[ Historie der Website-Entwicklung 4](#_Toc427153914)

[ Hintergründe zu HTML, CSS, JavaScript 4](#_Toc427153915)

[ Browser als Dreh- und Angelpunkt 4](#_Toc427153916)

[ Logik und Semantik von HTML 4](#_Toc427153917)

[ Zusammenhang von HTML, CSS, JavaScript 4](#_Toc427153918)

[ Dateistrukturierung für Webseiten 4](#_Toc427153919)

[ TextEdit/Texteditoren 4](#_Toc427153920)

[ Adobe Dreamweaver 4](#_Toc427153921)

[ Aufbau einer Auszeichnungssprache 4](#_Toc427153922)

[ Tags, Attribute, Werte, Kommentare 4](#_Toc427153924)

[ Codeformatierung 5](#_Toc427153925)

[ Sonderzeichen und Zeichensatz 5](#_Toc427153926)

[ Dokumenteigenschaften festlegen 5](#_Toc427153927)

[ Blockelemente 5](#_Toc427153928)

[ Bildelemente 6](#_Toc427153929)

[ Textelemente 6](#_Toc427153930)

[ Videoeinbettung und Codecs 6](#_Toc427153931)

[ Videoelement 6](#_Toc427153932)

[ Audioelemente 7](#_Toc427153933)

[ Canvaselement 7](#_Toc427153934)

[ Youtube/Vimeo einbinden 7](#_Toc427153935)

[ Tabellenelemente 8](#_Toc427153936)

[ Bildelemente, Voraussetzungen und Einbindung 8](#_Toc427153937)

[ Überschriften 8](#_Toc427153938)

[ Absätze, Textauszeichnungen, Textausrichtung 9](#_Toc427153939)

[ Listen 9](#_Toc427153940)

[ Zitate 10](#_Toc427153941)

[ Links 10](#_Toc427153942)

[ Tabellen erstellen 11](#_Toc427153943)

[ Kopfzeile definieren 11](#_Toc427153944)

[ Fusszeile Definieren 11](#_Toc427153945)

[ Tabelleninhalte 11](#_Toc427153946)

[ Absolute und relative Pfade 12](#_Toc427153947)

[ Links 12](#_Toc427153948)

[ Ankertags/Waschtags 13](#_Toc427153949)

[ iFrames nutzen 13](#_Toc427153950)

[ Formulare und Abschnitte definieren 13](#_Toc427153951)

[ Eingabefelder für Formulare 14](#_Toc427153952)

[ Textboxes für Formulare 14](#_Toc427153953)

[ Auswahllisten für Formulare 15](#_Toc427153954)

[ Checkboxen für Formulare 15](#_Toc427153955)

[ Radiobuttons 16](#_Toc427153956)

[ Buttons 16](#_Toc427153957)

[ Aufbau der deklarativen Sprache „CSS“ 16](#_Toc427153958)

[ Codenotation 16](#_Toc427153959)

[ CSS referenzieren im HTML 16](#_Toc427153960)

[ CSS Grundselektoren 17](#_Toc427153961)

[\* = es werden sämtliche HTML-Elemente auf der Seite angesprochen 17](#_Toc427153962)

[ Selektoren kommbinieren 17](#_Toc427153963)

[ Pseudoklassen 17](#_Toc427153964)

[ CSS3 media-queries 17](#_Toc427153965)

[ Kombinatoren 17](#_Toc427153966)

[ CSS3-Pseudoklassen und Selektoren 18](#_Toc427153967)

[ Boxmodell 18](#_Toc427153968)

[ Grössen festlegen 18](#_Toc427153969)

[ Farbräume im CSS & CSS3 19](#_Toc427153970)

[ Farben definieren 19](#_Toc427153971)

[ Aussenabstand festlegen 19](#_Toc427153972)

[ Innenabstand festlegen 19](#_Toc427153973)

[ Grössen festlegen 19](#_Toc427153974)

[ Ohne spezielle Positionierung positionieren 19](#_Toc427153975)

[ Absolute Positionierung 20](#_Toc427153976)

[ Relative Positionierung 20](#_Toc427153977)

[ Fixe Positionierung 20](#_Toc427153978)

[ Reihenfolge und Ebenen beeinflussen 20](#_Toc427153979)

[ Floats und Ausrichtungen 20](#_Toc427153980)

[ Rahmen festlegen 20](#_Toc427153981)

[ Übergrosse Inhalte 20](#_Toc427153982)

[ Schriftfamilien definieren 21](#_Toc427153983)

[ Webfonts einsetzen 21](#_Toc427153984)

[ Schriftschnitte definieren 21](#_Toc427153985)

[ Schriftausrichtung definieren 21](#_Toc427153986)

[ Absätze, Umbrüche, Glyphenformatierung 21](#_Toc427153987)

[ Listen und Aufzählungen 22](#_Toc427153988)

[ Farben Festlegen 22](#_Toc427153989)

[ Schlagschatten auf Texte legen mit CSS3 22](#_Toc427153990)

[ Tabellen Formatieren 22](#_Toc427153991)

[ Rahmenmodus umstellen 22](#_Toc427153992)

[ Zeilen formatieren 22](#_Toc427153993)

[ Zellen formatieren 22](#_Toc427153994)

[ Formularfelder Stylen 22](#_Toc427153995)

[ Head-Bereich 23](#_Toc427153996)

[ Body-Bereich 23](#_Toc427153997)

[ Layoutabschnitte definieren 23](#_Toc427153998)

[ Header und Footer mit Inhalten versehen 23](#_Toc427153999)

[ Inhalte Strukturieren 23](#_Toc427154000)

[ CSS Normalization 24](#_Toc427154001)

[ CSS-Grundformatierung 24](#_Toc427154002)

[ Beispiel 24](#_Toc427154003)

Video2brain – Notizen

* Historie der Website-Entwicklung

HTML = Hyper Text Markup Language

HTML5 neuste Version

* Hintergründe zu HTML, CSS, JavaScript

JavaScript wird benutzt um Animationen und interaktive Aktionen zu ermöglichen

ECMAScript Norm bildet Grundgerüst für zahlreiche neue Versionen

* Browser als Dreh- und Angelpunkt

Websites werden von Browser aus HTML, CSS und JavaScript zusammen gerendert

* Logik und Semantik von HTML

Semantik = Gliederung/Beschreibung von Inhalten

Fragen: 1. Was will ich eigentlich kommunizieren?

2. Welche Art von Information habe ich hier?

3. Wie kann ich meine Information am besten strukturieren?

4. Welche Bedeutung oder Funktion hat mein Inhalt?

5. Welche HTML-Elemente beschreiben den Inhalt am passendsten?

* Zusammenhang von HTML, CSS, JavaScript

HTML braucht die Grammatikregeln von SGMl/XML

Jedoch kann HTML zusammen mit SGML schon eine Website bilden, JavaScript und CSS sind Ergänzungen

* Dateistrukturierung für Webseiten

Index.html wird immer zuerst vom Browser aufgerufen und bildet die Grundstruktur der Website

Das CSS Hauptdokument nennt man stylesheet.css oder style.css

* TextEdit/Texteditoren

Man könnte HTML auch im Editor verwenden, ist aber mühseliger

* Adobe Dreamweaver

Ist ein Programm, welches einiges erleichtert

* Aufbau einer Auszeichnungssprache

Man beginnt ein einzelnes Element mit <tag> und schliesst es mit </tag>

Man kann auch Elemente verschachteln <tag><tag>Inhalt</tag></tag>

Gelb = Inhalt der äusseren tags

* Tags, Attribute, Werte, Kommentare

Tags bestehen aus:

-Tagname (z.B. „img“)

-Attributen

-Werten

Attribute werden gleich im Anschluss an den Tagname geschrieben: <tagname attribut1=“ „>

Werte kann man einem Element zufügen z.B. der Name eines Bildes dem Bild!!

Kommentare kann man so einfügen: <!-- <img title=“Bild Titel“> mein Kommentar -->

* Codeformatierung

Elemente am besten übersichtlich aufgliedern

Ebenen setzen d.h. vor untergegliederten Elementen ein Tab

* Sonderzeichen und Zeichensatz

Sonderzeichen können mit Austauschformeln Kodiert werden um diese auch in anderen Sprachen lesen zu können

Bei HTML-Entitäten muss man einfach amp; dahinter setzen (z.B. & = &amp;)

Bei Unicode muss man # und die dazugehörige Zahl eintragen (z.B. & = &#38;)

Um die Kodierungsart festzulegen, muss man <meta charset=“utf-8“ /> in das Header Element einbinden

Hilfe: <https://de.wikipedia.org/wiki/Hilfe:Sonderzeichenreferenz>

* Dokumenteigenschaften festlegen

Um den Dokumententyp als HTML festzulegen, muss man nur <!DOCTYPE HTML> zuoberst vor allem anderen eintragen

Das <html></html> nicht vergessen

Titel= <title></title>

Um eine Browser-Suchergebnis-Zusammenfassung der Website zu erstellen, muss man <meta name=“description“ content=“Zusammenfassung“> eintippen

Den Inhalt selber kommt in den <body></body>

* Blockelemente

<body> hier kommt der grundsätzliche Inhalt rein

<article> damit kann man Inhalt hervor heben, denn es wird durch zwei Umbrüche von den anderen Elementen getrennt

<aside> der Inhalt wird in eine Art seitenleiste dargestellt!!

<section> seitenabschnitt

<header> header

<footer> footer

<h1-h6> überschriften(zahlen zeigen grösse auf)

<hgroup> eine Gruppe von Überschriften!!

<hr> Trennlinie

<pre> vor Formatierter Text(kleiner als der Rest)

<blockquote> zitat

<ol> <ul> <li> <dl> <dt> <dd> listenpunkte

<figure> ?

<table> eine Tabelle

<form> ein Formular

<br> Zeilenumbruch

<strong> **Fett**

* Bildelemente

Ein Bild muss in das element <img> eingebunden werden

Dazu macht man: <img src=“bildlink“ width=“breite in px“ height=“bildhöhie in px“>

Um Beschreibungen zu einem Bild hinzuzufügen benutzt man alt=“Beschreibung“

Um dem Bild einen Titel hinzuzufügen, kann man noch title=“Titel“ anfügen

Inline Content=es entsteht kein abstand und man kann das Element überall platzieren

Blockelement=wird separat eingestuft und semantisch korrekt angeordnet

Das img Element ist normalerweise ein inline-element und kann mit <figure><img></figure> zu einem Blockelement gemacht werden

Wenn man eine zusätzliche Beschreibung hinzufügen will, muss man zwischen dem img Element und dem abschliessenden figure Element einfach <figcaption>Beschreibung</figure> einfügen (wird unter dem Bild angezeigt)

* Textelemente

Damit kann man Text auszeichnen

<p> steht für einen Absatz (der Text in diesem Element wird auf einem neuen absatz dargestellt)

<del> Text wird als gelöscht angezeigt (Text wird durchgestrichen)

<ins> Text wird als hinzugefügt angezeigt (Text wird unterstrichen)

<a> einen Link (um den Pfad des Links festzulegen muss man href=“link“ im ersten tag hinterlegen

<sub> der Text ist tiefer als der normale Text: so etwa

<sup> der Text ist höher als der normale Text: so etwa

* Videoeinbettung und Codecs

Videoeinbettung ist im HTML5 ohne Flash möglich

Je nach Videoformat können verschiedene Browser, die Videos nicht anzeigen

Mit Chrome sollte man den grössten Teil aller Videos anzeigen können

* Videoelement

Das Element <video></video> benutzt man zur Videoeinbettung und ist ein Blockelement

Das Videoelement funktioniert genau gleich, wie das Bildelement:

<video src=“videolink“ title=“Videotitel“ width=“videobreite“ heigth=“videohöhe“>

Doch damit wird nur das Video eingefügt und kann nur umständlich bedient werden

Um dies zu ändern kann man controls als Attribut hinter der Höhe, usw einfügen

Um einzustellen, dass das Video beim Öffnen der Seite automatisch abgespielt wird, kann man das Attribut autoplay einfügen

Das Attribut preload bewirkt, dass das Video zusammen mit der Seite vorgeladen wird. Man benötigt einen Wert: auto = wird jedes Mal beim Betreten der seite vollständig geladen

metadata = nur das oberflächliche wird geladen

none = wird gar nicht geladen !!

Um ein Thumbnail hinzuzufügen, kann man einfach das Attribut poster verwenden: poster=“bildlink“ (funktioniert nicht, wenn autoplay auch vorhanden ist)

Um das Video für fast alle Browser sichtbar und abspielbar zu machen muss man verschiedene Versionen von Video hinterlegen, wie z.B. eine .mp4 Version für Safari und Chrome und eine .ogg Version für Explorer und Opera.

Dazu kann man folgendes machen:

Das scr=“Videolink“ Attribut löschen und den Rest wie folgt editieren:

<video title=“Videotitel“ width=“videobreite“ heigth=“videohöhe“>

<source scr=“videolink1.mp4“ type=“video/mp4“>

<source scr=“videolink2.ogg“ type=“video/ogg“>

</video>

Jetzt benutzen die verschiedenen Browser jeweils die Datei, die ihnen passt

Ausserdem kann man für Browser, welche keine der beiden Formate unterstützt, noch einen Text hinzufügen, welcher dann angezeigt wird:

Einfach <p>text/link</a> zu den beiden source codes schreiben

* Audioelemente

Das Audioelement ist ebenfalls ein Blockelement

Ausserdem funktioniert das Audioelement gleich, wie das Videoelement:

<audio scr=“audiolink“ title=“audiotitel“>

Allerdings wird dann nichts auf der Website angezeigt und man sollte noch das Attribute console hinzufügen, sodass der user die Musik steuern kann

Wenn die Musik als Hintergrund Musik verwendet wird sollte man noch die Attribute autoplay, preload und loop hinzufügen.

Loop macht einfach, dass der Song in Endlosschleife abgespielt wird

Doch auch bei der Audio hat man das Browser Problem und kann eigentlich wieder dasselbe, wie beim Video machen:

<audio title=“audiotitel“ autoplay loop>

<source scr=“audiolink1.mp3“ type=“audio/mp3“>

<source scr=“audiolink2.ogg“ type=“audio/ogg“>

<source scr=“audiolink3.wav“ type=“audio/wav“>

</audio>

* Canvaselement

Damit kann man eine Bitmap anzeigen

Dieses Element ist sehr kompliziert und man braucht gute Kenntnisse im JavaScript

* Youtube/Vimeo einbinden

Ein YouTubevideo einzubinden ist eigentlich einfach, man muss nur auf das Youtubevideo selber und da über Teilen auf Einbetten klicken. Hier sollte nun ein iFrames-Code angezeigt werden. Weiter unten kann man Grösse und diverse kleinere Einstellungen unternehmen. Der iFrame-Code kann man nun ins HTML kopieren und schon wird das Video angezeigt.

Bei Vimeo dasselbe: nur heisst der einbetten-button embed und ist im Video-Interface selber. Es öffnet sich ein Fenster mit dem iFrame-Code und ein paar Einstellungen. Nun den Code kopieren und im HTML einfügen und fertig

* Tabellenelemente

<table> das Hauptelement 8die Tabelle selber)

<caption> Tabellentitel

<thead> Kopfzeile der Tabelle

<tbody> der Tabellenkörper (hauptcontent)

<tfoot> Fusszeile der Tabelle

<tr> tabellenzeile

<th> überschrift in der Kopfzeile

* Bildelemente, Voraussetzungen und Einbindung

Bilder sollte, von der Datenmenge her, relativ klein sein, da es die Ladezeit der Webseite verlängert.

Um ein Bild zu verkleinern, muss man es im Photoshop öffnen und dann als Webelement speichern.

Man kann die Grösse verringern, indem man die Qualität senkt, nicht verwendete Farben entfernt, die Grösse (pixel) ändert oder ein Häkchen vor Progressiv machen.

* Überschriften

Es gibt insgesamt 7 tags für Überschriften:

<hgroup> darin kann man alle anderen Überschriften verschachteln und anzeigen, dass sie zusammen gehören

<h1> grösste Überschrift

<h2> …

<h3> …

<h4> …

<h5> …

<h6> kleinste Überschrift

Normalerweise folgt auf eine Überschrift(und Überschriftsgruppe) ein Absatz (<p>)

Beispiele:

<hgroup>

<h1>Titel</h1>

<h2>Untertitel</h2>

</hgroup>

<p>Text</p>

<h1>Titel</h1>

<p>Text</p>

<h2>Zwischentitel (im Text selber)</h2>

<p>Text2</p>

H1 ist eine Überschrift der 1. Ebene, usw

Überschriften sind Inline-Elemente. Um sie zu Blockelemente zu machen, muss man die einzelnen Gruppierungen oder ebenen in ein <section> Element setzen:

<h1>Titel</h1>

<p>Text</p>

<section>

<h2>Zwischentitel (im Text selber)</h2>

<p>Text2</p>

</section>

Nun gilt die H2-Ebene als Blockelement

* Absätze, Textauszeichnungen, Textausrichtung

Man kann Texte mit verschiedenen tags innerhalb von anderen Elementen formatieren

Zum Beispiel kann man mitten in einem Text, einzelne Wörter zwischen ein <strong>**Fett**</strong>setzen um diese hervorzuheben

Für kursiven Text benutzt man <em>

<small> für unwichtigen Text

Um einem Text oder nur einem Teil davon eine Klasse zuzuweisen muss man <span class=“Klasse“></span> eingeben und kann so Klassen verteilen

Wenn man einem Wort/Element/wasauchimmer eine Klasse zugeteilt hat, kann man sich später im CSS auf diese Klasse beziehen:

.klasse {

Hier kommt der Befehl der diesem Wort/wasauchimmer zugeteilt werden soll

}

Man kann auch Abkürzungen kennzeichnen und das volle Wort anzeigen lassen, wenn man mit der Maus drüber fährt:

<abbr title="zum Beispiel">z.B.</abbr>

<del> zum durchstreichen von Text stellen

Um zu kennzeichnen, dass ein Text durch einen Neuen ersetzt worden ist macht man das:

<section>  
 <s>alter text</s>  
 <ins>neuer Text</ins>  
</section>

<s> streicht den Text eigentlich auch nur durch und<ins> unterstreicht ihn

Man kann auch Zeitangaben markieren:

<time datetime="Datum (speziell)">Angezeigtes Datum</time>!!

Falls man irgendwo mal auf der Website code darstellen will, ohne dass er vom Browser benutzt wird, muss man ihn einfach zwischen zwei <code> tags setzen

* Listen

Die Hauptelemente von Listen sind <ul> oder <ol>

Diese zwei kennzeichnen erst mal, wo überhaupt die Liste hin soll. Die Liste selber wird dann von Listenpunkten - <li> - erstellt. Listenpunkte werden zwischen die zwei ul/ol tags geschrieben und beinhalten den gelisteten Text.

Das sieht etwa so aus:

<section>  
 <ul>  
 <li>eins</li>  
 <li>zwei</li>  
 <li>drei</li>  
 <li>vier</li>  
 </ul>  
</section>

Der Unterschied zwischen ul und ol ist, dass ul einfach nur punkte vor den Txt macht und ol den Text nummeriert

Man auch definitionslisten machen, mit denen man eine Abkürzung oder ähnliches und dessen Definition oder Erklärung angeben kann und dass wird dann so dargestellt:

HTML

Hyper Text Markup Language

LoL

League of Legends

Der Code dazu:

<dl>  
 <dt>HTML</dt>  
 <dd>Hyper Text Markup Language</dd>  
 <dt>LoL</dt>  
 <dd>League of Legends</dd>  
</dl>

* Zitate

Es gibt Kurzzitate, welche als Inlineelmente gelten und Blockzitate, die als Blockelemente gelten

Fangen wir mit dem Kurzzitat <q> an:

<p><q>Hallo liebe Welt</q>, sagte er voller Freude</p>

Das Hallo liebe Welt wird vom Browser automatisch in diese ““ gesetzt

Blockzitate werden per <blockquote> gemacht:

<blockquote>  
 <p>Text1</p>  
 <p>Text2</p>  
 <cite>Quellenangabe</cite>  
</blockquote>

Die Quellenangabe wird Kursiv dargestellt

* Links

Um einem Wort/Text einen Link zu zuteilen, muss man es so machen:

<a href="http://www.internezzo.ch">internezzo</a>

Bei Links aus dem Internet das http(s):// nicht vergessen

Um eine e-mailadresse zu verlinken, die bei Betätigung auch gleich das Mail Programm öffnet setzt man einfach <mailto:email.adresse@email.com> als link

* Tabellen erstellen

Tabellen sind kompliziert

* Kopfzeile definieren

Zuerst muss man erst mal das Grundgerüst bilden, indem man den tag <tables> setzt. Dieser ist ein Blockelement

Diesen tag selber kann man auch wieder unterteilen in Head, Body und Foot.

Wir fangen mit dem Head an, dieser wird vom <thead> generiert.

Im Head kann man nun eine zeile erstellen <tr> und in der mehrere spalten <th>

Das sieht etwa so aus:

<table>  
 <thead>  
 <tr>  
 <th>Spalte 1</th>  
 <th>Spalte 2</th>  
 <th>Spalte 3</th>  
 </tr>  
 </thead>  
</table>

* Fusszeile Definieren

Die Fusszeile kommt gleich hinter dem Head mit dem code <tfoot>

Die Fusszeile funktioniert genau gleich, wie die Kopfzeile, ist aber unter der Tabelle

In der Fusszeile muss es genau gleich viele Spalte haben, wie in der Kopfzeile. Die Spalten in der Fusszeile werden mit <td> gekennzeichnet

<table>  
 <thead>  
 <tr>  
 <th>Spalte1</th>  
 <th>Spalte2</th>  
 <th>Spalte3</th>  
 </tr>  
 </thead>  
 <tfoot>  
 <tr>  
 <td>Foot1</td>  
 <td>Foot2</td>  
 <td>Foot3</td>  
 </tr>  
 </tfoot>  
</table>

* Tabelleninhalte

Den Hauptcontent der Tabelle kommt in den <tbody> Bereich, welcher wieder ähnlich aufgebaut ist, wie die anderen zwei Untersektionen

Es gibt jedoch einen entscheidenden Unterschied, denn man kann im body so viele Zeilen unterbringen, wie man will

Jedoch die Spaltenanzahl muss wieder gleich sein und auch erwähnt werden, obwohl sie keinen Inhalt haben.

Der tag für die Spalten ist wieder <td>

Der Code sieht dann etwa so aus:

<table>  
 <thead>  
 <tr>  
 <th>Spalte 1</th>  
 <th>Spalte 2</th>  
 <th>Spalte 3</th>  
 </tr>  
 </thead>  
 <tfoot>  
 <tr>  
 <td>Foot1</td>  
 <td>Foot2</td>  
 <td>Foot3</td>  
 </tr>  
 </tfoot>  
 <tbody>  
 <tr>  
 <td>body1a</td>  
 <td>body2a</td>  
 <td>body3a</td>  
 </tr>  
 <tr>  
 <td>body1b</td>  
 <td>body2b</td>  
 <td>body3b</td>  
 </tr>  
 </tbody>  
</table>

* Absolute und relative Pfade

Ein absoluter Pfad ist, wenn der link über das Laufwerk, die verschiedenen Ordner, bis zur eigentlichen Datei angegeben werden muss

Bei einem relativen Pfad kann man, wenn die Datei im gleichen Ordner ist einfach den Namen der Datei angeben und so kann diese Datei eigentlich im ganzen Ordner umhergeschoben werden

Man kann aber auch ../dateiname als link angeben und kann dann die Datei sich auch irgendwo im Ordner, wo der Order von der HTML-Datei sich befindet, befinden

* Links

Links zur Navigation auf der Webseite selber

Für die Navigation benutzt man den code <nav>

Innerhalb des <nav> macht man eine <ul> liste und ortnet jedem <li> einen Link zu

<nav>  
 <ul>  
 <li><a href="index.html">Index</a></li>  
 <li><a href="seite2.html">Seite 2</a></li>  
 </ul>  
</nav>

Bei den links kann man auch wieder verschiedene Attribute vergeben

Title: mit diesem Attribut kann man einstellen, was in dem kleinen mini Fenster erscheint, wenn man über den Link fährt

Target: damit kann man bestimmen, ob eine neue Seite geöffnet werden soll (target=“\_blank“)

* Ankertags/Waschtags

Man kann auch links zu unteren Bereichen von Seiten machen, sogenannte Ankertags

Dazu teilt man diesen Abschnitten einen Namen zu:

<a name="Ankertag1"><a href="http://www.internezzo.ch">internezzo</a></a>

Weiter oben kann man dann einen Link zu diesem Ankertag machen:

<a href="#Ankertag1">Zum internezzo-link</a>

* iFrames nutzen

Mit iFrame, kann man andere Websites in seine eigene Website einbinden

Der Haupttag hier ist <iframe>

Hier habe ich mal die internezzo-Website eingebunden

<section>  
 <iframe src="http://www.internezzo.ch" width="1345px" height="845px">  
 </iframe>  
</section>

Mit den Attributen width und height kann man die Breite und Höhe des Fensters bestimmen

Das Attribut seamless soll eine bessere Einbindung bewirken, funktioniert aber kaum

Sandbox ist das Attribut womit man den Internetzugang innerhalb dieser Einbindung beschränken kann dazu gehören auch einige Bezeichnungen:

sandbox=“allow-forms“ keine Formulare

sandbox=“allow-same-origin“ nur diese Domain

sandbox=“allow-scripts“ keine JavaScripte und ähnliches

sandbox=“allow-top-navigation“ navigation innerhalb dieser Seite erlauben

* Formulare und Abschnitte definieren

Formulare kann man verwenden, um Daten von Usern zu bekommen und um Suchfelder zu erstellen

Hautptag: <form> Blockelement

Attribute: id, action

Die id eines formulars muss immer festgelegt werden: id=“formular1“

Wo man nach dem Abschicken hingeleitet wird Action=“seite2.html“

Was soll mit den Daten des Formulars gemacht werden? Method=“delete, get, post, put“

Am besten benutzt man post, weil dann kriegt man diese sicher

Mit <fieldset> kann man das Formular in Felder einteilen

Diesen Feldern kann man mit title=“Titel“ eine Überschrift geben (Attribut)

So hat man schon mal die Grundstruktur:

<form id="coment" method="post">  
 <fieldset title="Commentsection">  
 </fieldset>  
</form>

* Eingabefelder für Formulare

Für die Beschreibung benutzt man <label>

Das Eingabefeld wird mit <input> gekennzeichnet (singletag)

Für das input-element gibt es mehrere Attribute:

Type: man bestimmt den typ (text, button ,e-mail, tel, number, date, url, search, passwort, usw)

Id: auch Eingabefelder brauchen ids

Required: dieses Feld muss ausgefüllt werden

Placeholder: das steht im Eingabefeld, wenn es noch unbenutzt ist

Min: minimalzahl hinzufügen

Max: maximalzahl hinzufügen

Um jetzt das label mit dem input zu verbinden, benutzt man das Attribut for im label und trägt die id, des inputs ein

Beispiel:

<form id="comment" method="post">  
 <fieldset title="Commentsection">  
 <label id="name" title="Name" for="name">Name</label>  
 <input id="name" type="text" required>  
 </fieldset>  
 <fieldset>  
 <label id="date" for="date" title="Datum">Datum</label>  
 <input id="date" type="date">  
 </fieldset>  
 <fieldset>  
 <label id="comment" for="comment" title="Kommanter zur Seite">Kommentar zur Seite</label>  
 <input id="comment" type="text" required>  
 </fieldset>

* Textboxes für Formulare

Um richtige Kommentarfelder zu erstellen braucht es ein spezielles Element <textarea>

Man braucht hier auch wieder eine ID um es mit einem label zu verknüpfen

Zusätzlich kann man mit dem Attribut rows die Zeilenanzahl angeben (rows=“20“)

Man kann auch die Zeichenanzahl pro Zeile festlegen mit dem attribut cols=“250“

In neueren Browsern kann man die Textfläche beliebig verändern, auch als user

Um diese Eigenschaft zu verändern benutzt man das Attribut draggable:

Auto = wird automatisch gemacht

False = wird nicht gemacht

True = wird immer gemacht

<label id="comment" for="comment" title="Kommanter zur Seite">Kommentar zur Seite</label>  
<textarea id="comment" rows="20" cols="100" required draggable="false"></textarea>

* Auswahllisten für Formulare

Selectboxen macht man mit <select> (wieder mit label+id)

Nun ist es aber so, dass select wie eine Listenüberschrift gilt und daher zwischen den beiden tags noch das Element option haben muss um eine Auswahlliste zu erstellen

Dieses Element braucht auch wieder eine ID und einen Wert, welcher mit value=“Wert“ angegeben wird und man muss auch den Namen dieser Option angeben, indem man ihn zwischen die option tags schreibt

Manchmal hat man viele optionen und sollte daher verschiedene <optgroup>‘s um die einzelnen options legen. Diese optgroups sind immer noch innerhalb des selects

Um mehrere Auswahlen zu ermöglichen, muss man dem select das Attribut multiple hinzufügen (ist aber mühselig und nicht empfehlenswert)

<form id="comment" method="post">  
 <fieldset title="Commentsection">  
 <label id="name" title="Name" for="name">Name</label>  
 <input id="name" type="text" required>  
 </fieldset>  
 <fieldset>  
 <label id="auswertung" for="auswertung" title="Wie finden Sie diese Seite?">Wie finden Sie diese Seite?</label>  
 <select id="auswertung">  
 <option id="auswertung" value="bad">Schlecht</option>  
 <option id="auswertung" value="good">Gut</option>  
 </select>  
 </fieldset>  
 <fieldset>  
 <label id="date" for="date" title="Datum">Datum</label>  
 <input id="date" type="date">  
 </fieldset>  
 <fieldset>  
 <label id="comment" for="comment" title="Kommanter zur Seite">Kommentar zur Seite</label>  
 <textarea id="comment" rows="20" cols="100" required draggable="false"></textarea>  
 </fieldset>  
</form>

* Checkboxen für Formulare

Um Checkboxen darzustellen, muss man das input-Attribut checkbox benutzen

Das Attribut checked bewirkt, dass die Box automatisch ausgewählt ist

Um den Namen, hinter den Checkboxen festzulegen muss man nach den input noch dein <br> schreiben und den Namen dazwischen hinterlegen

<fieldset>  
 <label id="like" for="like">Welche Elemente gefallen Ihnen?</label>  
 <input type="checkbox" id="like" name="like" value="Formular">Formular<br>  
 <input type="checkbox" id="like" name="like" value="Suchfeld">Suchfeld<br>  
 <input type="checkbox" id="like" name="like" value="Youtubevideo">Youtubevideo<br>  
</fieldset>

* Radiobuttons

Radiobuttons sind das gleich, wie Checkboxen, man kann aber nur eine der Optionen wählen

Ausserdem muss der typ anstatt checkbox zu radio geändert werden

Der Name der Radiobuttons muss unter allen gleich sein

* Buttons

Für einen „Abschicken-Button“ kann man einfach in einem Inputelement den Typ submit auswählen (reset-Button ist genau gleich einfach mit reset)

Das value-Attribut bestimmt die Aufschrift, des Buttons

Man sollte auch hier eine ID vergeben

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

Um die grösse der Buttons zu verändern, muss man zuerst die Eigenschaft diplay: block; zuweisen

* Aufbau der deklarativen Sprache „CSS“

CSS = Cscading Style Sheets

Damit kann man die HTML –Elemente formatieren

2.1 ist die aktuelle version

CSS wird in Regeln notiert, welche aus Selektor, Eigenschaft und Werten formatiert

Selektor {

Eigenschaft: wert;

}

* Codenotation

Kommentare macht man so:

/\* Dies ist ein Kommentar \*/

Selektor {

Eigenschaft: wert;

}

* CSS referenzieren im HTML

Um CSS mit HTML zu verbinden muss man einfach nur den folgenden Code im HTML einbetten

<link rel="stylesheet" type="text/css" href="stylesheet.css" />

Stylesheet.css ist der Name meiner CSS Datei

Um das stylesheet nur teilweise oder unter bestimmten Bedingungen zu laden, kann man noch das Attribut media hinzufügen:

All = es wird immer alles geladen

Print = es wird nur beim Drucken geladen

Screen = das Gegenteil von Print

* CSS Grundselektoren

\* = es werden sämtliche HTML-Elemente auf der Seite angesprochen

Man kann auch alle div, section, usw –Elemente ansprechen indem man deren Namen als selektoren verwendet

Mit dem Attribut class kann man im HTML den verschieden Elementen Klassen zuweisen

Diese Klassen kann man mit .klasse ansprechen

Mit #ID kann man id’s ansprechen

* Selektoren kommbinieren

Um selektoren zu kombinieren, kann man die einzelnen selektoren auf eine Zeile setzen und mit Kommas trennen.

* Pseudoklassen

Hover: damit kann man Hovereffekte ansprechen

z.B. bei einem Link gibt es einen Hovereffekt. Um nun diesen Effekt anzusprechen, muss man den selektor a:hover nehmen und kann so die Farbe der Schrift verändern, usw

Pseudoklassen werden per :pseudoklasse dergestellt

Links die schon besucht worden sind, kann man auch ansprechen. Und zwar mit :vistited

Man kann auch mit :focus ein aktiviertes Feld ansprechen

Um das erste Element seiner Art anzusprechen, kann man :first\_child verwenden

:last-child spricht das letzte Element an

* CSS3 media-queries

Man kann seine Website auch für Smartphones optimieren, indem man ein weiteres stylesheet einfügt.

Doch zuerst sollte man eine Anpassung im head-Bereich machen und folgenden Code einfügen:

<meta name="viewport" content="width=device-width, target-dens i typd i=device-dp, intial-scale">

Bei den verschiedenen Stylesheets muss man einfach immer nur den Pfad und das media verändern

Beispiel

<link rel="stylesheet" type="text/css" href="stylesheet.css" media=“only screen and (min-device-width:320px) and (max-device-width:480px)”>

Bei den verschiedenen Geräten muss man einfach immer den Link und die Pixelangaben ändern

* Kombinatoren

Wenn man Elemente innerhalb von anderen Elementen ansprechen will, kann man zuerst das äussere Element ansprechen und dann nach einem Lehrschlag die weiteren Elemente

Um Elemente direkt innerhalb eines äusseren Elements anzusprechen ersetz man einfach nur den Leerschlag durch ein >

Um Elemente, die in der gleichen Ebene sind anzusprechen, setzt man einfach ein + zwischen die zwei Elemente

* CSS3-Pseudoklassen und Selektoren

Man kann auch Selektore anhand von Attributen festlegen z.B. input[type=“text“]

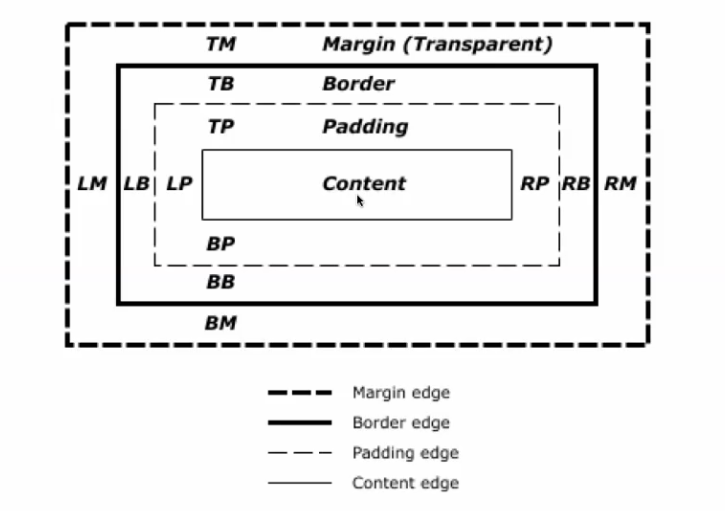
Wenn man ein Element, wessen Attribut in der Beschreibung mit bestimmten Buchstaben anfängt, sucht, kann man ein‘ (🡨 das häkchen oben) zwischen dem Element und dem [ machen

Wenn die Beschreibung mit bestimmten Buchstaben aufhören soll, kann man ein :after hinter das ] setzen und das‘ durch ein $ ersetzen

Um einfach ein Buchstabenfolge in einer Beschreibung zu selektionieren, kann man einfach ein \* nehmen und in den Platz von $/‘ setzen (ohne :after)

Mit dem Anhang :nth-child(2n) wird jedes zweite Element, wie ,das an das der Anhang gehängt worden ist. (2n = jedes zweite, 3n = jedes dritte)

* Boxmodell



Das Padding übernimmt die gleiche Hintergrundfarbe, wie der Content

* Grössen festlegen

Man kann den verschiedenen Elementen, diverse Grössen hinzufügen, um sicherzustellen, dass es richtig angezeigt wird, wie man will

Am besten macht man das mit der Eigenschaft width und height

Beiden kann man einfach den Wert auto geben, was aber nicht empfehlenswert ist, da es dann der Browser entscheidet

Man kann den Wert auch in Prozent angeben, also z.B. 80% des überstehenden Elements

Allerdings ist das keine feste grosse und es kann alles verschieben

Daher sollte man die Grösse in Pixel angeben (z. B. 800px)

Die Rahmengrösse kann man, wie die eigentliche grösse auch per Eigenschaft+Wert definieren. Jedoch ist es so, dass es nicht mehr Breite und Höhe gibt, sondern oben, rechts, unten, links d.h.

Margin: oben, rechts, unten, links

* Farbräume im CSS & CSS3

Man kann Farben in CSS durch ihre Namen oder einfach die Farbpalette benutzen

* Farben definieren

Hex-Werte bestehen aus # und verschieden zahlen/ziffern

Man kann Hex-Werte, welche aus 6 gleichen Zeichen bestehen auch abkürzen

Hex-Werte werden von allen Browser unterstützt

RGB-Werte sind relativ kompliziert

Sie bestehen aus: rgb(zahl1, zahl2, zahl3)

Wenn man dem rgb ein a hinzufügt (rgba) wird die Schrif transparent (wieviel kommt auf Zahl 4 an) 1 = 100% sichtbar 0.5 = 50%

Man kann auch mehrere Werte stapeln, um die Browserprobleme zu umgehen

* Aussenabstand festlegen

Man kann die verschiedenen Blockelemente verschieben, indem man das Margin verändert.

z.B. man will ein Element nach unten verschieben:

margin: 0 0 5px 0; in diesem Fall ist der Abstand unten nun grösser

margin: oben rechts unten links;

Wenn man einfach nur den Abstand von einer Seite verändern will kann man auch margin-left: 5px; eingeben und den linken abstand festlegen (left, bottom, right, usw)

* Innenabstand festlegen

Mit padding: 10px 10px 10px 10px; kann man den Abstand, der Box zu den Elementen innerhalb dieser Box

Wenn man nur einen Wert angibt, gilt der für alle vier Seiten

* Grössen festlegen

Das padding erweitert die Grösse des eigentlichen Elements, sodass das Element eigentlich immer gleich gross bleibt

Um den kompletten Rand zu entfernen, kann man mit dem Universalselektor \* das padding und margin auf 0 setzen (nicht empfehlenswert)

Man kann auch eine min-width und eine max-width

* Ohne spezielle Positionierung positionieren

Wenn man ein Element nach ganz rechts schieben will muss man folgendes eingeben:

#element { margin: auto 0 auto auto; }

Um Elemente zu zentrieren, muss man einfach in beiden Richtungen auto angeben

Elemente kann man auch überschneiden lassen, und zwar mit minuswerten

* Absolute Positionierung

Ein absolut Positioniertes Element kann von anderen Elementen nicht beeinflusst werden

Man benutzt position: absolute;

Top: 10px;

Left: 10px;

Bottom: 10px;

Right: 10px;

Um ein Elment abhängig von einem anderen zu machen, benutzt man wieder das margin

* Relative Positionierung

Hier nimmt man den Wert relative

Das Element wird nur Visuell verschoben, gilt aber immer noch als am gleichen punkt

* Fixe Positionierung

Ein solches Element, bleibt immer im Sichtfeld, egal wohin man scrollt

Der Wert fixed wird benutzt

* Reihenfolge und Ebenen beeinflussen

Mit der Eigenschaft z-index kann man die Reihenfolge, der Elemente ändern

Der Höchste Wert ist immer oben

* Floats und Ausrichtungen

Ein Blockelement beansprucht automatisch eine ganze Zeile, welche die gleiche Höhe hat, wie das Element

Diesen effekt kann man mit floats ausstellen

Float:left das Element steht nachher links

Float:right das Element steht nachher rechts

Um einem Floateffekt aufzuheben, benutzt man clear:both

* Rahmen festlegen

Man kann Rahmen verwenden, um Elemente hervorzuheben oder zu betonen

Die Eigenschaft heisst border

Der Aufbau ist wie folgt:

Border: dick/dünn gestrichelt/durchgezogen farbe

Solid=richtigen rahme

Dashed=gestrichelt

Dotted=gepunktet

Thin=dünn

Abgerundete ecken gehen mit border-radius

Das Element wird um den Rahmen vergrössert

* Übergrosse Inhalte

Wenn das Element den Inhaltliche Text nicht anzeigen kann, weil man es mit CSS zu klein gemacht hat, kann man eine Spezielle Eigenschaft verwenden: overflow

Bei dem Wert: scroll wird ein Scrollbalken angezeigt

Auto: es wird nur bei Überfluss ein scrollbalken angezeigt

Hidden: man sieht nur den halben text

* Schriftfamilien definieren

Da nicht jeder User alle Schriftarten installiert hat, kann man auch hier verschiedene Schriftarten angeben, um alle User abzudecken

Dazu gibt man font-family: primäre Schriftart, 1. Sekundäre Schriftart, 2. Sekundäre Schriftart;

Wenn eine Schriftart aus zwei Wörtern besteht, kann man diese in \*\* setzen

* Webfonts einsetzen

Man kann auch eigene Schriftarten laden. Allerdings braucht man dafür wieder die verschiedenen Formate, damit kein Browser ein Problem hat

@font-facde {

Font-family: \*Schriftname\*;

Src: url(„schriftart.eot“); ist für den alten Explorer

Src: url(„schrfitart.eot\*iefix“); ist für neuere Explorer

Format(„embedded-opentype“),

Src: url(„schriftart.woff“) für die neuste Browser

Format(„woff“);

Scr: url(„schriftart.ttf“) für die restlichen Browser

Format(„truetype“);

Nun kann man anstatt irgendwie Calibri oder Arial anzugeben, auch einfach seinen eigenen Schriftnamen verwenden

* Schriftschnitte definieren

Mit font-weigth kann man die dicke, der Schrift festlegen

Man kann dies mit Zahlen angeben, sollte aber mit bold., normal, usw angegeben werden

Font-style kann das mit dem Wert italic die Schrift kursiv darstellen

* Schriftausrichtung definieren

Um Text linksbündig, zentriert, rechtsbündig oder Blocksatz zu formatieren, benutzt man den befehlt text-allign: center(zentriert), left(linksbündig), rigth(rechtsbündig), justify(blocksatz);

Um die Silbentrennung zu aktivieren benutzt man folgendes:

-webkit-hyphens: auto;

-moz-hyphens: auto;

-o-hyphens: auto;

Hyphens: auto;

* Absätze, Umbrüche, Glyphenformatierung

Mit line-height kann man den Zeilenabstand verändern (1.25 ist Standart)

Nicht vergessen: 1.5 und nicht 1,5!

Den Abstand, zwischen den einzelnen Elementen, kann man wieder mit margin verändern

Wenn man nur den ersten Buchstaben, eines Textes verändern will, kann man den Pseudo-Selektor :first-letter verwenden

* Listen und Aufzählungen

Mit der Eigenschaft list-style kann man bei einer Ungeordneten Liste die Werte:

circle für einen Kreis anstatt Punkt

disc für den Satndart Kreis

square für ein Rechteck

Mit dem padding kann man die Auflistung nach links verschieben

* Farben Festlegen

Schriftfarben legt man mit color: farbe; fest

* Schlagschatten auf Texte legen mit CSS3

Hier benutzt man einfach text-shadow: abstand1 abstand2 blurr farbe;

Damit kann man noch einen Glow-effekt machen:

text-shadow: 0 0 10px rgba(255,255,255,1) , 0 0 20px rgba(255,255,255,1) , 0 0 30px rgba(255,255,255,1) , 0 0 40px #000000, 0 0 70px #000000, 0 0 80px #ff00de , 0 0 100px #ff00de ;  
-webkit-transition: all 200ms cubic-bezier(0.42, 0, 0.58, 1);  
-moz-transition: all 200ms cubic-bezier(0.42, 0, 0.58, 1);  
-o-transition: all 200ms cubic-bezier(0.42, 0, 0.58, 1);  
transition: all 200ms cubic-bezier(0.42, 0, 0.58, 1);

Man kann auch einen Effekt erzeug, dass es aussieht wie gestanzt:

Text-shadow: -1px -1px 0px Farbe;

* Tabellen Formatieren

Um die gesamte Tabelle zu Formatieren, muss man den Selektor table verwenden

Den Rahmen macht man mit der Eigenschaft border

In diesem Selektor kann man eigentlich die grundsätzlichen Eigenschaften verwenden

* Rahmenmodus umstellen

Bei den einzelnen Zeilen/Spalten kann man nicht einfach, mit dem Befehl border, den Zeilen oder Spalten einen Rahmen zuweisen

Man muss zuerst dem table die eigenschaft border-collapse: collapse; hinzufügen

Nun sollte man einfach tr { border: 1px solid black; } verwenden können

* Zeilen formatieren

Hier kann man auch wieder mit background-color, color, usw arbeiten

Mit tbody, thead, tfoot kann man die einzelnen Sektionen bearbeiten

* Zellen formatieren

Einzelne Zellen kann man mit td ansprechen

Bei den Zellen kann man auch wieder die Grundlegenden Eigenschaften verwenden

Man kann die einzelen Rahmenseiten mit border-left/top/right/bottom ansprechen

Um nur die Fusszeilen-Zellen anzusprechen benutzt man den Selektor tfoot td

* Formularfelder Stylen

Hier kann man auch wieder die Grunsätzlichen Eigenschaften verwenden: margin, border, backgroundcolor, color, usw

Wenn man dem label-Element die Eigenschaft display: block; zuordnet, wird die Überschrift auf einer einzelnen Zeile dargestellt

Über input kann man Eingabefelder ansprechen und vergrössern

Man kann auch mit der Eigenschaft border den Rahmen des Eingabefelds bearbeiten

* Head-Bereich

Man fängt wie immer mit <!DOCTYPE HTML> an

Dann leitet man mit <html> ein (am besten fügt man noch das Attribut lang=“de“ an)

Im <head> Element kann man mit <title> den Titel der auf dem Tab angezeigt wird Festlegen

Ausserdem kann man mit <meta charset=“utf-8“> die Kodierung festlegen

Mit <meta name=“description“ content=“Beschreibung“> kann man seine Website beschreiben.

Den Autor kann man mit <meta name=“author“ content=“Name“ seinen Namen angeben

Um eher gefunden zu werden kann man mit <meta name=“keywords“ content=“Schlüsselwörter“ eine Sammlung von Wörtern, die zur Website passen

Um das kleine Icon neben dem Titel(im Tab) festzulegen gibt man <link rel=“shortcut icon“ href=“favicon.ico“>

Um das Icon für Mobiles festzulegen ersetzt man den rel durch rel=“apple-touch-icon-precomposed“

Mit dem Attribut sizes kann man die grösse festlegen

Um die Website Smartphonetauglich zu machen muss man folgender befehl eingeben

<meta name="viewport" content="width=device-width, target-dens i typd i=device-dp, intial-scale=1.0">

Und hier noch das dazu gehörige stylesheet

* Body-Bereich

Den Body-Bereich markiert man mit <body>

In diesem Bereich kommt der ganze Content der grundsätzlich auf die Seite kommt rein

* Layoutabschnitte definieren

Innerhalb des Body-Bereichs gibt es auch wieder Unterteilungen: der Header, der Inhalt und der Footer

Den Header kann mit dem Element <header> gekennzeichnet werden

Für den Content gibt es kein spezielles Element, doch man kann einfach div nehmen

Der Footer wird vom Element <footer> verkörpert

Am besten teilt man den Elementen noch die ID=header, content, footer zu

* Header und Footer mit Inhalten versehen

Im header sollte man erst mal einen Titel per Überschriftenelement darstellen

Der footer kann dazu verwendet werden, um die Seite noch mal zu beschreiben oder andere Seiten zu verlinken

* Inhalte Strukturieren

Man sollte im Content selber auch wieder verschiedenen Sections erstellen, welche man dan später per CSS verschieben und Formatieren kann um die Seite schön zu gliedern

Den Sections auch wieder eine ID zufügen

Man kann nun innerhalb der Sections den ganzen Hauptcontent

* CSS Normalization

Die verschiedenen Browser stellen das HTML immer ein bisschen anders da, das heisst man sollte immer alles genau festlegen

Das macht man am besten, indem man auf github.com/necolas geht und sich die normalize.css - Datei runterlädt

In dieser Datei sollte man zuerst mal alle Kommentare rauslöschen und sie dann ins CSS kopieren

* CSS-Grundformatierung

Nun kann man mit den Grundlegenden Formatierungen beginnen

EEE ist eine gute Hintergrundfarbe

Man kann das <div>-Element als Container für grössere Dinge verwenden

320px ist etwa das Beste für Smartphones und 1140px sollte als max-Wert reichen

Den Scrollbalken unten kann man per overflow-bottom ansprechen

Bilderbreiten kann man mit max-width: 100%;

* Beispiel

Ich werde hier nun meinen HTML-code als Beispiel hinsetzen:

<**html**>  
 <**head lang="de"**>  
 <**meta charset="UTF-8"**>  
 <**title**>Meine Website</**title**>  
 <**meta name="author" content="Roman Schilter"**>  
 <**meta name="keywords" content="Übung, Roman Schilter, HTML, CSS"**>  
 <**meta name="description" content="Diese Seite ist eine Übung von Roman Schilter"**>  
 <**link rel="stylesheet" type="text/css" href="stylesheet2.css"**>  
 <**link rel="shortcut icon" href="img/favicon%20(1).ico"**>  
 </**head**>  
 <**body**>  
 <**header id="header"**>  
 <**h1 id="title"**>Willkommen auf meiner Webseite</**h1**>  
 <**h2 id="subtitle"**>Ich habe diese Seite als Test und Übung gemacht</**h2**>  
 </**header**>  
 <**div id="content"**>  
 <**section id="navi"**>  
 <**nav**>  
 <**ul**>  
 <**li**><**a href="index.html" class="na1"**>Hauptseite</**a**></**li**>  
 <**li**><**a href="seite2.html" class="na2"**>Seite 2</**a**></**li**>  
 <**li**><**a href="seite3.html" class="na2"**>Seite 3</**a**></**li**>  
 </**ul**>  
 </**nav**>  
 </**section**>  
 <**section id="suche1"**>  
 <**form method="post" id="search"**>  
 <**label title="Suche" for="suche"**>Suche</**label**>  
 <**input id="suche" type="search"**>  
 </**form**>  
 </**section**>  
 <**section id="text"**>  
 <**h4**>Hallo</**h4**>  
 <**p**>  
 Wie man bereits im Titel lesen kann, ist dies eine Testseite und wird niemals online gehen. Also ist dieser Text, denn ich hier gerade schreibe, sinnlos.  
 Er ist einfach nur ein Platzhalter und lässt die Seite nicht so leer da stehen.  
 </**p**>  
 <**p**>  
 Freundliche Grüsse Roman  
 </**p**>  
 </**section**>  
 <**section id="me"**>  
 <**h4**>Über mich</**h4**>  
 <**p**>  
 Über mich kann ich eigentlich nicht allzuviel sagen, ausser vielleicht, dass ich ein Mensch bin. Meine Hobbys könnte ich auch noch aufzählen.  
 Nämlich sind das Webseiten basteln und Sinnlose Texte für diese Webseiten schreiben. Übrigens kann ich lesen und schreiben.  
 </**p**>  
 </**section**>  
 <**section id="media"**>  
 <**h4**>Hier ist noch ein Video, dass ich gemacht habe:</**h4**>  
 <**iframe width="853" height="480" src="https://www.youtube.com/embed/9ZU5jt7f6k8?rel=0" frameborder="0" allowfullscreen title="CS:GO"**></**iframe**>  
 </**section**>  
 <**section id="internezzo"**>  
 <**iframe title="internezzo" src="http://www.internezzo.ch/" width="853" height="600"**></**iframe**>  
 </**section**>  
 </**div**>  
 <**footer id="footer"**>  
 <**a href="http://www.internezzo.ch"**>internezzo ag</**a**>  
 </**footer**>  
 </**body**>  
</**html**>

Zur Übersicht kommt hier auch noch der CSS-Teil

**html** {  
 **background-color**: **#cbcbcb**;  
}  
**body** {  
 **width**: 80%;  
 **margin**: 5**px auto auto auto**;  
}  
**#header** {  
 **background-color**: **#4f85bc**;  
 **padding-top**: 15**px**;  
 **margin-top**: 15**px**;  
 **height**: 120**px**;  
 **border-radius**: 15**px**;  
}  
**#header h1**{  
 **text-align**: **center**;  
 **color**: **white**;  
 **text-shadow**: 0 0 10**px rgba**(255,255,255,1) , 0 0 20**px rgba**(255,255,255,1) , 0 0 30**px rgba**(255,255,255,1) , 0 0 40**px #000**, 0 0 70**px #000**, 0 0 80**px #000** , 0 0 100**px #000** ;  
 **-webkit-transition**: **all** 200**ms cubic-bezier**(0.42, 0, 0.58, 1);  
 **-moz-transition**: **all** 200**ms cubic-bezier**(0.42, 0, 0.58, 1);  
 **-o-transition**: **all** 200**ms cubic-bezier**(0.42, 0, 0.58, 1);  
 **transition**: **all** 200**ms cubic-bezier**(0.42, 0, 0.58, 1);  
}  
**#header h2** {  
 **text-shadow**: 1**px** 1**px** 1**px gray**;  
 **text-align**: **right**;  
 **margin-right**: 20**px**;  
}  
**#content** {  
 **background-color**: **#ffffff**;  
 **width**: 98%;  
 **margin-left**: 1.25%;  
 **padding-bottom**: 5**px**;  
}  
**#navi** {  
 **padding**: 10**px**;  
 **float**: **left**;  
}  
**#navi a** {  
 **list-style-type**: **none**;  
 **color**: **#000**;  
 **background-color**: **#4286ff**;  
 **border**: **solid** 1**px #4286ff**;  
 **border-radius**: 5**px**;  
 **text-decoration**: **none**;  
 **padding**: 5**px**;  
 **display**: **block**;  
 **width**: 90**px**;  
}  
**#navi a**:**hover** {  
 **color**: **#fff**;  
 **background-color**: **#363636**;  
 **border-color**: **#363636**;  
}  
**#navi li** {  
 **margin**: 15**px**;  
 **width**: 90**px**;  
 **float**: **left**;  
}  
**#text** {  
 **background-color**: **#fff**;  
 **padding**: 10**px** 10**px** 10**px** 20**px**;  
 **line-height**: 1.25;  
 **width**: 45%;  
 **clear**: **both**;  
 **float**: **left**;  
 **margin-left**: 20**px**;  
}  
**#text h4** {  
 **margin-bottom**: 3**px**;  
}  
**#text p** {  
 **margin-top**: 4**px**;  
 **margin-bottom**: 3**px**;  
}  
**#media** {  
 **background-color**: **#fff**;  
 **clear**: **both**;  
 **margin**: 20**px**;  
 **text-align**: **center**;  
}  
**#media h4** {  
 **margin-bottom**: 7**px**;  
}  
**#me** {  
 **background-color**: **#fff**;  
 **padding**: 10**px** 10**px** 10**px** 20**px**;  
 **line-height**: 1.25;  
 **width**: 45%;  
 **float**: **right**;  
}  
**#me h4** {  
 **margin-bottom**: 3**px**;  
}  
**#me p** {  
 **margin-top**: 4**px**;  
 **margin-bottom**: 3**px**;  
}  
**#footer** {  
 **background-color**: **#000**;  
 **color**: **white**;  
 **margin-top**: -20**px**;  
 **border-radius**: 15**px**;  
 **text-align**: **center**;  
 **height**: 20**px**;  
}  
**#footer a** {  
 **list-style-type**: **none**;  
 **color**: **#ffffff**;  
}  
**#suche1** {  
 **border-radius**: 2**px**;  
 **margin-top**: **px**;  
 **padding**: 10**px**;  
 **margin-left**: **auto**;  
 **padding-top**: 30**px**;  
 **width**: 25%;  
 **margin-right**: 0;  
}  
**li**:**nth-child**(1) {  
 **list-style-type**: **none**;  
}  
**li**:**nth-child**(2) {  
 **list-style-type**: **none**;  
}  
**#internezzo** {  
 **margin**: 20**px**;  
 **text-align**: **center**;  
 **margin-bottom**: 50**px**;  
}  
**ul** {  
 **list-style-type**: **none**;  
}