliche Bedeutung von small nicht. Mittlerweile existiert der alte Gegenpart von small, das Element big, in HTML5 nicht mehr. Es gibt noch weitere solche Beispiele, von denen Sie im Verlauf dieses Buchs erfahren. HTML5 definiert auch neue Elemente wie **header, footer, nav, article, section** und viele andere mehr, die die Semantik des Inhalts bereichern.

Doch egal, ob Sie nun ein HTML-Element verwenden, das seit Anbeginn der Sprache existiert, oder eines, das durch HTML5 neu hinzukam, Ihr Ziel sollte das Gleiche bleiben: Wählen Sie die Elemente, die die Bedeutung Ihres Inhalts am besten beschreiben, ungeachtet ihrer Darstellung.

2.0.2 Warum diese Struktur

- Verbesserte Zugänglichkeit und Interoperabilität (die Inhalte stehen für Besucher mit bestimmten Einschränkungen oder Behinderungen, die anhand von assistiver Technologie darauf zugreifen, und für Browser auf Desktop, Handy, Tablet-PC und anderen Geräten gleichermaßen zur Verfügung)
- Verbessere Suchmaschinenoptimierung (Search Engine Optimization, SEO)
- (Üblicherweise) leichterer Code und schnellerer Seitenaufbau
- Leichtere Formatierung und Codepflege

Accessibility: Allen Nutzern zur Verfügung stellen!

„Die Macht des Internets liegt in seiner Universalität. Es kann sein volles Potenzial erst dann entfalten, wenn jeder Zugriff hat und eine Behinderung keine Rolle spielt “- Tim Berners-Lee

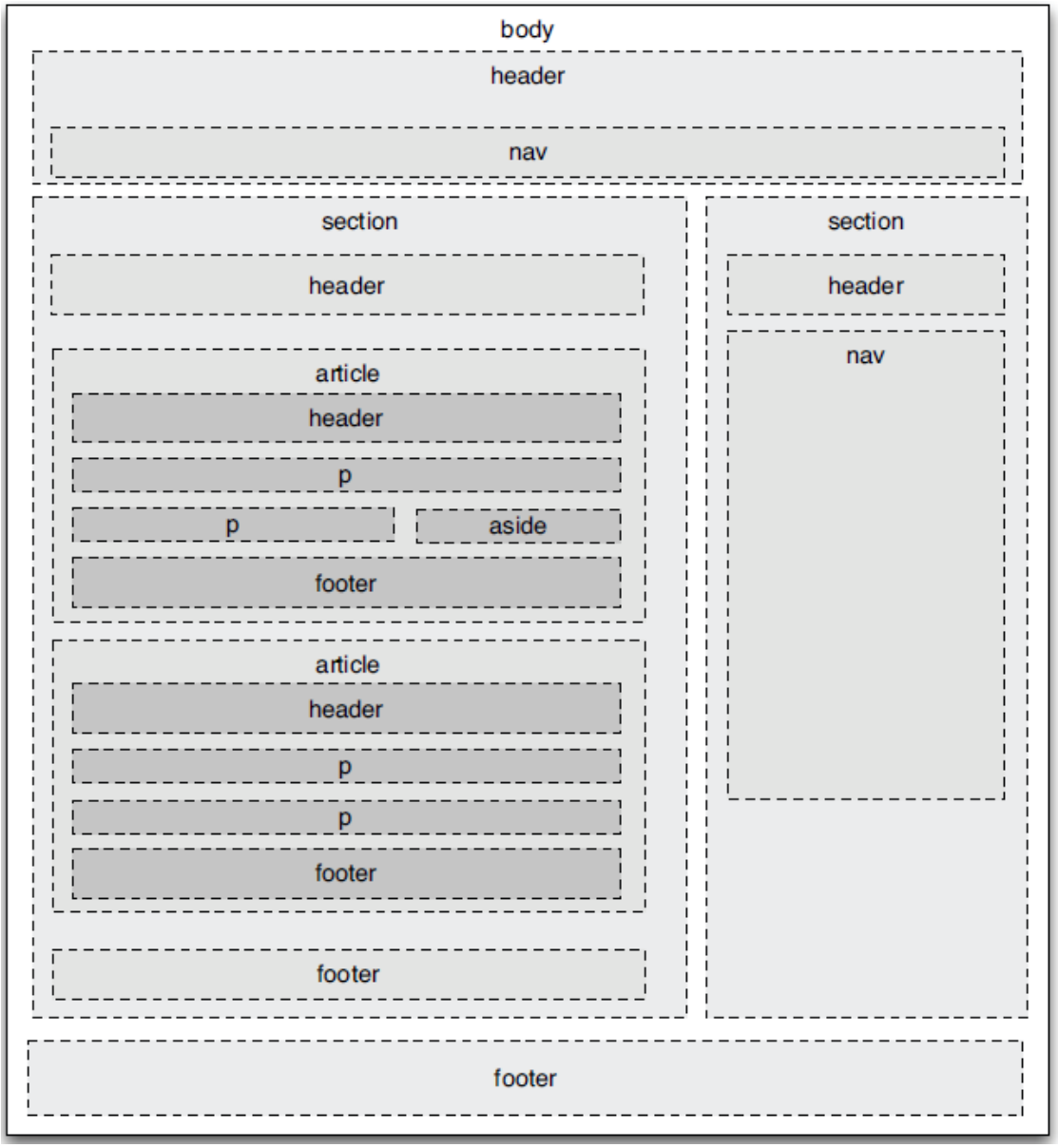

Abbildung 5: Beispielaufbau einer Blog-Seite mit Html5 Elementen.

3 Suchmaschinen

Die Suchmaschinen haben das Internet erst wirklich für alle nutzbar gemacht. Ohne Suchmaschinen bräuchte man eine Art Telefonbuch in dem die Internetseiten aufgrund ihres Inhalts gelistet wären. Die erste Suchmaschine Namens “Archie “startete bereits im November 1990. Dieses Programm durchsuchte Datei- und Ordnernamen in FTP-Verzeichnissen. Die Basis heutiger Suchmaschinen bildetet aber der “World Wide Web Wanderer “, nämlich einen Searchbot, der von Link zu Link Springt und so das Internet *vermessen* sollte. Hier widmen wir uns eigentlich nur der Suchmaschine von Google, da diese einen Marktanteil von ca. 95 % hat. Die Basisfunktionen sollten aber bei anderen Suchmaschinen ähnlich, wenn nicht gleich sein.

3.1 Funktionsweise

Eine Suchmaschine ist ein Programm, dass nach Inhalten sucht. Eine Optimale Suchmaschine würde genau das finden was wir meinen zu Suchen. Die Suchanfrage sollte deshalb detailliert gestellt werden. Die Kommunikation mit Suchmaschinen funktioniert über Syntax, mit Hilfe von Keywords, die man eingibt. Damit die relevanten Informationen für den Endbenutzer sichtbar sind, müssen diese zuerst in einer Art Liste gespeichert werden. Die Informationsbeschaffung erfolgt nicht in Echtzeit sondern läuft non-stop im Hintergrund ab. Diese Daten werden dann Verarbeitet und Anhand der Relevanz sortiert.

Abbildung 6: Prinzip von Suchmaschinen-Algorithmus. `https://www.semotion.de/wie-funktionieren-suchmaschinen/`

- **Crawling:** Der Crawler (auch Spider oder Bot) durchsucht das Internet nach Inhalten, analysiert ihre Verbindungen und folgt Links auf bereits indexierte Seiten. Letzteres wird als sogenanntes “Harvesting “bezeichnet. Primär sind HTML-Seiten das Ziel eines Crawlers, wobei auch .ppt oder .pdf-Dateien erfasst werden können. Zudem analysiert er die **Meta-Tags** der Website. Nachdem eine Internetpräsenz gecrawlt wurde, speichert er die erfassten Informationen auf einem Server ab. Bereits vorhandene Websites werden in unterschiedlichen Abständen besucht, um so mögliche Veränderungen zu registrieren. Dieser vollautomatisierte Prozess verläuft komplett im Hintergrund. Der normale Endnutzer registriert diesen Arbeitsvorgang nicht.

Circa 7% des gesamten Netzverkehrs wird inzwischen durch Robots verursacht.

- **Index:** Der Indexer verarbeitet die vom Crawler gefundenen Daten und strukturiert den Prozess. Ein Bot ordnet die gecrawlten Seiten in ein Verzeichnis, überschreibt ggf. alte Daten und indexiert die Inhalte, damit sie fortan schnell und einfach zu durchsuchen sind. In einem Rechenprozess wertet das System die gesammelten Daten aus, um eine Relevanz der Homepage zu einem Schlüsselwort festzulegen. Wenn ein Nutzer anschließend eine Suchanfrage startet, greift die Suchmaschine auf das Verzeichnis zurück und analysiert, auf welcher Website der Suchbegriff vorhanden ist. Somit muss bei der Suchanfrage nicht mehr das gesamte World Wide Web durchforstet werden, sondern ausschließlich der angelegte Index. Die Gewichtung erfolgt durch das sogenannte IR-System (Information Retrieval).

  Anschließend werden aus dem Repository (deutsch: Zwischenspeicher, Lager) passende Informationen geladen. Dazu gehören unter anderem der Seitentitel und die Seitenbeschreibung. Daraufhin werden die Ergebnisse in den sogenannten SERPs dargestellt. Unter SERP (Search Engine Result Pages) wird die Ergebnisseite verstanden, welche der Nutzer nach der Suchanfrage angezeigt bekommt.

- **Ranking:** Das Suchmaschinen Ranking ist für Webseiten-Betreiber ein zentrales Element. Eine bessere Platzierung bedeutet auch mehr Besucher auf der Homepage, eine größere Reichweite und im Zweifelsfall auch mehr potentielle Kunden für Unternehmen.Deshalb ist die Suchmaschinenoptimierung (kurz SEO) mittlerweile auch ein zentraler Bestandteil des Online Marketings von Unternehmen.

  Welches Ergebnis an erster Stelle in den SERPs steht, hängt stark von der jeweiligen Suchmaschine ab. Jeder Betreiber nutzt andere Mechanismen und Algorithmen. Sie sorgen dafür, dass für den Nutzer die Treffer angezeigt werden, welche die Suchanfrage am besten beantworten. Die Faktoren von Google & Co. setzen sich aus über zweihundert Komponenten zusammen, die darüber entscheiden, welche Suchmaschinen Platzierung Ihre Homepage in den Ergebnissen erreicht. Der Bereich der Suchmaschinenwerbung ist davon ausgeschlossen, da hier andere Faktoren gelten.

  Wichtig für das Ranking ist hierbei:

  - Die Inhalte der Webseite du deren Semantik
  - Vorhandene Verlinkungen, sowohl extern als auch intern
  - Aktualität
  - Informationsarchitektur der Seite
  - Einwandfreie technische Funktion

3.2 Optimierung der Websuche

Da Suchmaschinen Programme sind, können diese auch spezielle Befehle verstehen um das Programm optimal zu nutzen. Hier ein paar Tipps zum Umgang mit Google:

- 1: Mehrere Wörter aneinander reihen, um z.B. den Ort oder Öffnungszeiten zu spezifizieren. **kino regensburg öffnungszeite**
- 2: Wörter gezielt auswählen, heißt: anstatt **mein Kopf tut weh** , **Kopfschmerzen** eingeben.
- 3: Groß- und Kleinschreibung kann ignoriert werden und in den meisten Fällen auch die Rechtschreibung.

Google funktioniert auch als Rechner für Einheiten (**3 dollar in euro**) als auch für mathematische Operationen(**3*9123**). Außerdem kann man mit dem Keyword **define:** ohne Leerzeichen, vor einem Wort sich die Definition anzeigen lassen.

3.2.1 Suchtoperatoren

Die Suchmaschine von Google verwendet auch Suchoperatoren, mit denen man z.B. den Suchradius verkleinern kann.

- Soziale Netzwerke: Fügen Sie ein @-Zeichen vor einem Wort hinzu, um in sozialen Netzwerken zu suchen. Beispiel: @twitter
- Preis: Fügen Sie ein €-Zeichen vor einer Zahl hinzu. Beispiel: kamera €400
- Hashtags: Fügen Sie ein #-Zeichen vor einem Wort hinzu. Beispiel: #throwbackthursday
- Ausschluss: Fügen Sie einen Bindestrich (-) vor einem Wort hinzu, das Sie ausschließen möchten. Beispiel: jaguar geschwindigkeit -auto
- Passend: Setzen Sie ein Wort oder eine Wortgruppe in Anführungszeichen. Beispiel: “größtes gebäude“
- Bereich: Fügen Sie .. zwischen zwei Zahlen ein. Beispiel: kamera €50..€100
- Kombinieren: Fügen Sie OR (das englische Wort öder") zwischen verschiedenen Suchanfragen ein. Beispiel: marathon OR rennen
- Websitesuche: Fügen Sie site: vor einer Website oder einer Domain hinzu. Beispiel: site:youtube.com oder site:.gov
- Ähnlichkeit: Fügen Sie related: vor einer Webadresse hinzu, die Sie bereits kennen. Beispiel: related:spiegel.de
- Cache-Version: Fügen Sie cache: vor der Webadresse hinzu.

Außerdem kann man die Filter setzten, um z.B. die Sprache, oder die Zeit einstellen. Unter `https://www.google.com/advanced_search` finden sich auch zahlreiche Einstellmöglichkeiten um eine detaillierte Suche aufzugeben.

4 Sicherheit im Internet

4.1 SSL - Secure Socket Layer

SSL steht für“Secure Socket Layer “und bezeichnet ein Verschlüsselungsprotokoll. Streng genommen ist SSL die Vorgängerversion von TLS (Transport Layer Security), jedoch wird die Bezeichnung SSL für beide Versionen verwendet. TLS ist eine modifizierte Variante der letzten Version von SSL, mit der einige kritische Schwachstellen von SSL beseitigt wurden. Der Einsatz eines Verschlüsselungsprotokolls sorgt dafür, dass Datenübertragungen verschlüsselt und somit sicherer sind. Nur eine Webseite mit einem Verschlüsselungsprotokoll kann als https angezeigt und somit auch im Google-Ranking höher gewertet werden.

4.1.1 Passendes Zertifikat

Grundsätzlich stehen drei verschiedene SSL-Zertifikate zur Verfügung:

- 1. Extended Validation (EV) - höchste Verschlüsselungsstufe: Um dieses Zertifikat zu erhalten, wird von den ausgebenden Stellen eine große Zahl an Informationen abgefragt. Die Kriterien gelten als die strengsten, die für den Erhalt der SSL-Verschlüsselung erfüllt werden müssen. Dabei wird nicht nur eine einzelne Seite zertifiziert, sondern das gesamte Unternehmen.
- 2. Organisation Validated (OV) - mittlere Verschlüsselungsstufe: Auch diese SSL-Zertifikate beinhalten eine Authentifizierung Deines Unternehmens. Für den Erhalt des Zertifikats prüft das jeweilige Unternehmen einige Daten, die Du bereitstellst. Deine Informationen werden aber nicht so stark hervorgehoben wie beim umfangreicheren EV-Zertificate. Wollen Besucher diese Daten sehen, müssen sie die einzelnen Details separat abrufen.
- 3. Domain Validated (DV) - niedrigste Stufe der Verschlüsselung: Ein DV-Zertifikat verschlüsselt Deine Website ebenfalls per SSL. Doch tatsächlich sind im Zertifikat deutlich weniger Daten zu Dir und Deinem Unternehmen enthalten. Das DV-Zertifikat ist lediglich eine Validierung dafür, dass Du der Inhaber der Website bist und die Seite aktiv verwaltest.

Da solche Zertifizierungsstellen fast immer mit monatlichen oder jährlichen Kosten verbunden sind, gibt es mittlerweile `https://letsencrypt.org/`. Diese Zertifizierungsstelle wurde unter der Leitung von Chrome, Facebook, Mozilla und einigen anderen Unternehmen gegründet und man kann sich ein kostenloses Zertifikat für nicht kommerzielle Zwecke ausstellen lassen.

4.2 Sichere Nutzung des Internets

Da die größte Sicherheitslücke immer noch der Nutzer ist, gibt es zahlreiche Tipps für die Nutzung des Internets. Hier ein paar Beispiele vom BSI:

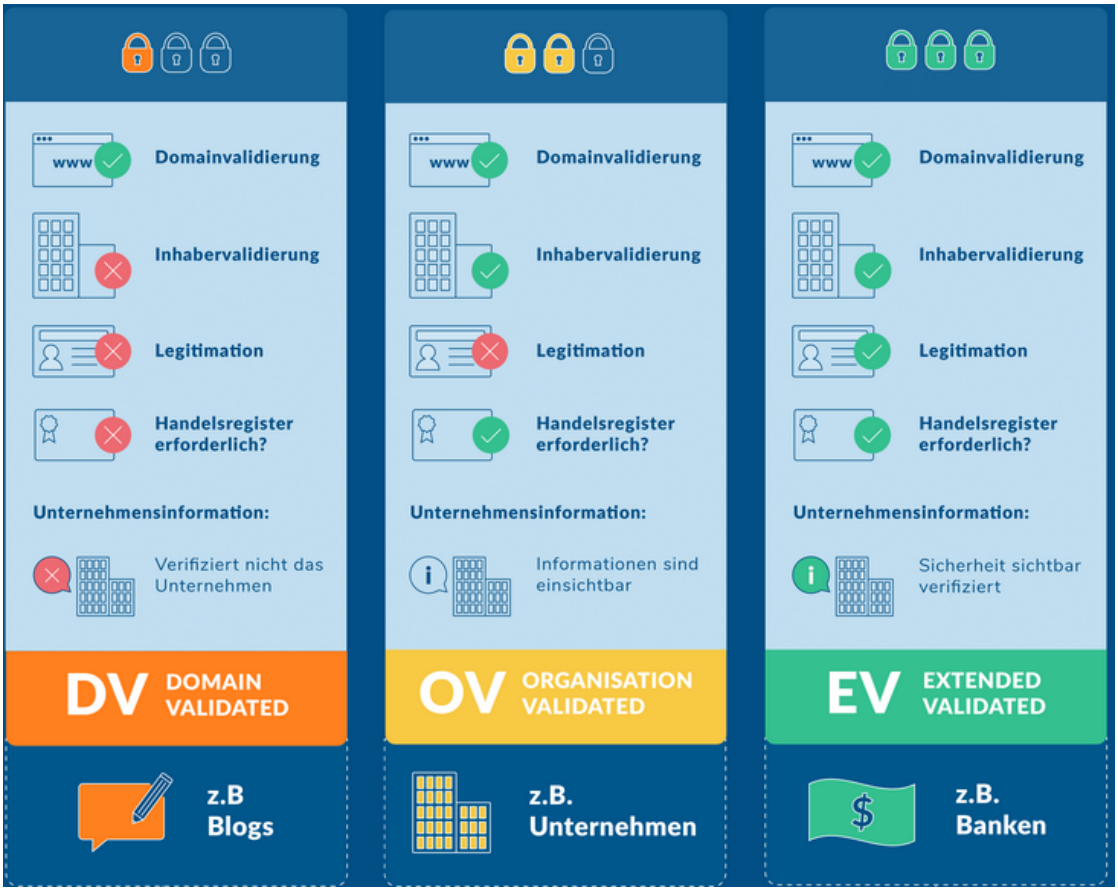

Abbildung 7: Mögliche SSL-Zertifikate.

- Passen Sie Ihren Webbrowser an und halten Sie ihn aktuell: Bei Erweiterungen, Add-ons oder auch Plug-ins handelt es sich um kleine Programme, die Ihren Browser mit zusätzlichen Funktionen ausstatten können. Deaktivieren oder deinstallieren Sie die Programme, die Sie nicht unbedingt benötigen.
- Halten Sie Ihr Betriebssystem und andere Software aktuell: Deinstallieren Sie Programme, die sie nicht länger nutzen. Je weniger Anwendungen installiert sind, desto kleiner ist die Angriffsfläche Ihres gesamten Systems.
- Schützen Sie Ihre Online- und Benutzerkonten mit sicheren Passwörtern - Die für ein sicheres Passwort sind:
  - Sie müssen sich ein Passwort gut merken können.
  - Je länger das Passwort ist, desto besser.
  - Das Passwort sollte mindestens acht (dreizehn) Zeichen lang sein.
  - Für ein Passwort können in der Regel alle verfügbaren Zeichen genutzt werden, also Groß- und Kleinbuchstaben, Ziffern und Sonderzeichen.

- Das vollständige Passwort sollte nicht im Wörterbuch vorkommen. Gängige Zahlenfolgen oder Tastaturmuster kommen ebenfalls als sicheres Passwort nicht in Frage.
- Einfache Ziffern oder Sonderzeichen vor oder nach einem normalen Wort zu ergänzen, ist nicht empfehlenswert.
- Seien Sie vorsichtig bei E-Mails und deren Anhängen: Verzichten Sie, wenn möglich, auf die Darstellung und Erstellung von E-Mails im HTML-Format und verwenden Sie stattdessen ein reines Textformat. Die Nutzung des HTML-Formats können Sie über die Einstellungen Ihres Mailprogramms ändern. Seien Sie vorsichtig beim Öffnen von E-Mail-Anhängen oder beim Klick auf einen Link, denn Schadprogramme werden oft über in E-Mails integrierte Bilder oder Dateianhänge verbreitet oder verbergen sich hinter Links.
- Seien Sie vorsichtig bei Downloads, insbesondere von Programmen: Meiden Sie Quellen, bei denen Sie Zweifel an der Seriosität haben. Vergewissern Sie sich vor dem Download von Programmen, ob die Quelle vertrauenswürdig ist.
- Seien Sie zurückhaltend mit der Weitergabe persönlicher Daten: Kriminelle im Internet steigern ihre Erfolgsraten, indem sie ihre Opfer individuell ansprechen: Zuvor ausspionierte Daten, wie etwa Surfgewohnheiten oder Namen aus dem persönlichen Umfeld, werden dazu genutzt, Vertrauen zu erwecken.

4.3 Sicherheitszonen

Unter Windows kann man sog. Sicherheitszonen einrichten und auch einzelne Webseiten zu diesen Zonen hinzufügen. Internet Explorer weist alle Websites einer von vier Sicherheitszonen zu: “Internet“, “Lokales Intranet “, “Vertrauenswürdige Sites “oder “Eingeschränkte Sites “. Die Sicherheitseinstellungen für eine Website werden anhand der Zone festgelegt, der die Website zugewiesen ist.

Um Sicherheitszonen einzurichten gehen Sie folgendermaßen vor:

- Geben Sie “Internetoptionen “in das Windows-suchfeld ein oder navigieren Sie zu Systemsteuerung → Netzwerk und Freigabecenter → Internetoptionen.
- Unter dem Reiter Sicherheit kann man nun eine Sicherheitszone auswählen.
- Nun kann man z.B. bei der Sicherheitszone “Vetrauenswürdig “eine Webseite unter “Sites “hinzufügen.

5 HTML 5

HTML steht für 'Hypertext Markup Language', auf Deutsch Auszeichnungssprache für Hypertext. Hyper wird hier im Sinne von „erweitert“ gebraucht und bedeutet, dass Hypertext nicht einfach zur geradlinigen Lektüre gedacht ist. Statt dessen bietet ein Hypertext-Dokument Verknüpfungen zu anderen Textstellen an - die Hyperlinks. Das Ziel solcher Links kann im gleichen Dokument liegen oder auch in einem ganz anderen. Der Leser kann damit jederzeit in seinem Lesefluss an eine andere Stelle verzweigen.

Hypertext lässt sich auf unterschiedliche Weisen herstellen. HTML ist eine rein auf Text basierende Realisierung dieses Konzepts und verwendet spezielle Zeichenfolgen, um Hyperlinks zu erzeugen und Einsprungpunkte für Hyperlinks zu markieren. Gleichzeitig bieten diese Markierungen auch die Möglichkeit, den Inhalt des Dokuments zu strukturieren, so dass bei der Anzeige des Dokuments eine passende Darstellung erzeugt werden kann.

Diese Markierungen benötigen kein spezielles Programm, um sie zu erstellen, sondern können genau wie der übrige Text mit einem einfachen Texteditor geschrieben werden. Man nennt sie Tags (engl. für Marken), und sie geben an, wo bestimmte Teile eines Dokuments beginnen und enden. Ein solches Tag besteht am Beginn eines Dokumentteiles aus einem Namen, der in spitze Klammern eingeschlossen ist. Für das Ende fügt man vor dem Namen noch einen Schrägstrich hinzu. Eins dieser Tags heißt **h1** und markiert die Haupt-Überschrift einer Seite.

5.1 Grundgerüst

Ein valides HTML5-Dokument, also eine Webseite, die dem vom W3C vorgegebenen Standard entspricht, besteht mindestens aus der Angabe des Dokumententyps und aus folgenden Elementen:

- html → Wurzelelement umschließt das gesamte Dokument.
- head → Informationen über die Seite, für Browser und Suchmaschinen.
- title → Wird im Browser-Tab angezeigt.
- body → Der Hauptteil der gezeigten Webseite.

Der DOCTYPE gibt hier an, das wir den HTML 5 Standard nutzen. Der **<p>**-tag ist ein Absatz und der **<h>**-tag (h1-h6) ist für Überschriften.

5.1.1 Jede Menge Elemente und Attribute

Da es viele Elemente, die auch einige Attribute besitzen, gibt. Empfiehlt es sich bei Unklarheiten in dem HTML 5 Standard nach zu schauen. `https://html.spec.whatwg.org/#toc-semantics`

```html
<!doctype html>
    <html lang="de">
    <head>
        <meta charset="utf-8">
        <title>Beschreibung der Seite</title>
    </head>
    <body>
        <h1> Das ist eine Überschrift</h1>
        <p>Inhalt der Seite</p>
        <!-- Ein Kommentar-->
        <p> Noch mehr Inhalt </p>
    </body>
</html>

```

Abbildung 8: Das Grundgerüst einer HTML 5 Datei.

Das <a> Element hat mehrere Funktionen, abhängig vom Kontext. Meist wird es im Interaktiven Kontext benutzt, da mit dem **href** Attribut es auf eine Adresse referenziert.

Für die Verlinkung auf lokale Dateien mit z.B. <a href ="main.html» Main verwendet man den relativen Pfad, das heißt den Pfad von der Datei aus, von der aus der Link folgt. Bei dem Verweis auf einen Order benutzt man z.B „Bilder\bilder.html “. Um einen Ordner zurückzuspringen benutzt man „..\main.html “. Die Benutzung von absoluten Pfaden erfolgen mit / wie z.B „/Bilder/bilder.html “doch sollten diese nicht öfter eingesetzt werden, da manche Webseiten unter mehreren URL's erreichbar sind und diese nicht unbedingt die gleich Pfadstruktur aufweisen können. Stellen Sie sich beispielsweise eine mehrsprachige Website vor, bei der die deutschsprachigen

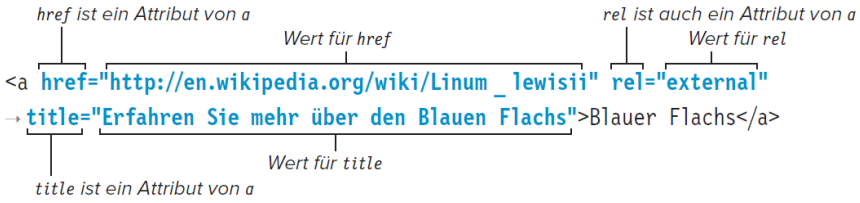

Abbildung 9: Aufbau eines <a> -Tags mit Atributen. Der Text zwischen <a>... wird auf der Seite angezeigt.

```
<a href="https://www.youtube.com/"> Youtube </a>
```

Abbildung 10: Das a-Element mit Hyperlink auf eine andere Web-Adresse.

```
<nav>
    <ul>
        <li><a href="header.html">  Eine neue Seite </a></li>
    </ul>
</nav>
```

Abbildung 11: Das <nav> Element umschließt Navigationsleisten. Das <ul> als auflistung von Elementen und das <a> Element, dass auf eine andere html-datei verlinkt.

Inhalte sowohl unter http:// www.my-international-website.com/de/... als auch unter http://de.my-international-website. com/... aufgerufen werden können. Hier wäre der jeweilige absolute Pfad natürlich unterschiedlich, etwa /de/new/news.html in der ersten und /new/news.html in der zweiten URLStruktur. Relative Pfade funktionieren dagegen in beiden Fällen identisch.

Mit dem **<a> ... ** lässt sich auch ein Hyperlink zu einer E-Mail Adresse erstellen. Mit dem Attribut href = “mailto:ich@mein-provider.de “öffnet sich automatisch ihre Mail-Anwendung.

Bei sehr umfangreichen Seiten, deren Länge sich über mehrere Bildschirmhöhen erstreckt, bietet es sich an, einzelne Passagen der Seite per Hyperlink erreichbar zu machen. Zu diesem Zweck werden sogenannte Anker oder Textmarken gesetzt, die einen bestimmten Punkt im Dokument als Ziel für einen Hyperlink definieren. Beim Anklicken eines Links, der auf einen Anker verweist, wird der Inhalt des Browserfensters automatisch so gescrollt, dass sich die entsprechende Stelle am oberen Rand befindet. Mit   lässt sich so ein Anker erstellen. Mit Zum Ankerpunkt springt man zu dem vorher gesetzten Anker. Um einen Ankerpunkt einer anderen Datei anzuspringen benutzt man <a href= “DateiName#AnkerName“>

Durch das <img src= “Pfad/Name des Bilds “alt = “Bildbeschreibung “> - Element können wir auch noch ein Bild einfügen.

- ... ungeordnete Liste
- ... geordnete Liste
- ... Listeneintrag

- <table> ... </table> Anlegen einer Tabelle, innerhalb dieser kann man mit
- <tr> ... </tr> eine neue Zeile anlegen und mit
- <th> ... </th> den Tabellenkopf beschriften und mit
- <td> ... </td> die Tabelleneinträge
- <dl> ... </dl> Bezeichnet eine Definitionsliste, wobei
- <dt> ... </dt> den Titel bzw. den zu definierenden Begriff und
- <dl> ... </dl> den Eintrag bzw. die Definition anzeigt.

Bei geordneten Listen (<ol type= > bzw. <li type = >) kann man außerdem das **type**, oder **start**-Attribut verwenden. Der „type“gibt an um welche Art der Nummerierung es sich handelt wie z.B 1 (standart), A, a (Groß-,Kleinbuchstaben), I, i(Große bzw. kleine römische Zahlen). Der „start “beeinflusst ab welchem Wert die Nummerierung anfängt. Bei ungeordneten Listen gibt es beim **type** die Auswahl zwischen disc, circle und square.

5.2 HTML5 - Strukturelemente

Zur besseren Strukturierung von Dokumenten führt HTML5 einige neue Tags ein. Da alte Browser, die diese nicht verstehen, sie einfach ignorieren, können Sie sie bedenkenlos einsetzen. Es handelt sich um folgende Elemente:

- <section> ... </section> umschließt einen Sinnabschnitt, der typischerweise aus einer Überschrift und einem oder mehreren Absätzen besteht.
- <article> ... </article> ist eine logische Informationseinheit, die sich klar von anderen solchen Einheiten abgrenzen lässt – beispielsweise einer von mehreren gleichzeitigen Blogeinträgen.
- <aside> ... </aside> beschreibt eine Nebeninformation, etwas, das man in einem Buch oder einer Zeitschrift beispielsweise in einem separaten Kasten darstellen würde.
- <header> ... </header> stellt den Kopf des Dokumentinhalts dar, der üblicherweise das Logo und den Namen der Organisation oder Website sowie die Hauptnavigation enthält.
- <footer> ... </footer> ist entsprechend der Dokumentfuß, typischerweise mit Copyright- Informationen und ein paar Service-Links (Impressum, FAQ, Kontakt etc.).
- <nav> ... </nav> ist für Navigationsbereiche gedacht.
- <figure> ... </figure> ist die strukturelle Auszeichnung für eine Abbildung, eine Grafik oder auch ein Video samt Beschriftung.

- Die Beschriftung steht innerhalb des figure-Blocks zwischen <figcaption> und </figcaption>.

Bei früheren HTMl Versionen wurde das „align “-Attribut benutzt um die Textausrichtung zu bestimmen. Dies kann man nach wie vor machen, doch sollte die Ausrichtung beim HTML-5 Standard über CSS erfolgen. Das **align = ““** Attribut kann in z.B. in Absätzen oder Überschriften benutzt werden.

- ... für einen linksbündigen Absatz.
- ... für einen rechtsbündigen Absatz.
- ... für einen zentrierten Absatz.

Eine spezielle Formatierung erzeugt der **<pre>...</pre>** tag. Dieser zeigt einen vorformatierten Text an, also samt Zeilenumbrüchen und Leerzeichen. Mit **
** lässt sich übrigens auch ein Zeilenumbruch erzwingen.

5.3 Zeichenformatierung

Obwohl die Zeichenformatierung, sowie das allgemeine Design einer Webseite, zunehmend mit CSS gestaltet wird, gibt es bestimmte Zeichenformatierungstags. Diese werden direkt im Textfluss eingebettet und können miteinander verschachtelt werden. Wobei man hier eine Unterscheidung zwischen den älteren Struktur-Tags und den neueren Layout-Tags. Die Struktur-Tags beschreiben die Bedeutung bestimmter Zeichen und die Layout-Tags beschreiben das Aussehen. In Abbildung 12 sind die wichtigsten Abgebildet.

5.4 Mapping

Anstatt ein ganzes Bild zu einem durchgehenden Hyperlink zu machen, besteht auch die Möglichkeit, einzelne Bereiche des Bildes anklickbar zu machen und individuell auf einen Klick in die jeweiligen Regionen zu reagieren. Grundsätzlich werden zwei Arten dieser sogenannten Image Maps (verweissensitiven Grafiken) unterschieden: die nur noch selten verwendeten serverseitigen Image Maps sowie die clientseitigen Image Maps.
Bei einer serverseitigen Image Map wird einfach das Attribut ismap=ïsmap"2 in das <img>-Tag gesetzt. Das Bild wird als normaler Hyperlink auf ein serverseitiges Skript verwendet, das in der Lage ist, die übertragenen Mauskoordinaten des Bildes zu verarbeiten.

Bei einer clientseitigen Image Map wird ein Bild mit beliebig vielen anklickbaren Bereichen, sogenannten Hot Spots, versehen, die jeweils getrennt als Hyperlinks angezeigt werden sollen. Eine solche Image Map wird durch die Angabe des Attributs usemap=“#Map-Name “erzeugt. Mit diesem Map-Namen muss eine <map>-Definition übereinstimmen, die die einzelnen Hot Spots definiert. Das Bild selbst muss (und sollte)

Tag	Wirkung	Beispiel
Layout-Tags		
`<i>...</i>`	kursiv (*italic*)	*kursiv*
`<b>...</b>`	fett (*bold*)	**fett**
`<u>...</u>`	unterstrichen (*underlined*)	unterstrichen
`<strike>...</strike>`	durchgestrichen (*strike through*)	~~durchgestrichen~~
`<sup>...</sup>`	hochgestellt (*superscript*)	normal $^{\text{hochgestellt}}$
`<sub>...</sub>`	tiefgestellt (*subscript*)	normal $_{\text{tiefgestellt}}$
`<tt>...</tt>`	Festbreitenschrift (*teletype*)	`Festbreitenschrift`
Struktur-Tags		
`<em>...</em>`	betont (*emphasis*); wird meist kursiv dargestellt	*betont*
`<strong>...</strong>`	stark betont; wird meist fett dargestellt	**stark betont**
`<code>...</code>`	Quellcode – zur Darstellung von Programmierbeispielen etc.; meist durch Festbreitenschrift dargestellt	`Code`
`<address>... </address>`	Adressangaben; meist kursiv dargestellt	*Rheinwerkallee 4, 53227 Bonn*

Abbildung 12: Die wichtigsten Tags zur Zeichenformatierung.[2]S.1020

```
<img src="maptest.gif" width="400" height="400" alt="Map-Bild" usemap="#mymap" />
<map name="mymap">
  <area shape="rect" coords="10, 10, 100, 100"
  href="rectangle.html" alt="Rechteckiger Bereich" />
  <area shape="circle" coords="300, 300, 50"
  href="circle.html" alt="Kreisförmiger Bereich" />
  <area shape="poly" coords="150, 150, 200, 10, 250, 150"
  href="polygon.html" alt="Polygonförmiger Bereich" />
</map>
```

Abbildung 13: Beispiel der verschiedenen Area-Formen.

dabei kein Hyperlink sein. Eine solche Map- Definition sieht zum Beispiel folgendermaßen aus (Abbildung 13) :

Die einzelnen <area>-Tags definieren die anklickbaren Bereiche. Das Attribut shape gibt die Form des jeweiligen Bereichs an:

- shape = rect: bezeichnet ein Rechteck und legt die linke obere und rechte untere Ecke fest.(x1,y1,x2,y2)
- shape = circle: bezeichnet ein Kreis und legt den Mittelpunkt (x- , y-Koordinaten) und den Radius.
- shape = poly: bezeichent Ein Polygon und erwartet eine Reihe von Koordinaten.

5.5 Tabellen

Wir haben bereits Tabellen in HTML kennengelernt. Diese lassen sich zum großen Teil bereits ohne CSS formatieren, auch wenn dies nicht empfohlen wird. Unterschieden wird hier zwischen der direkten Tabellenattribute und der Zellenparameter. Als Tabellenattribute gibt es z.B.:

- **align="..."** (left; right; center) richtet die Tabelle entsprechend aus.
- **width="..."** setzt die Breite fest. Wird ignoriert falls es nicht reinpasst.
- **height="..."** setzt die Höhe fest. Wird ignoriert falls es nicht reinpasst.
- **border="..."** setzt die Breite des Rahmen in Pixeln fest.
- **cellpadding="..."** bestimmt die Entfernung des Zelleninhalts zum Zellenrand.
- **cellspacing="..."** setzt die Breite der Zellenränder fest.

Tabellen kann man außerdem in Kopf-(**<thead>...</thead>**), Haupt-(**<tbody>...</tbody>**) und Fußteil(**<tfoot>...</tfoot>**) aufteilen. Außerdem besitzt das <table> Element noch weitere Zusatzattribute. Mit **<table rules="...">** kann man die Regeln für das Zeichnen der Gitternetzlinien steuern. rules="..."

- none: sind keine Innenlinien zusehen;
- rows: nur Zeilenlinien;
- cols: nur Spaltenlinien;
- groups: nur um Kopf-,Haupt- und Fußteil;
- all: Standard - alle werden gezeichnet.

Mit Hilfe von **<table frame="... ">** kann man die Außenrahmen steuern.

- none: kein Außenrahmen;

- above: nur der obere Rand;
- below: nur der untere Rand;
- lhs: (left hand sided) - nur der linke Rand;
- rhs: (right hand sided) - nur der rechte Rand;
- hsides: nur die beiden horizontalen Randlinien;
- vsides: nur die vertikalen Linien;
- box: Standard - alle Randlinien werden gezeichnet.

5.5.1 Zellenparameter

Der Inhalt einer Zelle kann übrigens beliebige Elemente, die HTML unterstützt, enthalten. Also Texte, Bilder, Hyperlinks oder auch weitere Tabellen. Eine Auflistung verschiedener Zellenparameter finden Sie in Abbildung 14.

5.6 Das head-Element

Das head-Element beinhaltet Metadaten, also Daten über Daten. Es wird am Anfang des HTML-Dokumentes platziert, direkt hinter <html> und vor <body>. Diese Metadaten werden nicht angezeigt und dienen z.B. als Informationsquelle für Suchmaschinen. Der <head>-tag kann folgende Informationen beinhalten:

- <title> Setzt den Namen für die Seite, wird auch im Browser-Tab angezeigt.
- <style> Setzt den CSS-Style für diese HTML-Seite.
- <link> Erzeugt einen Zusammenhang zwischen dieser Seite und einem anderen Dokument, z.B. einem Stylesheet. Mit <link rel= “stylesheet “href = “style.css “> verbindet man das stylesheet **style.css** mit dieser html Seite.
- <script> Hier wird Cient-basierter Javascript definiert.
- <base> Definiert eine Basis-URL auf die sich die anderen Links dieser Seite beziehen.

5.6.1 Das meta-Element

Das <meta>-Element bietet weitere Möglichkeiten Einstellungen und Informationen zu der HTML-Seite zu erstellen.

- <meta charset= “... “> stellt den Zeichensatz, der benutzt wird ein.
- <meta name=“keywords “content= “Beispiel, Übung, ... “> setzt die Wörter die bei den Suchmaschinen berücksichtigt werden.

- `align="center"|"right"|"left"|"justify"` richtet den Inhalt der Zelle horizontal aus (zentriert, rechtsbündig, linksbündig, Blocksatz). Standard ist die linksbündige Ausrichtung.
- `valign="middle"|"top"|"bottom"` richtet den Zelleninhalt vertikal aus (mittig, oben, unten). Standard ist die mittige Ausrichtung.
- `colspan="..."` gibt an, dass sich diese Zelle über die angegebene Anzahl von Spalten der Tabelle erstrecken soll; natürlich gibt es dann in der entsprechenden Zeile der Tabelle weniger Zellen.
- `rowspan="..."` bestimmt, dass sich die Zelle über die angegebene Anzahl von Zeilen der Tabelle erstrecken soll. Die Definition erfolgt in der obersten Zeile, in der die Zelle beginnt; in allen anderen betroffenen Zeilen braucht die betreffende Zelle nicht mehr definiert zu werden.
- `width="..."|"...%"` setzt die Zellenbreite in Pixeln beziehungsweise relativ zur gesamten Tabelle. Wird die Breite jeder Zelle in Pixeln angegeben, sollte eine Angabe der Tabellenbreite unterbleiben. Es genügt übrigens, wenn Sie diese Angaben einmal in den Zellen der obersten Zeile vornehmen (alternativ verstehen neuere Browser auch die im nächsten Abschnitt vorgestellten `<colgroup>`-Tags).

  Generell muss zu `width`-Angaben noch gesagt werden, dass die Breitenangabe ignoriert wird, falls der Inhalt einer Zelle in der jeweiligen Spalte zu breit ist, um in der gewünschten Zellenbreite angezeigt zu werden.
- `height="..."|"...%"` stellt die Höhe der Zelle in Pixeln beziehungsweise in Prozent der Tabellenhöhe ein. Auch hier wird die vollständige Darstellung des Inhalts gegenüber der Höhenangabe bevorzugt.

Abbildung 14: Verschiedene Attribute für die Formatierung der Zellen einer Tabelle. [2]S.1036

- <meta name= “description “content = “Beschreibung der Seite “> beschreibt den Inhalt der Seite
- <meta name= “author “content = “Roman Schmal “> definiert einen Autor.
- <meta http-equiv= “refresh “conten = Zeit in Sekunden (z.B.) “30 “> lässt die Seite alle x Sekunden aktualisieren.
- <meta name= “viewport “content= “width=device-width, initial-scale=1.0 “> Setzt die Breite des gezeigten auf die Breite des Displays.
- <meta name= “robots “content=“noindex “> weißt alle Suchmaschinen-Crawler an, diese Seite nicht in Suchergebnissen anzuzeigen.

5.7 Formulare

Suchmaschinen, Bestellseiten oder Webforen bestehen aus diversen Eingabeelementen wie Textfeldern oder Auswahlbuttons. Der übergeordnete Container für eine Gruppe solcher interaktiven Elemente ist das HTML-Formular. Seine Definition enthält eine URL, an die sämtliche eingegebenen Daten versandt werden. Formulare werden mit**<form action= “.“method= “.“>.....</form>** erzeugt. Beim action-attribut steht meist *URL* und gibt die Adresse eines Serverskripts an. Die verbreitetste Skriptsprache hierfür ist **PHP** und wird hier nicht besprochen. Das Attribut *method* gibt an über welche HTTP-Methode die Daten versendet werden. Eine Andere Methode ist z.B. *get* und hängt die Formulardaten in der Form **url?date** an die URL an. Dies sieht man oft bei Suchanfragen von Suchmaschinen.

5.7.1 Formularelemente

Innerhalb der des <form>...</form> Containers gibt es viele Unterschiedliche Eingabeelemente, die mit **<input type= “Elementtyp “name = “Elementname “value = “Wert “/>**. Das type-attribut gibt an, um welchen Eingabetypen es sich handelt. Value und name bilden ein Namen-Werte-Paar, dass mit den Daten übersendet wird. Die Eingabetypen sind: (also <input type = “... “)

- **radio:** definiert einen sogenannten Radiobutton. Der Name stammt daher, dass sich dieser Button verhält wie die Knöpfe an alten Radios: Wird einer von ihnen gedrückt, springt der zuvor ausgewählte automatisch heraus. Mit Radiobuttons können Sie die Auswahl einer einzigen Option aus mehreren Alternativen ermöglichen. name bezeichnet dabei den Namen der Gruppe, zu der der Radiobutton gehört. value ist der Inhalt, der als Auswahl für diese Gruppe in den Formulardaten erscheinen soll. Der Text, mit dem der Button beschriftet werden soll, wird einfach hinter das Tag gesetzt, oder es kommt das später beschriebene <label>-Tag zum Einsatz.

- **checkbox:** definiert eine Checkbox, mit der mehrere Optionen an- und wieder abgewählt werden können. Der optionale Parameter checked="checked"kann bei Checkboxen und Radiobuttons stehen und hat zur Folge, dass die entsprechende Option innerhalb ihrer Gruppe bereits ausgewählt ist.

- **text:** bietet ein Feld zur Texteingabe an; optional gibt size die Breite in Zeichen und maxlength die maximale Eingabelänge an. Der ebenfalls optionale Parameter value sorgt dafür, dass der betreffende Text bereits voreingetragen im Textfeld steht.

- **password:** unktioniert im Prinzip genau wie ein Textfeld – mit dem Unterschied, dass die Eingabe als **** angezeigt wird.

- **submit:** stellt einen Button zur Verfügung, der durch Mausklick den Inhalt des Formulars an die URL versendet, die im <form>-Tag angegeben wurde. value hat hier eine etwas andere Bedeutung: Es enthält den Text, mit dem der Button beschriftet wird.

- **reset:** setzt alle Änderungen, die im Formular vorgenommen wurden, auf den Ursprungszustand zurück. value enthält wiederum die Beschriftung des Buttons.

- **button:** stellt eine allgemeine Schaltfläche zur Verfügung, deren Aussehen einem Absenden- oder Löschen-Button entspricht. In Zusammenarbeit mit JavaScript kann sie ein benutzerdefiniertes Ereignis auslösen.

- **hidden:** ist im engeren Sinne kein Eingabefeld, denn es wird vom Browser nicht angezeigt, und es können keine Eingaben vorgenommen werden. Es handelt sich um eine festgelegte Angabe, die zusammen mit den im Browser eingegebenen Formulardaten übertragen wird. Nützlich sind Hidden-Felder etwa für Ordnungszwecke (welches Formular wurde eigentlich versandt?) oder für Zwischenwerte, die von serverseitigen Programmen generiert wurden und wieder mit den Daten verschickt werden müssen, um vom nächsten Skript aus darauf zurückgreifen zu können.

- **file** bietet die Möglichkeit, den Pfad einer lokalen Datei einzutippen oder über einen mitgelieferten Button interaktiv auszuwählen. Wenn das Empfängerskript über eine entsprechende Einrichtung verfügt, wird die angegebene Datei zusammen mit den anderen Formulardaten hochgeladen. Beachten Sie, dass der Formulardatentyp über das <form>-Attribut enctype auf “multipart/form-data “gesetzt werden muss, damit es funktioniert.

5.7.2 Weitere Formularelemente

Ein Formular kann auch Menüs enthalten, aus denen eine Auswahl getroffen werden kann. Diese Menüs haben die Syntax, die in Abbildung 15 dargestellt ist.

Die Beschriftung der Input-Elemente wird mit **<label for= “... “> Beschriftung </label>** gemacht. Das *“for “*-Attribut gibt hier eine eindeutige Zuweisung, wie eine ID, die die Beschriftung einem Formularelement zuordnet.

```
<select name="..." size="...">
  <option value="..."> 1. Auswahlmöglichkeit </option>
  <option value="..."> 2. Auswahlmöglichkeit </option>
  <!-- bei Bedarf weitere Optionen -->
</select>
```

Abbildung 15: Eine solche Struktur stellt ein Menü zur Verfügung. Die size gibt hier die anzahl der sichtbaren Zeilen an. Mit size = “1 “wird automatisch ein Pull-down-Menü erzeugt. Bei <option>...</option> steht die Auswahlmöglichkeit und mit dem Attribut *selected= “selecet “* setzt man die Standartauswahl. Mit *disabled* kann man eine Auswahlmöglichkeit ausschalten.

6 Cascading Style Sheets - CSS

Die erste Version von CSS erschien erst, nachdem HTML schon ein paar Jahre im Einsatz war, und wurde 1996 offiziell verabschiedet. Wie HTML5 und seine Beziehungen zu den früheren Versionen von HTML auch ist CSS3 eine natürliche Erweiterung der vorangegangenen CSS-Versionen. CSS3 ist leistungsfähiger als die früheren Versionen von CSS und führt verschiedene visuelle Effekte ein, z.B. Schlagschatten, Textschatten, abgerundete Ecken und Farbverläufe.

6.1 CSS-Regel konstruieren

Jede CSS-Regel in einem Stylesheet hat zwei Hauptteile: den Selektor , der bestimmt, welche Elemente betroffen sind, und den Deklarationsblock , der aus einem oder mehreren Eigenschaft-Wert- Paaren besteht (jedes Paar macht eine Deklaration aus) und in dem angegeben wird, was genau gemacht werden soll. Diese Angaben können sowohl an verschiedenen Stellen des Dokumentes stehen, als auch in eigener Datei. Das Vorgehen eine solche Regel zu konstruieren ist folgende:

- **1.** Tippen Sie den selektor ein (selektor definiert die jeweils zu formatierenden Elemente). Sie erfahren in Kapitel 9, wie man alle möglichen Selektoren erstellt.
- **2.** Geben Sie (öffnende geschweifte Klammer) ein, um den Deklarationsblock zu beginnen.
- **3.** Tippen Sie eigenschaft: wert;, wobei eigenschaft der Name der CSS-Eigenschaft ist, die die von Ihnen gewünschte Formatierung anwendet, und wert eine der verschiedenen, für diese Eigenschaft erlaubten Optionen ist.
- **4.** Wiederholen Sie bei Bedarf Schritt 3. Üblich ist, jedes eigenschaft: wert-Paar (zusammen als Deklaration bezeichnet) in einer eigenen Zeile einzutragen.
- **5.** Mit schließen Sie die Deklaration und die CSS-Regel ab.

Um Klassen, welche man selber definiert hat zu formatieren, verwendet man einen Punkt-Operator und um eine ID anzusprechen das Raute-Symbol, wie Abbildung 18 und Abbildung 19 aufgeführt. Das die ID eine höhere Spezifikation aufweist, da eine ID nur einmal vorkommen darf. Wird diese die anderen Regeln überschreiben. Außerdem wird wie in Abbildung 18 zu sehen eine Eigenschaft bzw. dessen Wert für dieselbe ID zwei mal gesetzt und die letzte Regel überschreibt die zuvor (weil ganz unten).

6.1.1 Selektoren

CSS erlaubt noch weitere Kontext-Selektoren, um mehrere Elemente auf einmal auszuwählen und zwar abhängig von der Art der "Verwandtschaft".

- Nachfahren: Kontext Selektiertes Element → a img {...} Alle img-Tags, die innerhalb eines a-Tags liegen, auch in zweiter oder dritter Ebene.

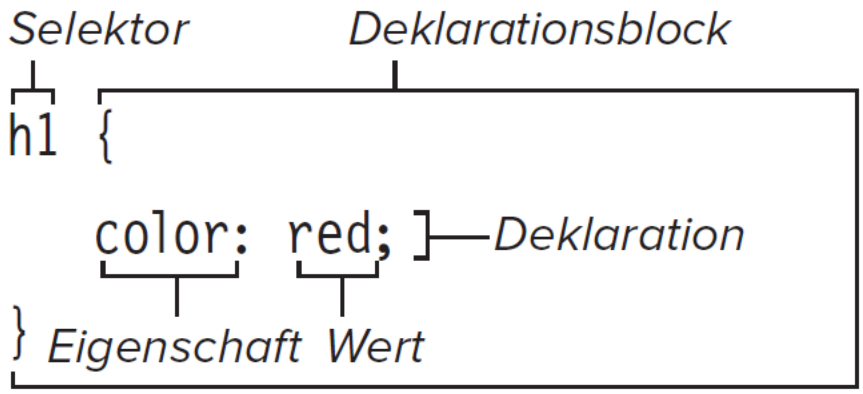

Abbildung 16: Eine CSS-Regel besteht aus einem Selektor und dem Deklarationsblock.

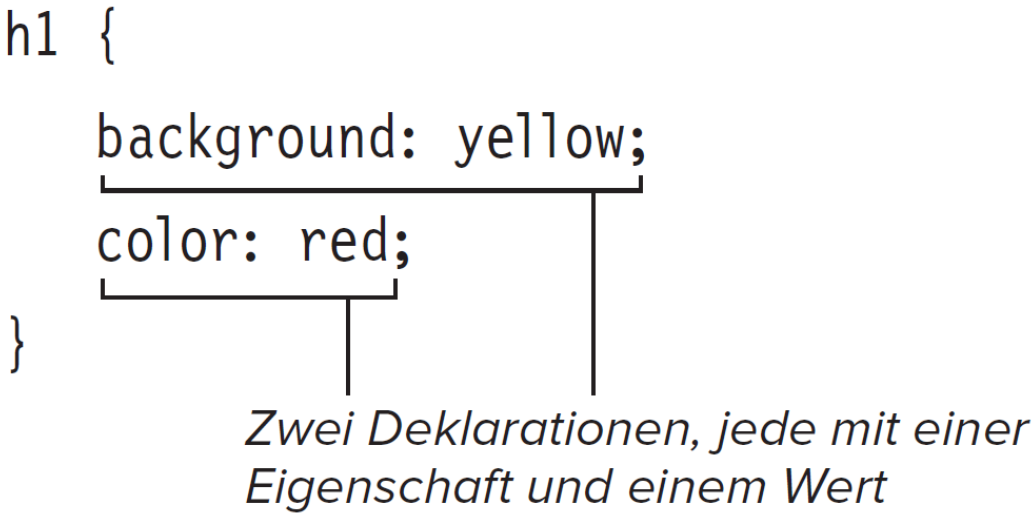

Abbildung 17: Bei mehreren Deklaratione ist die Reihenfolge nicht wichtig.

- Kind: Kontext > Selektiertes Element → div>table {...} Alle table-Elemente, die direkt innerhalb eines div-Elements liegen.
- Direkter Nachbar: Kontext + Selektiertes Element →h5 + p {...} Das p-Element, das direkt auf eine Überschrift h5 folgt.
- Indirekter(Allgemeiner) Nachbar → h5 p {...} Alle p-Elemente, die der Überschrift h5 folgen und auf derselben Ebene innerhalb eines Eltern-Elements liegen.
- Element[Attribut=String] wählt Elemente des entsprechenden Typs aus, bei denen das angegebene Attribut exakt den Wert String hat. Funktioniert auch ohne String,

```
p {
    color: red;
}

p.group {
    color: blue;
}

p#last {
    color: green;
}

p#last {
    color: magenta;
}
```

Abbildung 18: Beispiel von vier Regeln die abhängig von Klasse oder ID überschrieben werden.

heißt alle Elemente mit diesem Attribut.

- Element[Attribut*=String] schließlich trifft zu, wenn der String an irgendeiner Stelle im Attributwert vorkommt.
- Element:first-child wählt das erste Kindelement des angegebenen Elements aus.
- Element:last-child dient entsprechend der Auswahl des letzten Kindelements.
- Element:nth-child(n) ermöglicht den Zugriff auf ein bestimmtes Kindelement – beispielsweise :nth-child(3) für das dritte oder :nth-child(3n) für jedes dritte.

6.1.2 Responsive Designe

Mithilfe von Media Queries können Sie festlegen, dass bestimmte CSS-Angaben nur für bestimmte Ausgabemedien gelten sollen. Grundlegende Angaben über den Medientyp sind älter als CSS3; es war also schon recht lange möglich, etwa getrennte Stylesheets für die Bildschirm- und die Druckausgabe zu schreiben. Mit media screen { ... }, media print

```
<p>Hier ist ein allgemeines <code>p</code>-
→ Element. Es wird rot sein.</p>

<p class="group">Hier ist ein <code>p</code>-
→ Element mit der <code>class</code>
→ <code>group</code>. Es gibt zwei Regeln, die
→ angewendet werden, aber weil die Regel
→ <code>p.group</code> spezifischer ist, wird
→ dieser Absatz blau.</p>

<p id="last" class="group">Hier ist ein
→ <code>p</code>-Element mit der <code>id</
→ code> <code>intro</code>. Es gibt vier Regeln,
→ die auf diesen Absatz angewendet werden
→ könnten. Die ersten beiden werden von den
→ spezifischeren letzten beiden überschrieben.
→ Die Position unterbricht die Verbindung
→ zwischen den letzten beiden: die Regel, die
→ später kommt, gewinnt, und darum ist die
→ Schrift in diesem Absatz in Magenta.</p>

</body>
</html>
```

Abbildung 19: Beispielcode für die Abbildung 18.

{ ... } geben Sie die Angaben, welche beim screen(Bildschirm) oder beim print(Drucken) verwendet werden. Außerdem kann man mit media(min-width: ...px) { ... } die mindest-Anzahl an Pixeln angeben, bei dem die verwendeten CSS-Angaben gültig sein werden.

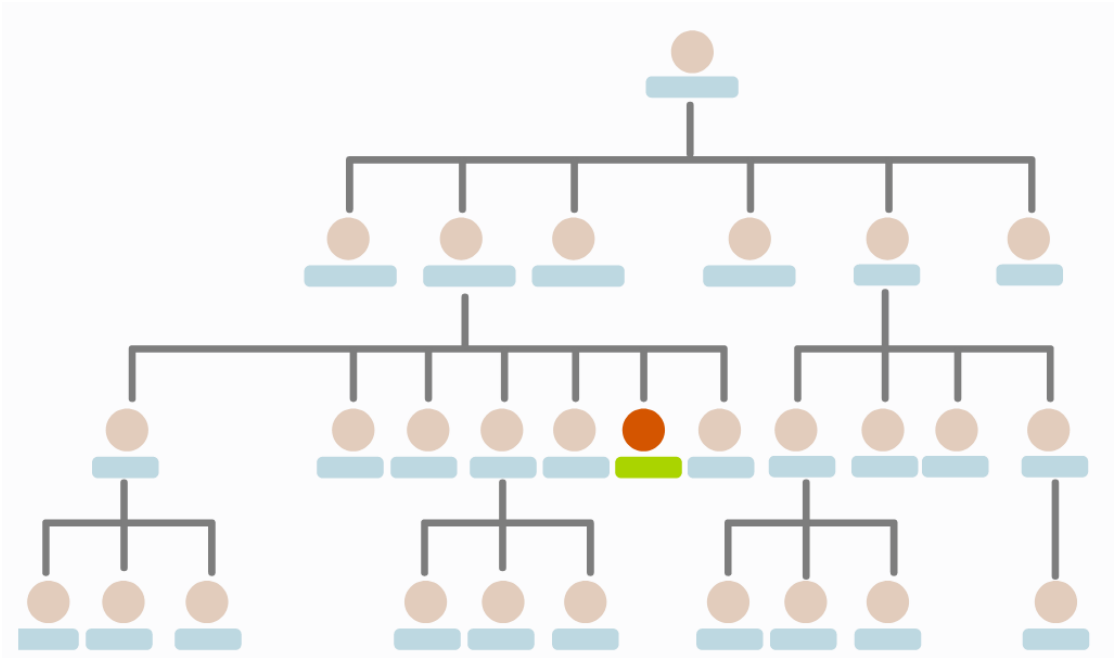

Abbildung 20: Kontext Selektoren in CSS werden relativ zu ihren Eltern-, Kind-, Nachbarn- und Nachfahrenelementen ausgewählt.

6.2 Rahmen und Linien

Mit CSS lassen sich auch Rahmeneinstellungen und die Abstände regeln.

- **margin-...**top; bottom; left; right; Diese vier Attribute bestimmen den Abstand des Bereichs zur Umgebung nach oben, unten, links und rechts. Der Wert ist numerisch. Alternativ gibt margin einen identischen Abstand auf allen vier Seiten an, wenn Sie nur einen Wert verwenden, oder Sie können vier durch Leerzeichen getrennte Einzelwerte angeben, die im Uhrzeigersinn (oben, rechts, unten, links) interpretiert werden.

- **padding-...**top; bottom; left; right; Diese Attribute geben den inneren Abstand eines Elements zu seinem Rand an. Auch hier ermöglicht ein einfaches padding eine einheitliche Angabe oder vier Einzelwerte für alle Seiten. Der Unterschied zu den margin-Einstellungen zeigt sich, wenn eine sichtbare Rahmenlinie verwendet wird: padding setzt den inneren Raum bis zu diesem Rahmen, während margin den Außenabstand des Rahmens zur Umgebung festlegt.

- **border-...**top; bottom; left; right; Die Werte für diese Attribute sind eine durch Leerzeichen getrennte Liste von Linienattributen für den äußeren Rand auf der angegebenen Seite eines beliebigen Elements. Stattdessen können Sie mit border auch einen einheitlichen Rahmen auf allen vier Seiten festlegen. Der erste Wert ist die Linienstärke mit den möglichen Werten *thin* (dünn), *medium* (mittel), *thick* (dick)

oder einem beliebigen numerischen Wert. Alternativ kann dieser Wert in Stylesheets auch separat als border-width (beziehungsweise border-top-width etc.) angegeben werden. Außerdem kann man den Linienstil bzw. Schattierungsarten(groove, ridge, inset, outset) angeben:

- solid: einfach durchgezogen;
- dotted: gepunktet;
- dashed: gestrichelt;
- double: doppelter Rahmen;

6.2.1 Werteangaben in CSS

Neben den festgelegten Werten, die CSS schon kennt, wie bei **font-family** die verschieden Schriftarten oder die Ausrichtung von Elementen(left, center, right, justify), werden auch numerische Werte sowie Farbwerte hergenommen. Die folgenden Abbildungen aus [2]S. 1056 f. zeigen die möglichen Werte.

Maßeinheit	Bedeutung	Hinweise
px	Pixel	Die am häufigsten vorkommende und wichtigste Maßeinheit.
pt	Punkt	Der DTP-Punkt, 1/72 Zoll. Wird gern für die Schriftgröße verwendet.
mm	Millimeter	Diese Angaben sind von der Bildschirmgröße und -auflösung abhängig. Gebrochene Werte werden mit einem Punkt (und nicht mit einem Komma) getrennt: `2.5cm` – nicht `2,5cm`.
cm	Zentimeter	
in	Inch/Zoll (1" = 2,54 cm)	
em	Breite des M	Entspricht dem Geviert, also der Höhe der aktuellen Schriftart. Praktisch als relative Angabe für die Schriftgröße – etwa `1.2em` für eine Schrift, die 1,2-mal so groß ist wie die Standardgröße.
ex	Breite des x	Wird manchmal als relative Angabe für Wort- oder Zeichenabstände verwendet.
%	Prozent	Im Allgemeinen relativ zum umgebenden Element (Tabellenzelle, Browserfenster). Bei Schriftangaben relativ zur Schriftgröße.

Abbildung 21: Die verschiedenen Maßeinheiten, die bei CSS zum Einsatz kommen.

Rotwert	Grünwert	Blauwert	HTML-Farbcode	Farbe
0	0	0	#000000	Schwarz
255	0	0	#FF0000	Rot
0	255	0	#00FF00	Grün
0	0	255	#0000FF	Blau
255	255	0	#FFFF00	Gelb
255	0	255	#FF00FF	Magenta
0	255	255	#00FFFF	Cyan
255	255	255	#FFFFFF	Weiß

Abbildung 22: Beispiele für Farbangaben bei CSS.

6.3 CSS - Grid

Mit Hilfe der Grid Layouttechnik, lassen sich Webseiten schnell Strukturieren. Dies ist ein zweidimensionales Rastersystem, das es dem Designer leichter macht die Objekte einer Webseite am richtigen Platz zu halten. Dieses Konzept ist für Elemente gedacht, anders als bei Flexbox, bei denen die "Box "- größen schon bekannt sind. Das Grid Layout kann man sich wie eine Tabelle vorstellen. Um mehr anschauliche Beispiele zu bekommen siehe [3].

Das Grid Layout besteht aus einem **Parent**-Element z.B. Container und einem **Child**-Element z.B. Item. Im HTML-Code definiert man dazu z.B. div Container mit Klassennamen und darin enthaltene div Container.

```
1
2   <div class="container">
3     <div class="item-1"> </div>
4     <div class="item-2"> </div>
5     <div class="item-3"> </div>
6   </div>
```

Im CSS Code wird dann der Container, also das Parent-Element als Grid-Element definiert.

```
1   .container {
```

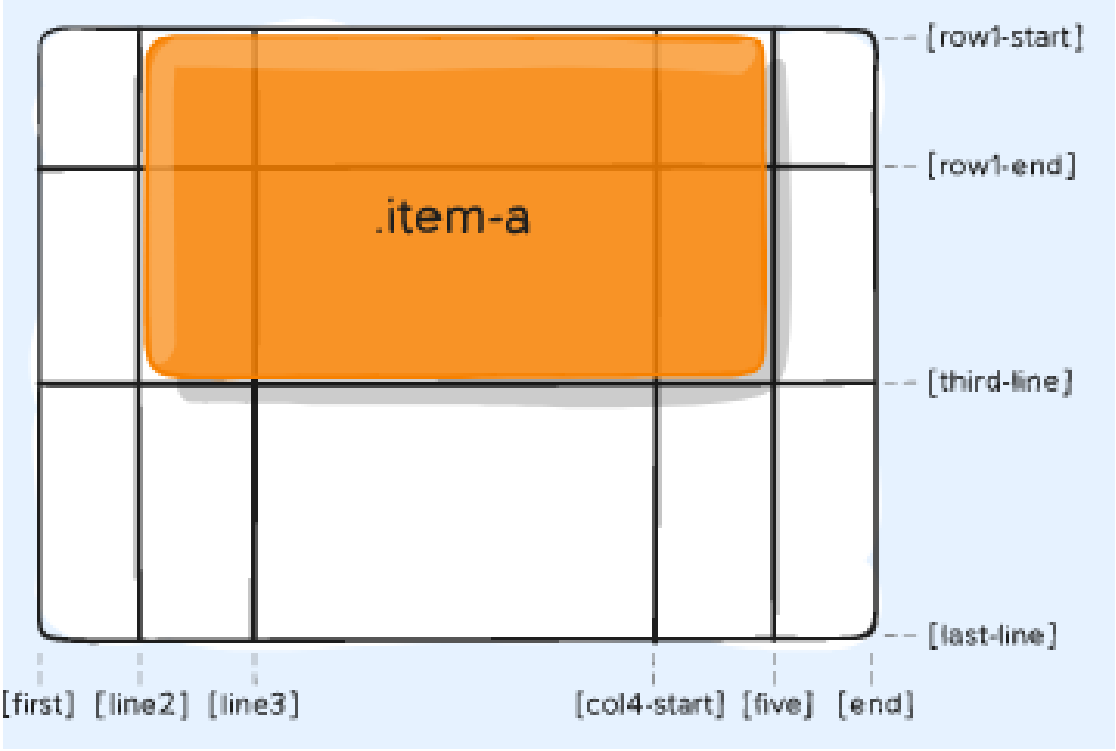

Abbildung 23:

```
    display: grid;
  }
```

Die Chlid-Elemente wiederum werden als solche deklariert. Der untere Code(.item-a { ...}) zeigt das Beispiel aus Abbildung 23.

```
  .item {
    grid-column-start: <number> | <name> | span <number> | span <name> |
        auto;
    grid-column-end: <number> | <name> | span <number> | span <name> |
        auto;
    grid-row-start: <number> | <name> | span <number> | span <name> |
        auto;
    grid-row-end: <number> | <name> | span <number> | span <name> | auto;
  }
Zum Beispiesl:
  .item-a {
    grid-column-start: 2;
    grid-column-end: five;
    grid-row-start: row1-start;
    grid-row-end: 3;
  }
```

Mit **grid-template-areas:** kann man ein direkt die Child-Elemente “namentlich “in die gewünschte Position Platzieren bzw. diese über mehrere “Zellen “verteilen. Der

untere Code erzeugt das Bsp. aus Abbildung 24.

```
1
2   .item-a {
3     grid-area: header;
4   }
5   .item-b {
6     grid-area: main;
7   }
8   .item-c {
9     grid-area: sidebar;
10  }
11  .item-d {
12    grid-area: footer;
13  }
14
15  .container {
16    display: grid;
17    grid-template-columns: 50px 50px 50px 50px;
18    grid-template-rows: auto;
19    grid-template-areas:
20    "header header header header"
21    "main main . sidebar"
22    "footer footer footer footer";
23  }
```

Weitere nützliche Eigenschaften:

- Mit der Eigenschaft **justify-self:** und den Werten: start, end, center, stretch; lässt sich die Ausrichtung der Zelle innerhalb der Zelle einstellen.
- Um den Zelleninhalt auszurichten, benutzt man **align-self**
- Mit **column-gap:** und **row-gap:** lässt sich ein Abstand zwischen den Zeilen bzw. Spalten einstellen.
- Mit **min(...)**, **max(...)** oder **minmax(...)** lässt sich der mindest- bzw. Maximalwert für die Länge einstellen
- Mit **justify-content** lässt sich der Inhalt des Containers anhand des Rasters ausrichten.

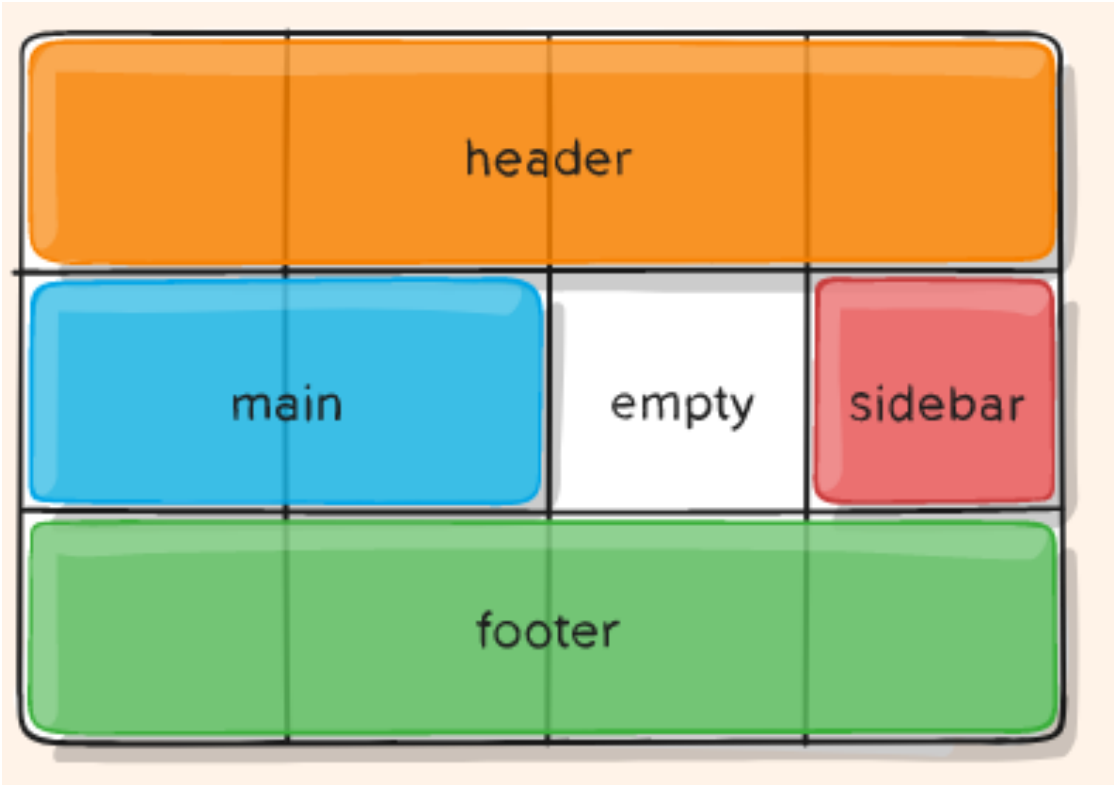

Abbildung 24:

7 JavaScript

Ursprünglich wurde JavaScript(JS) für dynamisches HTML in Webbrowsern entwickelt. Mittlerweile lässt sich JS auch Serverseitig betreiben. Der standardisierte Sprachkern von JavaScript beschreibt eine dynamische typisierte, objektorientierte, aber klassenlose Skriptsprache. In JS lässt sich je nach Bedarf objektorientiert, prozedural oder funktional programmieren.

objektorientiert(OOP): Komplexe Systeme werden durch Zusammenspiel von Objekten beschrieben.

prozedural: Anderes als bei OOP haben Daten und Funktionen kein Zusammenspiel. Auch traditionelle Programmierung genannt.

funktional: Funktionen werden wie Daten miteinander verknüpft z.B. rekursive Funktionen.

7.1 Grundlagen

Um JavaScript auf der HTML-Seite auszuführen, wird wie beim styling mit CSS der Code direkt im HTML Dokument eingebettet, oder in einer zusätzlichen Datei geschrieben. Bei beiden Methoden verwendet man den **<script>...</script>**-Tag. Falls man JS im HTML-Dokument schreibt, sollte man beachten, dass die Initialisierung im Head erfolgt. Script, welches etwas im HTML-Dokument ausgibt im Body und man möglichst viel Javascript-Code ans Ende des Dokumentes setzten sollte.

```
1  <script type="text/javascript">
2  // Hier steht JavaScript-Code.
3  </script>
```

Falls man bestimmte JS-Funktionalitäten in mehreren Dateien möchte oder sein Projekt etwas Übersichtlicher gestalten möchte, dann kann man mit folgender Angabe die Datei mit JS-Code angeben.

```
1  <script type="text/javascript" src="extern.js></script>
```

Um einfachen JavaScript Code auszuführen bzw. die ersten Ausgaben zu erzeugen sind folgende JS-funktionen nützlich:

- document.write(...); : schreibt etwas direkt auf die HTML-Seite, wobei wie man auch tags wie im HTML-dokument verwenden kann.
- prompt(“Anweisung: “,““); : Öffnet ein Dialogfeld mit dem eingegebenen Text(Anweisung:) und einem Texteingabefeld.
- console.log(...); console.warn(...); console.error(...);: Ausgabe auf der Konsole des Webbrowsers.
- alert(“...“): Es taucht ein Pop-up Fenster mit dem String auf.

```
// var: bleibt gültig
for (var i = 1; i <= 10; i++) {
  console.log("Aktueller Wert von i: " + i);
}
console.log("Wert von i nach Ausführung der Schleife: " + i);
// let: nur innerhalb des jeweiligen Codeblocks gültig:
for (let j = 1; j <= 10; j++) {
  console.log("Aktueller Wert von j: " + j);
}
console.log("Wert von j nach Ausführung der Schleife: " + j); // Fehlermeldung
```

Abbildung 25: Unterschied zwischen var und let. [2]S.1181

7.1.1 Variablen

Viele Aspekte im Umgang mit Ausdrücken, Operationen und Variablen in JavaScript sind den entsprechenden Konzepten in Java, Python oder PHP ähnlich. Die "Datentypen "in JavaScript sind **var**, **let** und **const**. Während Variablen, die mit var deklariert wurden um gesamten Code gültig sind, sind let-Variablen nur im aktuellen Codeblock gültig(siehe Abbildung 25). Konstanten(const) bilden eine festgelegte Referenz, so kann man einen const Array erzeugen, dessen Werte unverändert bleiben aber der an sich erweiterbar ist.

7.1.2 Arrays und Objekte

Echte Arrays haben in JavaScript immer numerische Indizes. Formal betrachtet, ist eine Sammlung mit String-Indizes kein Array, sondern ein (anonymes) Objekt. Ein solches können Sie wie folgt erzeugen:

```
var Objekt = {
  'ObjektName1': 'Wert1',
  'ObjektName2': 'Wert2',
  ...
};
```

Auf die Felder können Sie anschließend mit Objekt[‘ObjektName1‘] oder Objekt. ObjektName1 zugreifen. Diese Werte müssen nicht Strings sein, sondern dürfen auch Zahlen, boolsche Werte weitere Objekte oder anonyme Funktionen sein. Mit for - in bzw. for - of kann man in Javascript über die Elemente oder die Indizes iterieren:

```
let testArray = ['erste', 'zweite','dritte'];

for( let i in testArray) {
  // i hat dann die Werte 0, 1, 2.
  // und mit testArray[i] greift
  // man auf die Werte des Array zu
}
```

```
8
9  for( let werte of testArray) {
10   // werte hat dann die Werte 'erste', 'zweite', 'dritte'.
11 }
```

Bei Objekte benutzt man hingegen die Object.key Methode und nur die for - of schleife:

```
1   let person = {
2   "Vorname": "Roman",
3   "Nachname": "Schmal",
4   "Arbeit": "Dozent"
5  };
6
7  for( let key of Object.key(person)) {
8   // key ist nun nacheinander "Vorname", "Nachname", "Arbeit",
9   // und mit person[key] greift man auf die Werte(Roman, Schmal, Dozent )
        zu
10 }
```

7.1.3 Funktionen

Auf das Schlüsselwort function folgt der Bezeichner der Funktion, anschließend steht in Klammern die (möglicherweise leere) Parameterliste; optionale Standardwerte dürfen in JavaScript allerdings nicht angegeben werden. Den Rumpf der Funktion bildet schließlich ein Block, also ein Bereich in geschweiften Klammern. Schematisch sieht das Ganze daher wie folgt aus:

```
1   function BSPFunktion( arg1, arg2, ...) {
2   }
```

Funktionen sind in JavaScript eine spezielle Form von Objekten. Sie können in einzelnen Variablen, Arrays oder Objekten gespeichert und sogar von anderen Funktionen als Rückgabewert geliefert werden. Das vorherige Beispiel ist eigentlich nur eine Kurzfassung für:

```
1   var BSPFunktion = function( arg1, arg2, ...) {
2   }
```

7.1.4 Lambdafunktionen

In Javascript gibt es sog. "Arrow-Functions", die eine Ergänzung zu den normalen Funktionsdeklarationen sind. Eine normale Funktionen wie z.B:

```
1   (function (a, b) {
2     return a + b + 100;
3   });
4   // bzw. ohne Argumente
5   (function() {
6     return a + b + 100;
7   });
```

kann man in Javascript auch so schreiben:

```
1
2   (a, b) => a + b + 100;
3   // bzw. ohne Argumente
4   () => a + b + 100;
```

Wichtige und Nützliche Hilfsmethoden:

- map() das Element kann beliebig bearbeitet werden und gibt dann ein neues Element mit den überarbeiteten Werten zurück.

```
1       let numbers = [1, 2, 3, 4, 5];
2       let squares = numbers.map(number => number * number); // [1, 4,
            9, 16, 25]
```

- filter() erwartet einen boolean bzw. eine logische Bedingung und übernimmt das Element falls es true ist.

```
1       let even = numbers.filter(zahl => zahl % 2 == 0); // [2, 4]
```

- some() gibt lediglich einen Bool zurück und zwar true, wenn mindestens ein Element der Bedingung genügt.

```
1       let areThereEvenNumbers = numbers.some(number => number % 2 ==
            0); // true
```

- reduce() erwartet zwei Argumente. Das erste ist der Aggregator, also eine temporäre Variable die geändert wird und in der geänderten Form weitergegeben wird. Das zweite ist das jeweilige Element.

```
1       let sum = numbers.reduce((s, number) => s + number); // 15
2       let product = numbers.reduce((p, number) => p * number); // 120
```

- split() ist eine String-Methode aber auch sehr nützlich. Diese Unterteilt einen Satz in Substrings, anhand des übergebenen Arguments und gibt einen Array aus diesen Strings zurück.

```
1       let words = "JavaScript kann viel".split(" ");
2       let nWords = words.map((word, i) => (i + 1) + ". " + word);
3       // ["1. JavaScript", "2. kann", "3. viel"]
```

7.2 Formulare und Event-Handler

Das einzige Problem besteht darin, dass Skripte aus Formularen heraus explizit aktiviert werden müssen. Die bisherigen Beispiele mit den prompt()-Boxen liefen automatisch beim Dokumentaufruf ab, ein Formular steht dagegen die ganze Zeit auf der Seite zur Verfügung, und typischerweise entscheidet ein Klick auf einen Button, wann das entsprechende Skript aufgerufen wird. Dazu werden sogenannte Event-Handler (etwa »Ereignisverarbeiter«) verwendet. In ihrer klassischen Form handelt es sich um HTML-Attribute, die jedoch den speziellen Zweck haben, JavaScript-Code auszuführen, der ihren Parameterwert bildet.

```
<input type="button" value="OK"
onclick="document.form1.test.value="Hallo" />
```

Mit dieser Anweisung erstellt man ein <input>-Element des Typs Button, dass wenn man diesen anklickt das kleine Skript ausgeführt wird. In diesem Beispiel wird Das Formular mit der **id=“form1 “** und dem input-element mit der **id=“test “** angesprochen und dessen Wert, also **value** = der angezeigte Text, geändert in “Hallo “.
Da dies etwas Umständlich bzw. unübersichtlich werden kann gibt es in Javascript **EventHandler**.

```
objekt.addEventListener(event, funktion);
```

Wobei *event* ein String mit dem gewünschtem Event-Namen wie etwa “click “ist und *funktion* die “function “die dabei ausgelöst wird.

7.2.1 Zugriff auf Formulare/Elemente

Der Zugriff auf Formulare und deren Elemente geschieht grundsätzlich mit Hilfe von **document.forms**[]. Da sich alle <form>...</form> Elemente innerhalb eines Dokumentes in einem Objekt-Array befinden, heißt das erste Form-Element ist 0 das zweite ist 1 usw. .

Falls man dem Formular einen Namen gegeben hat, also **<form name="formname»....</form>**, dann kann man auch dieses Formular mit **documen.formname** oder **document.forms[‘formname ‘]** ansprechen. Das selbe Beispiel gilt auch für die Child-Elemente des Formulars. z.B:

```
<form name="test">
<input type="text" name="textInput" />
<input type="button" value="OK" />
</form>
// Der Zugriff auf das Input-Element "textInput"
// kann somit auf folgende Weise erfolgen:

document.forms[0].elements[0]
document.forms[0].elements["textInput"]
document.forms[0].textInput
document.forms["test"].elements[0]
document.forms["test"].elements["textInput"]
document.forms["test"].textInput
document.test.elements[0]
document.test.elements["textInput"]
document.test.textInput
```

7.2.2 Datum und Uhrzeit

Da es oft wichtig ist die Anwendungen von Dem Datum bzw. Wochentag und Monat abhängig zumachen und oder der derzeitigen Uhrzeit, kann man in Javascript ein neues Datumsobjekt mit z.B. **var now = new Date()** erzeugen.

Dieses Objekt hat eine Reihe von Methoden mit denen man die einzelnen Felder ansprechen kann:

- now.getYear() : gibt das zweistellige Jahr zurück. z.B. 22
- now.getFullYear() : gibt das vierstellige Jahr zurück
- now.getMonth() : gibt den Monat als Zahl aus, wobei der Januar eine 0 und der Februar eine 1 ist usw.
- now.getDate() : liefert den Tag im Monat von 1 bis 31.
- now.getDay() : gibt den Wochentag in numerischer Form aus, wobei der Sonntag die 0, der Montag die 1 usw. ist.
- now.getHours() : gibt die Stunde zwischen 0 und 23 zurück.
- now.getMinutes() : liefert die Minute zwischen 0 und 59.
- now.getSeconds(): gibt die Sekunde zwischen 0 und 59 zurück.

- now.toString() gibt Datum und Uhrzeit als String zurück.
- now.toTimeString() liefert nur die Uhrzeit als String.
- now.toLocaleString() gibt Datum und Uhrzeit in der Schreibweise als String zurück, die der aktuellen Ländereinstellung des Browsers/Betriebssystems entspricht.
- now.toLocaleTimeString() liefert nur die Uhrzeit in lokaler Schreibweise.

Die **setTimeout(AnweisungsString, Millisekunden)** Methode ermöglicht es Anweisung nach Ablauf einer bestimmten Zeit auszuführen.

7.3 Document Object Model (DOM)

Mithilfe des DOM können Sie auf jedes einzelne Element einer Webseite zugreifen. Dazu wird das Objekt als Baummodell aus verschiedenen Arten von Knoten betrachtet. Es existieren verschiedene Möglichkeiten, um auf einen Knoten vom Typ HTML-Tag zuzugreifen:

- document.getElementById(ID-String): liefert eine Referenz auf das Tag zurück, dem über das Attribut id eine spezielle ID zugewiesen wurde.
- document.getElementsByTagName(Tag-String): erhalten Sie eine Referenz auf ein Array aller Elemente, die dem angegebenen HTML-Tag entsprechen.
- document.querySelector(CSS-Selektor) bzw. document.querySelectorAll(CSS-Selektor): Hier können Sie einen beliebigen CSS-Selektor angeben siehe Unterunterabschnitt 6.1.1. Wobei der querySelector das erste Element zurückgibt und der querySelectorAll ein Array aller passender Elemente erzeugt.

7.4 Ajax

Mithilfe der DOM-Technik können HTML-Seiten nach belieben Umgebaut werden. Und mit Hilfe der Ajax-Technik muss die Seite dafür nicht komplett neu geladen werden. Bei Ajax geht es hauptsächlich um:

- Asynchrone HTTP-Anfragen – finden hinter den Kulissen statt, während die Besucher auf der Seite weiterlesen, ein Formular ausfüllen oder ähnliche Tätigkeiten durchführen. Eine Callback-Funktion liest die Serverantwort, sobald sie vorliegt.
- JavaScript – das benötigte HTTP-Anfrageobjekt gehört (unter verschiedenen Namen) zur JavaScript-Bibliothek aller aktuellen Browser.
- XML – die Antwort des Webservers kann (muss aber nicht) verschachteltes XML sein, das Sie ebenso per DOM verarbeiten können wie die Webseite selbst.

Als alternative gibt es auch moderne JS-Bibliotheken bzw. API's we z.B. React.js

Literatur

[1] Brian P. Hogan, HTML5 und CSS3 der Meisterkurs, O'Brian Verlag.

[2] Sascha Kersken, IT-Handbuch für Fachinformatiker*innen, 2022, Rheinwerk Verlag GmbH, ISBN 978-3-8362-8132-4 (E-Book).

[3] `https://css-tricks.com/snippets/css/complete-guide-grid/#aa-grid-area`