HTML für Anfänger

Version 0.1

Inhaltsverzeichnis

Warum schreibe ich dieses Tutorial?

Da ich mich schon seit einiger Zeit mit HTML beschäftige, denke ich, dass ich einem Einsteiger viele Tipps für den Einstieg in dieses Thema bieten kann. Oft ist es passiert, dass jemand nach den Basics sucht. Ein Verweis zu www.SelfHTML.org bringt ihn zwar zu der Quelle des Wissens, allerdings ist der Umfang dieser Seite meist erschlagend und man findet keinen richtigen Einstieg in das Thema. Viele Anfänger machen zudem den Fehler, schlechten Code zu lernen. Dieser ist unnötig kompliziert und ich will zeigen, wie man mit nur wenigen Elementen komplette Webseiten erstellen kann.

An wen ist das Tutorial gerichtet?

Da ich mich nur sehr schwer in meine Anfänge des Computerlernens zurückversetzen kann, schreibe ich dieses Tutorial für Computeranwender, die sich schon einmal im Internet bewegt haben und einen gewissen Umgang mit ihrem Computer haben. Falls sich herausstellt, dass ich schon auf einer zu hohen Ebene einsetze und mit Fachbegriffen um mich werfe, die dir als Leser gar nichts sagen, so bitte ich um Feedback, damit ich an den entsprechenden Stellen Änderungen vornehmen kann. Nun aber genug der Vorrede, lasset das Tutorial beginnen.

Was wird benötigt?

Computer

Das wichtigste für den Anfang ist der Computer, da du dieses Dokument liest, wirst du wahrscheinlich Zugriff zu einem haben. Das Betriebssystem spielt bei der Arbeit mit Webseiten nur einen geringen Unterschied, der sich in den Tools und Browsern unterscheidet.

Editor

Um Webseiten zu erstellen brauchen wir einen Editor. Dies geht auf Windows mit dem Editor oder Notepad, aber nicht mit Word. Besser geeignet sind Programme wie Phase5 (Windows, nur privat kostenlos) oder Quanta Plus (Linux), diese Editoren bieten Syntax Highlighting, was den Code besser lesbar macht. Zudem unterstützen sie den Anwender noch mit Projektverwaltung und zusätzlichen Möglichkeiten.

Der Anfang

Tags, was sind das?

HTML ist eine reine Textstrukturierungssprache, die keine große Ähnlichkeit mit höheren Programmiersprachen hat. Man will also den Text strukturieren, dies erreicht man durch Überschriften, Absätze, Listen und ähnliche Elemente. Fehlen dürfen dort natürlich keine Links, die ein Grundbestandteil für das Bewegen zwischen Webseiten sind. Der Text, den man strukturieren will, bettet man nun in HTML Tags ein, sodass ein Browser erkennen kann, welche Elemente er wie verstehen muss.

Welche Tags brauche ich?

Im Gegensatz zu der oft gehörten Meinung braucht man gar nicht viele Elemente, um vernünftigen HTML Code zu schreiben. Weniger ist meiner Meinung nach oft mehr. Folgend die wichtigsten Tags:

<html> </html>	Dieses Tag leitet eine Webseite ein und beendet diese.
<head> </head>	Dieses Tag markiert den Kopfbereich einer Webseite, hier wird der Titel angegeben, der oben auf dem Browserfenster erscheint. Zudem können hier weitere Elemente definiert werden, die aber für den Anfang nicht so wichtig sind.
<title> </title>	Dieses Tag gibt den Titel an, der oben auf dem Browser erscheint.
<body> </body>	Dieses Tag markiert den Textbereich eines HTML Dokumentes. Hier wird der Inhalt hineingeschrieben.
	Eine Überschrift erster Ordnung
	Eine Überschrift zweiter Ordnung
	Eine Überschrift dritter Ordnung (geht bis zu 6. Ordnung)
	Dieses Tag gibt einen Textblock an. Nach einem Textblock folgt dann ein Absatz. Oft wird dieses Element fälschlicherweise für einen Absatz benutzt, dieser ergibt sich aber aus der Strukturierung selber.
 	Erzwungener Zeilenumbruch
<a> 	Ein Link
	Eine nicht nummerierte Liste
	Eine geordnete (nummerierte) Liste
	Listenelemente dieser Liste
<img> </img>	Fügt ein Bild ein
	Dieses Tag markiert einen bestimmten logischen Bereich. Dieser kann z.B. mehrere Textblöcke umfassen oder mehrere Links oder … Dieses Tag hat zuerst einmal keine Auswirkung, übernimmt aber später bei der

Gestaltung der Seite eine sehr wichtige Rolle

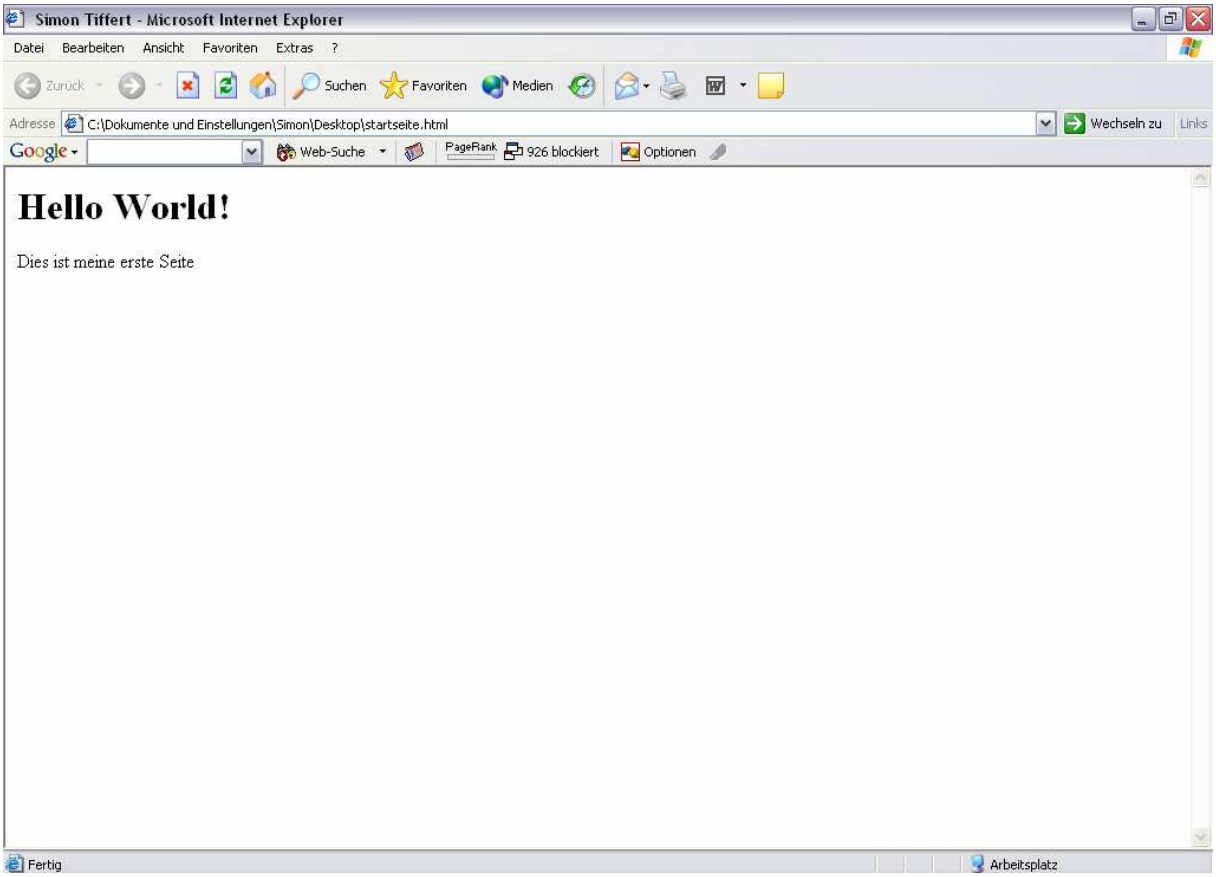
Sind doch nicht so viele, oder?
Natürlich gibt es eine Menge mehr, aber für den Anfang reichen diese aus, um alles Wichtige zu erzeugen.
Viele werden sich jetzt fragen, wo ist das Tag für „fett“ oder wie ändere ich die Schriftgröße. Aber HTML bestimmt nicht das Aussehen einer Webseite, es strukturiert sie nur. Natürlich ist es möglich dies fest in den Quellcode zu schreiben, aber mit den oben genannten Tags kann man eine Webseite komplett strukturieren und der Browser stellt sie auch dementsprechend dar. Warum will man Text „fett“ machen oder die Schriftgröße ändern? Vielleicht um eine Überschrift hervorzuheben? Aber genau dafür gibt es schon Tags, die auch dafür benutzt werden.
Ähnlich verhält es sich in meinem Textverarbeitungsprogramm, hier nennt sich dieses Verfahren Formatvorlagen. Diese markieren die Strukturierung des Textes und man kann dann eine Formatvorlage anpassen und alle Elemente, die damit markiert worden, ändern sich damit auch. Will ich nun einen gewissen Teil der Überschriften unterstrichen haben, so muss ich bei einem hundertseitigen Dokument nicht erst alle Seiten durchsuchen, sondern ändere dies an einer Stelle in sekundenschnelle. Genauso geht man auch bei guten Webseiten vor.
Die Webseite wird mit oben stehenden Elementen erst einmal aus schwarzem Text bestehen, nicht besonders schön, aber erst einmal die Basics durcharbeiten und lernen, bis man dann ans Aussehen der Seite geht. Bei richtiger Durcharbeitung meiner Tutorials stellt es kein Problem dar, das Aussehen zu ändern, ohne große Änderungen im Quellcode vorzunehmen.

Die erste Seite

Wir öffnen unseren Editor und schreiben folgenden Inhalt in die Datei

```
<html>
	<head>
		<title>Euer Name</title>
	</head>

	<body>
		<h1>Hello World!</h1>
		<p>Dies ist meine erste HTML Seite</p>
	</body>
</html>
```

Dann speichern wir diese Datei unter dem Namen startseite.html in einem Ordner unserer Wahl.
Öffnen wir nun diese solltet ihr folgendes Ergebnis erhalten:



Als erstes habt ihr euch bestimmt gefragt, warum am Anfang mancher Zeilen soviel Platz gelassen wurde. Dies ist eine Einrückung, die den Code lesbarer macht. Solange ihr ein Tag noch nicht geschlossen habt, so rückt ihr das folgende Tag einen gewissen Abstand ein. So wird der Quellcode viel strukturierter und ihr findet euch in Seiten schneller zurecht. Diese können nämlich schon einmal über mehrere hundert Zeilen gehen und bei schlechter Formatierung dieser Datei findet man dann die entsprechende Stelle nur noch schlecht.

Nun kurz die Erläuterung zu den einzelnen Zeilen.
In der ersten Zeile öffnen wir unser HTML Dokument mit dem <html>-Tag, welches wir erst ganz am Ende des Datei wieder schließen. In der zweiten Zeile beschreiben wir den Kopf (<head>) unseres Dokuments. Hier steht nun der Titel (<title>), der angibt, was in der Fensterleiste des Browsers erscheint. Setzt man eine Seite als Lesezeichen (auch Favoriten genannt), so wird dort auch dieser Titel als Beschreibung gespeichert. Der Name sollte also aussagekräftig gewählt werden, sodass er im besten Fall schon einen Schluss auf den Inhalt zulässt, gerade wenn man mit seiner Seite in Suchmaschinen auftaucht, da dort dieser Titel auch verwendet wird. Damit haben wir für unser Dokument den Kopf auch schon fertig und können ihn in der nächsten Zeile schließen (</head>).
Nun beginnen wir mit dem Body des HTML Dokuments mit dem Tag <body>.
Als erstes setzen wir eine Überschrift im Dokument ein, welche wir mit <h1> beginnen und mit </h1> beenden. Danach setzen wir Text ein, den wir mit beginnen und mit beenden. Man könnte den Text auch einfach so ins Dokument schreiben und er würde angezeigt werden. Allerdings kann

der Browser den Text keinem Element zuordnen und man hat später keinen Zugriff mehr darauf, wenn man das Design der Seite gestalten will.
Nachdem wir den Body beendet haben, schließen wir ihn mit </body> und können nun auch unser HTML-Dokument mit </html> beenden.

Das erste HTML-Dokument ist somit erstellt und ich hoffe, euch ist das Grundgerüst einer HTML-Datei an diesem Beispiel etwas klarer geworden. Im nächsten Kapitel werde ich nun auf die wichtigsten Tags eingehen und diese an kurzen Beispielen erläutern.

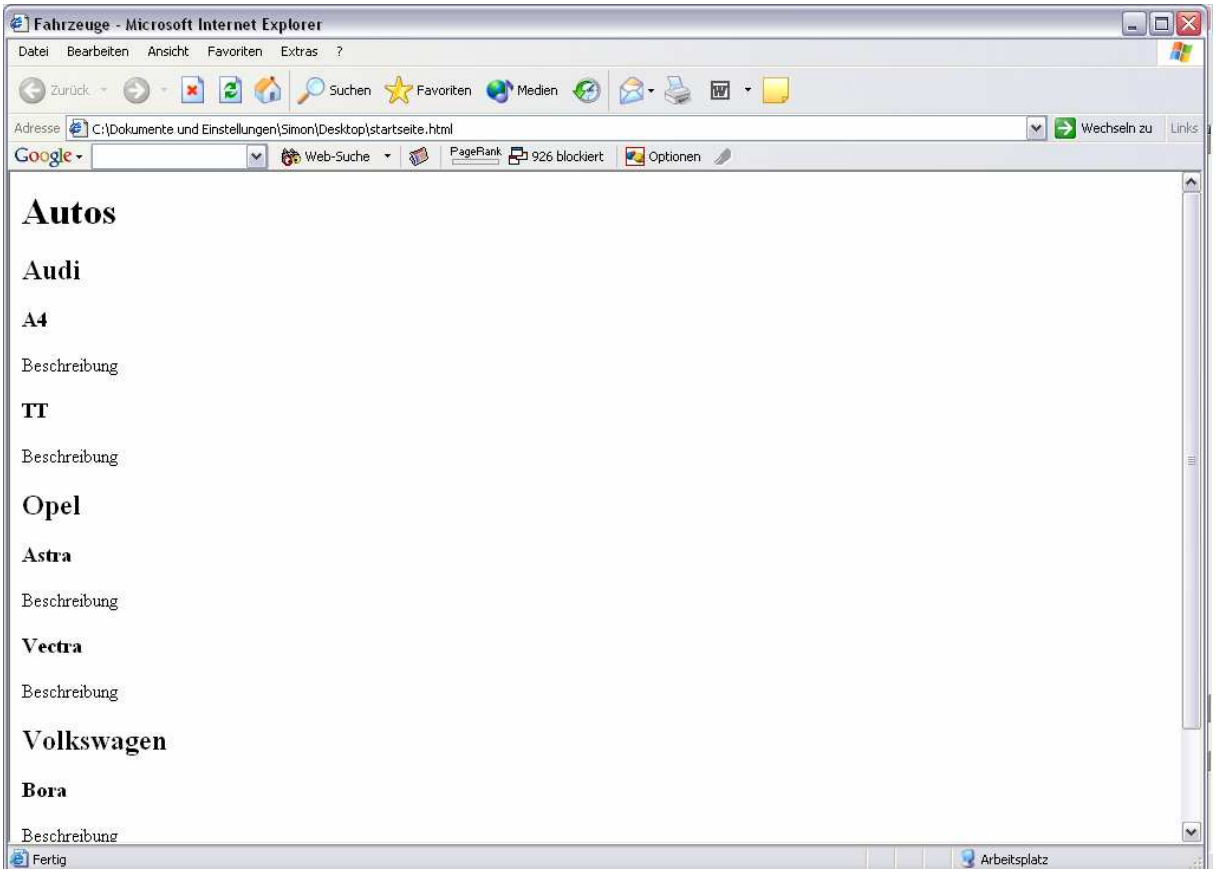
Die wichtigsten Tags

<h1>-<h6>

Diese Tags sind für die Überschriften in einem HTML-Dokument zuständig. Dabei hat das Tag <h1> die höchste Ebene, die bis zur sechsten Ebene heruntergebrochen werden kann. Diese Tags geben die Gliederung des Dokuments mit ihren Unterteilen bzw. Unterkapiteln an.
Beispiel einer Gliederungsstruktur, die man für Autos aufbauen würde:

Autos	`<h1>Autos</h1>`
Audi	`<h2>Audi</h2>`
A4	`<h3>A4</h3>`
Beschreibung	`<p>Beschreibung</p>`
TT	`<h3>TT</h3>`
Beschreibung	`<p>Beschreibung</p>`
Opel	`<h2>Opel</h2>`
Astra	`<h3>Astra</h3>`
Beschreibung	`<p>Beschreibung</p>`
Vectra	`<h3>Vectra</h3>`
Beschreibung	`<p>Beschreibung</p>`
Volkswagen	`<h2>Volkswagen</h2>`
Bora	`<h3>Bora</h3>`
Beschreibung	`<p>Beschreibung</p>`
Golf	`<h3>Golf</h3>`
Beschreibung	`<p>Beschreibung</p>`

Das Beispiel sieht in HTML dann wie folgt aus:



Man sieht hier sehr schön, wie der Browser die verschiedenen Elemente erkennt und diese auch der Gliederungsebene entsprechend darstellt.

In HTML ist es nicht nur wichtig einfach Inhalte darzustellen. Bevor man sich an Themen heranbewegt erstellt man eine Gliederung für die Texte. Dies macht es leichter, dies später umzusetzen und bringt vor allem der Person, für die man die Inhalte erstellt große Vorteile. Strukturierte Texte sind leichter und schneller zu lesen. Man erkennt die wichtigen Kapitel, kann sich die nötigen Informationen direkt an der richtigen Stelle holen, ohne sich durch ein ganzes Dokument zu lesen. Außerdem sind Informationen schneller wieder zu finden, wenn man etwas bestimmtes nachlesen will.
Es empfiehlt sich also für seine Dokumente eine komplette Gliederung anzulegen und diese dann auch mit den entsprechenden Tags in HTML umzusetzen.

<p>

Wir haben das Element schon etwas öfters vorher erwähnt, trotzdem führe ich es an dieser Stelle noch einmal auf, da es das Tag für Inhalte darstellt. Sobald wir Fließtext schreiben, wird das Tag <p> verwendet. Braucht man keine Absätze in seinem Inhalt, so umfasst das Anfangs- und Endtag den kompletten Text. Werden Absätze benötigt, so grenzen mehrere <p>-Bereiche aneinander.

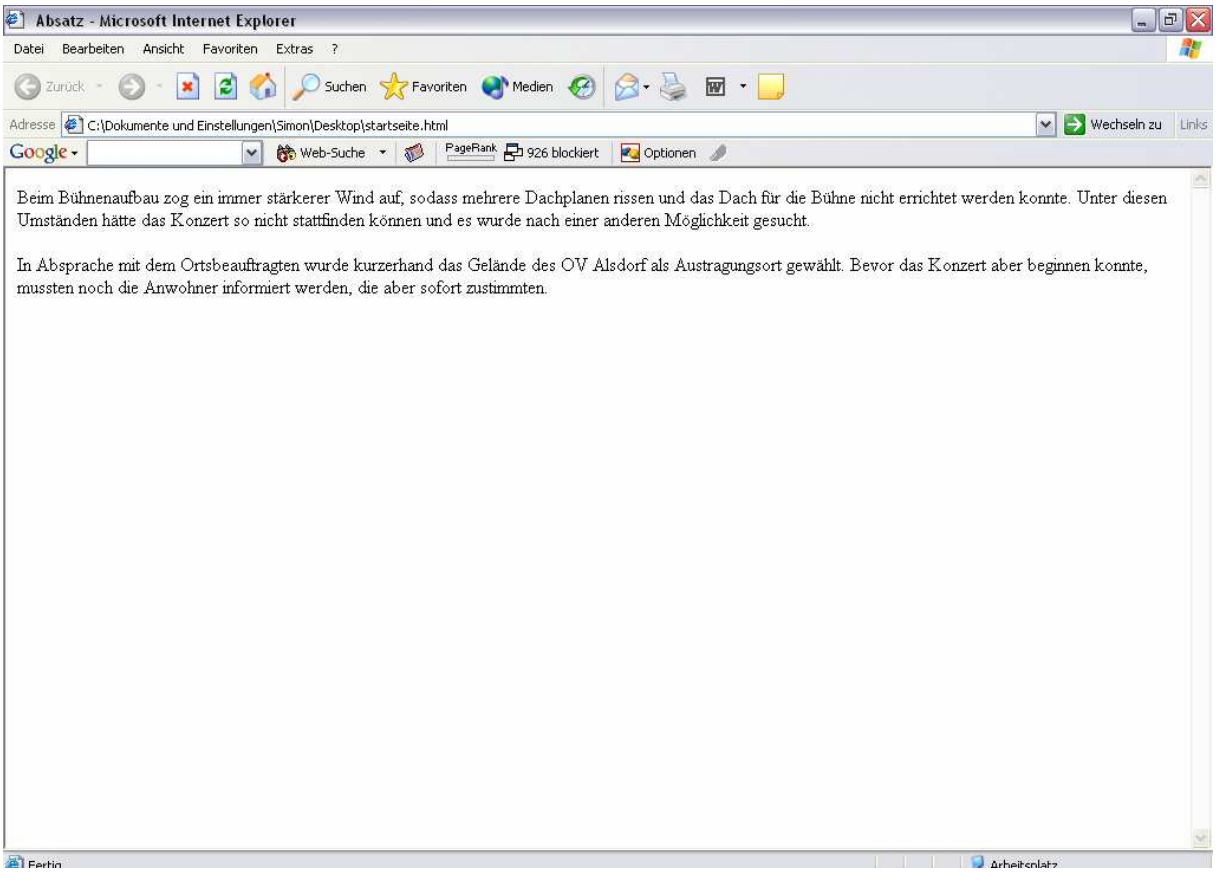
Hier auch wieder ein Beispiel mit seiner Umsetzung in HTML:

Beim Bühnenaufbau zog ein immer stärkerer Wind auf, sodass mehrere Dachplanen rissen und das Dach für die Bühne nicht errichtet werden konnte. Unter diesen Umständen hätte das Konzert so nicht stattfinden können und es wurde nach einer anderen Möglichkeit gesucht.

In Absprache mit dem Ortsbeauftragten wurde kurzerhand das Gelände des OV Alsdorf als Austragungsort gewählt. Bevor das Konzert aber beginnen konnte, mussten noch die Anwohner informiert werden, die aber sofort zustimmten.

```
<p>

  Beim Bühnenaufbau zog ein immer
  stärkerer Wind auf, sodass
  mehrere Dachplanen rissen und das
  Dach für die Bühne nicht
  errichtet werden konnte. Unter
  diesen Umständen hätte das
  Konzert so nicht stattfinden
  können und es wurde nach einer
  anderen Möglichkeit gesucht.

</p>
<p>

  In Absprache mit dem
  Ortsbeauftragten wurde kurzerhand
  das Gelände des OV Alsdorf als
  Austragungsort gewählt. Bevor das
  Konzert aber beginnen konnte,
  mussten noch die Anwohner
  informiert werden, die aber
  sofort zustimmten.

</p>
```

Und so sieht dann der Absatz im Browser aus:



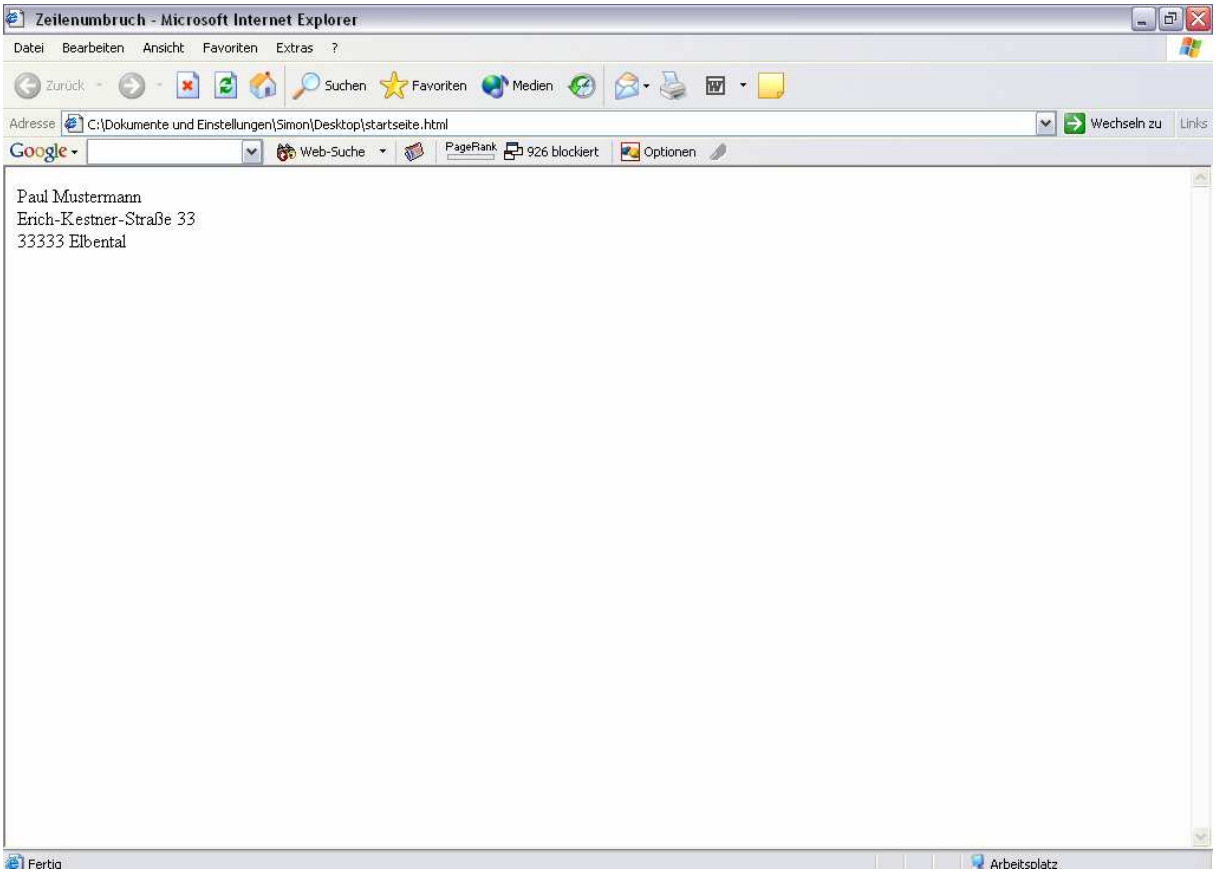
Ist etwas breit gezogen, aber das ist eben das besondere an HTML. Der Browser bricht den Text selber um und würde sich sofort einer Änderung der Fenstergröße anpassen. So muss man sich keine Gedanken machen, wie man Zeilenumbrüche und ähnliches in einen Text einfügt. Dies erledigt HTML und die Umsetzung im Browser für einen.

*
*

Viele die schon einmal in HTML Code geschaut haben, werden sich an dieser Stelle fragen, warum ich nicht
 schreibe, wie es sehr oft zu finden ist. Ich gehe mit diesem Tutorial etwas über HTML heraus, um aber später nicht Sachen neu lernen muss, gewöhnt euch diese Schreibweise an.
Dieses Tag erzeugt einen Zeilenumbruch, allerdings sollte man es nicht verwenden, um Absätze zu erzeugen, da dies durch das <p>-Tag und die Trennung von Textblöcken erreicht wird (Kapitel zu <p>). Allerdings ist es oft nötig, einen einfachen Zeilenumbruch zu haben. Ich nutze dies schon einmal öfters, wenn ich Adressen aufschreibe (es gibt auch noch schönere Lösungen dafür).

```
<p>
      Paul Mustermann <br />
      Erich-Kestner-Straße 33 <br />
      33333 Elbental
</p>
```

Als Ausgabe erhalten wir:



Zum
-Tag gibt es sonst nicht mehr allzu viel zu sagen, außer dass man es so selten wie möglich verwenden sollte. Um Freiraum zu schaffen, gibt es viel bessere Möglichkeiten, zu denen ich später noch kommen werde. Denn setzt man zu viele
-Tags in seinen Quellcode, so hat dieser schon eine Art Aussehen. Will man nun aber das Aussehen seiner HTML-Dokumente ändern, so muss man je nachdem wieder in den Quellcode und
-Tags löschen und dies will man vermeiden.

<a>

Das Tag, um Links zu erstellen. Davon gibt es verschiedene Arten, nämlich interne Links und externe Links. Interne Links beziehen sich auf das Dokument selber und ermöglichen eine Navigation innerhalb einer Seite mittels Ankern. Öfters sind allerdings externe Links zu anderen Seiten seiner eigenen Webseiten oder zu anderen Seiten im WWW.

Interne Links

Damit man innerhalb einer Seite navigieren kann, muss der Browser erst einmal Stellen im Dokument haben, zu denen er springen kann. Diese werden auch mit dem <a> Tag beschrieben und haben folgendes Aussehen:

```
<a name="Ankername">Text auf den der Anker zeigt</a>
```

Dann können wir diese Stelle im Dokument anspringen, indem wir einen Verweis (Link) mit folgendem Aussehen definieren:

```
<a title="Link zu Anker" href="#Ankername">Link</a>
```

Interne Links sind sehr gut bei langen Dokumenten geeignet, um von oben eine Navigation durch das Dokument zu ermöglichen. Außerdem können diese verwendet werden, um von anderen Seiten direkt an eine bestimmte Stelle im Dokument zu springen.
Man muss allerdings sagen, dass ein Dokument eigentlich schon zu lang ist, wenn interne Links verwendet werden müssen. Normale Dokumente sollen so lang sein, dass sie maximal zwei Bildschirmseiten umfassen, sind die Seiten länger, so empfiehlt es sich, diese auf mehrere Dokumente aufzuteilen.
Natürlich gibt es immer Gründe, warum man auch einmal eine lange Seite braucht, dies könnte ein Handbuch oder ein Tutorial sein, welches zusammenhängend in einer Datei erstellt wird. Dort sind Anker innerhalb des Textes unerlässlich.

Externe Links

Externe Links werden dazu verwendet, um auf Dateien zu verweisen. Dies können sowohl HTML-Dokumente, wie auch andere Dateiformate sein. Am häufigsten wird allerdings der Link zu einem HTML-Dokument sein, da dies für die Navigation zwischen verschiedenen Webseiten verwendet wird.

Ein externer Link hat den Aufbau:
```
<a title="Titel des Links" href="Dokument">Name der erscheint</a>
```

Das Attribut „title" ist dafür da, einem Link einen Titel zu geben, dies kann sehr hilfreich sein, um eine weitere Erklärung zum Inhalt des Links zu geben, da der Name meist nur aus einem oder wenigen Worten besteht.
Das Attribut „href" ist dann der Name des Dokuments auf das wir verweisen. Es gibt nun verschiedene Arten auf dieses Dokument zu verweisen:

Absoluter Link

Ein absoluter Link gibt das Dokument, dass auf unserem Rechner liegt, vom Ursprungsverzeichnis unseres Verzeichnisses aus. Er fängt also ganz oben im Dateibaum an und hangelt sich dann bis zur entsprechenden Datei. Ein Beispiel auf einem Linuxrechner wäre:
```
/home/www/kunden/kundennummer/index.html
```
Ein absoluter Link ist daran zu erkennen, dass er immer mit einem „/" anfängt.

Relativer Link

Ein relativer Link bezeichnet die Position des Links vom aktuellen Verzeichnis aus gesehen. Befinden wir uns in unserem Heimverzeichnis und wollen eine Datei im gleichen Verzeichnis aufrufen, dann können wir dies einfach mit dem Dateinamen tun:
```
datei2.html
```
Eine andere Schreibweise ist

`./datei2.html`
wobei „./“ einfach nur das aktuelle Verzeichnis ist.

Befindet sich der Link dagegen ein Verzeichnis höher, so wird der Verzeichniswechsel durch das voranstellen eines „../“ gekennzeichnet.
`../datei2.html`
erreicht also die Datei, die eine Ebene höher liegt. Für jede weitere Ordnerebene setzt man ein weiteres „../“ vor den Pfad.
`../../../datei2.html` geht um drei Ebenen höher im Verzeichnisbaum.

Wenn man in ein Verzeichnis Wechseln will, dass parallel zu dem jetzigen Verzeichnis liegt, so kann man einfach eine Ebene hoch wechseln und dann den Ordner adressieren:
`../anderer_ordner/datei2.html`

Will man dagegen in Unterverzeichnisse wechseln, so setzt man einfach den (oder die) Ordnernamen davor:
`anderer_ordner/datei2.html`
Dies ist auch wieder gleich mit:
`./anderer_ordner/datei2.html`

Webpfad

Ein Link zu einer anderen Homepage setzen wir, in dem wir den Namen der Domain nehmen.
www.thw-alsdorf.de
Wenn man aber eine Datei direkt innerhalb dieser Webseite erreichen will, so spricht man von einem Deep Link, der dann den Domainnamen und dann den Pfad zur bestimmten Datei hat:
www.thw-alsdorf.de/code/sonstiges/guestbook.htm führt uns direkt zum Gästebuch der Seite des Ortsverbands Alsdorf.

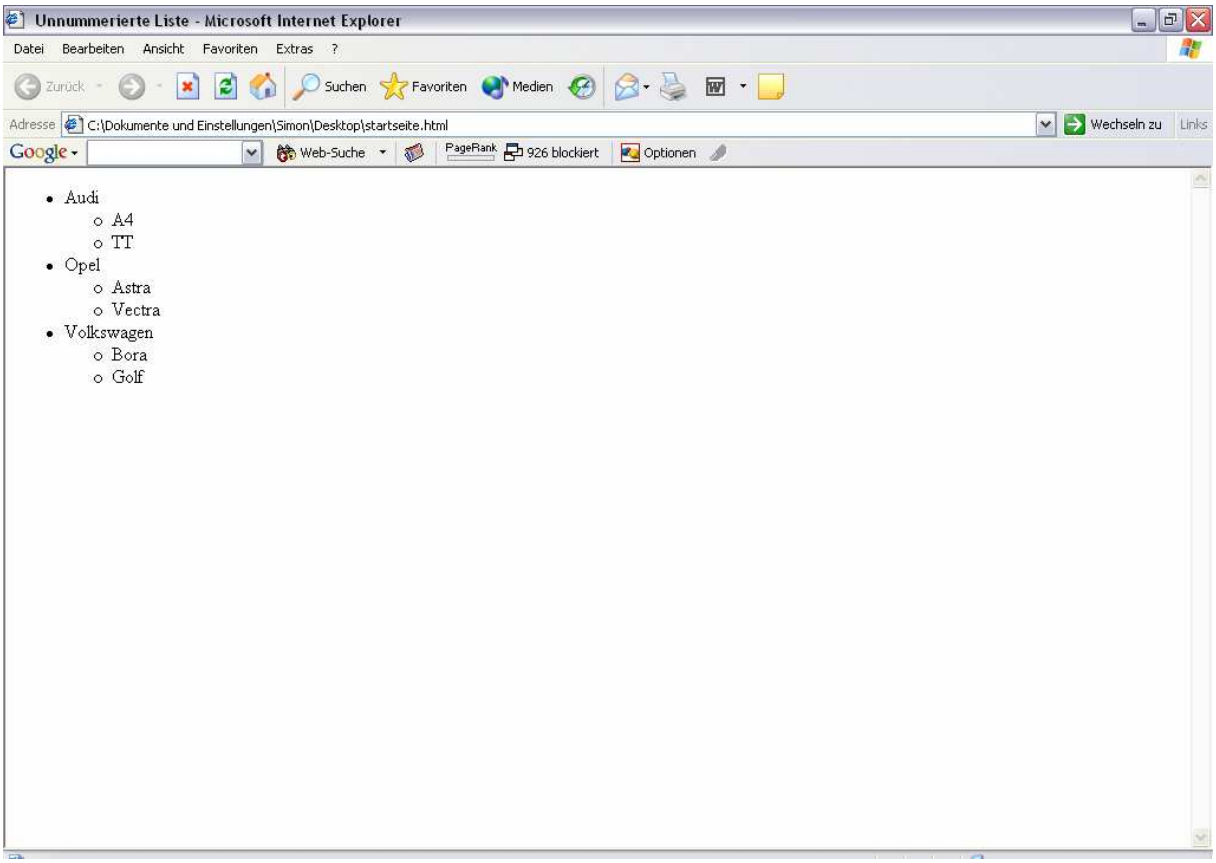
* und *

Kommen wir nun zu den Listen. Davon gibt es zwei Arten, nämlich Aufzählungslisten mit Nummerierung und Aufzählungslisten ohne Nummerierung. Da man Listen vielfältig einsetzen kann und dort selten eine Nummerierung braucht, kommt sehr oft die <ul> (unordered list) zum Einsatz. Listen kann man auch in Ebenen schachteln, was sie für Gliederungen, wie zum Beispiel die Navigation auf einer größeren Seite interessant macht.
Eine Liste wird von <ul> bzw. umschlossen, wobei die Listenelemente durch das Tag <li> umschlossen werden.
Ich hatte weiter oben das Beispiel mit den Autos gebracht, welches ich hier wieder ähnlich verwende. Allerdings soll es diesmal eine reine Aufzählung von Autos sein (keine Dokumentgliederung):

```
Audi
   A4
   TT

Opel
   Astra
   Vectra

Volkswagen
   Bora
   Golf
```

```
<ul>
   <li>Audi</li>
   <ul>
      <li>A4</li>
      <li>TT</li>
   </ul>

   <li>Opel</li>
   <ul>
      <li>Astra</li>
      <li>Vectra</li>
   </ul>

   <li>Volkswagen</li>
   <ul>
      <li>Bora</li>
      <li>Golf</li>
   </ul>
</ul>
```

Dies wird dann vom Browser wie folgt umgesetzt:



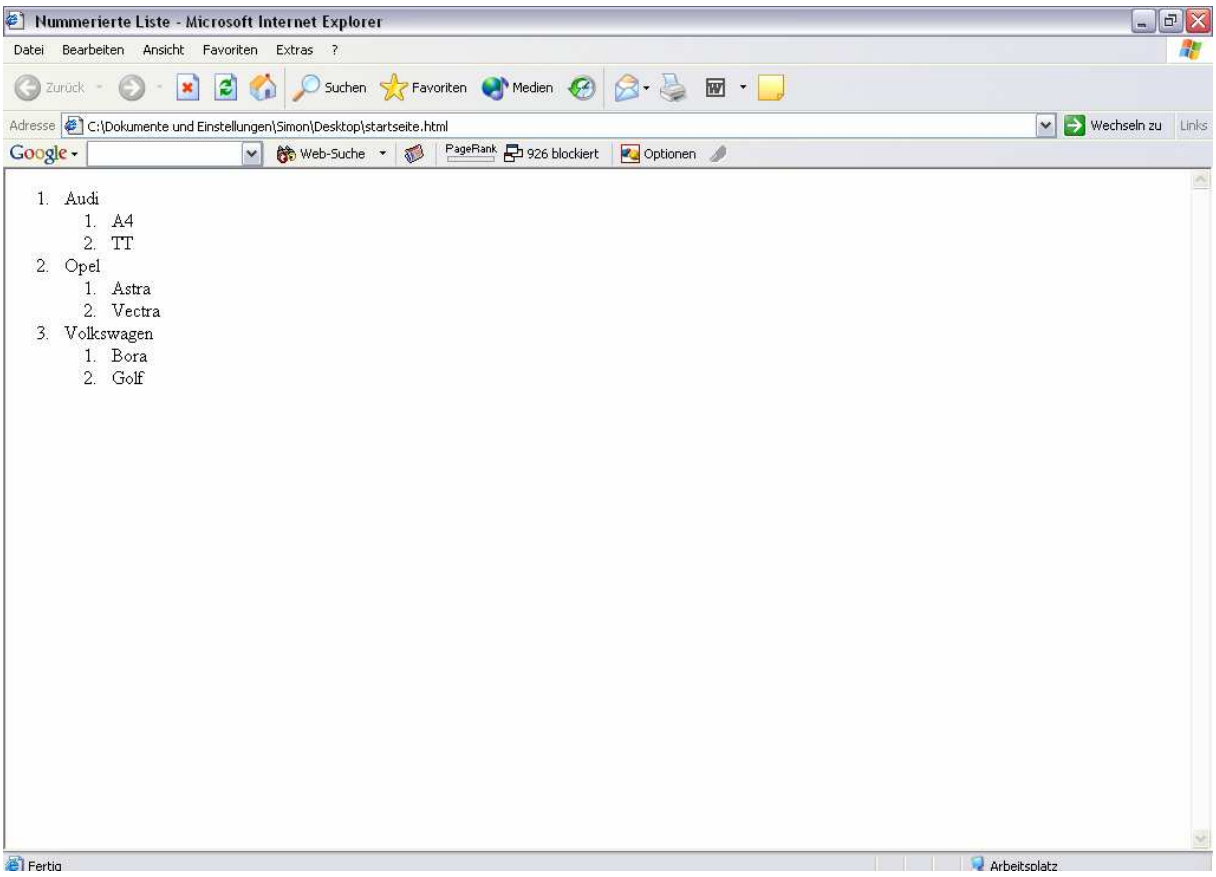
Wenn man eine nummerierte Aufzählung braucht, so ersetzt man das <ul> Tag im oberen Beispiel durch das <ol> Tag:

Audi
- A4
- TT

Opel
- Astra
- Vectra

Volkswagen
- Bora
- Golf

```
<ol>
   <li>Audi</li>
   <ol>
      <li>A4</li>
      <li>TT</li>
   </ol>

   <li>Opel</li>
   <ol>
      <li>Astra</li>
      <li>Vectra</li>
   </ol>

   <li>Volkswagen</li>
   <ol>
      <li>Bora</li>
      <li>Golf</li>
   </ol>
</ol>
```



Listen sollten immer dann zum Einsatz kommen, wenn man sie sinnvoll im Fließtext unterbringen kann. Außerdem bieten sie gute Möglichkeiten zur Gliederung, weswegen man die Navigation seiner kompletten Webseite auch darüber aufbauen sollte.

<img>

Auf Dauer ist aber Text alleine etwas langweilig und man will die Seite mit ein paar Bildern aufpeppen. Aber Vorsicht, erstens sind Bilder riesige Dateien im Vergleich zum Text und erhöhen somit die Ladezeit einer Seite drastisch. Zudem können zum Beispiel blinde Menschen mit Bildern in HTML Seiten nichts anfangen und man grenzt diese damit aus. Somit ist es nur begrenzt sinnvoll, Bilder sehr ausführlich auf einer Seite zu nutzen. Ich selber verwende Bilder meist nur noch in einer Galerie auf der Seite und sehr begrenzt im Design, wobei ich sie dort über eine andere Technik einbinde, da die Bilder keine sinnvollen Nutzinformationen mit sich bringen.
Bei News- oder Portalseiten kann dies etwas anders sein, weil dort auch Informationen mit den Bildern mitgeteilt werden, aber auch dort sollte die Verwendung auf das nötige Minimum reduziert werden.
Nun aber zu dem Tag selber:

Das Attribut „alt" ist dafür da, einen Alternativtext anzuzeigen, falls das Bild nicht angezeigt werden kann oder die Bildansicht deaktiviert ist.
Das Attribut „title" gibt eine Beschreibung des Bildes an. Am einfachsten stellt man sich vor, man will einer Person das Bild in wenigen Worten übers Telefon beschreiben.
Beim Attribut „src" folgt dann schließlich der Pfad zum Bild.

Bei diesem Tag gibt es noch weitere Attribute, wie z.B. Höhe und Breite des Bildes, einfach mal in der Dokumentation nachschauen.

* *

Dieses Tag ist dafür da, einen bestimmten Bereich einer Webseite logisch zu kennzeichnen. Eine Webseite besteht ja nicht nur aus wenigen Zeilen Text, sondern meist aus mehreren Elementen. Dort gibt es einmal die Navigation, dann den Inhalt und zum Beispiel noch eine Fußzeile. Diese Elemente kann man dann in ein Tag einschließen. Dies bringt einem im Moment noch nicht sehr viel, da wir noch nicht beim Design einer Webseite sind. Allerdings können einem Bereich bestimmte Eigenschaften zugewiesen werden, was im weiteren Verlauf noch größere Bedeutung erhalten wird.

Abschlussbemerkung

Mit den vorgestellten Tags ist es möglich komplette Webseiten aufzubauen, die einen sehr lesbaren Code erzeugen. Ich werde in diesem Tutorial noch ein größeres Beispiel unterbringen, welches eine komplette Webseite aufbaut und gliedert, möchte aber erst einmal etwas Kritik und Vorschläge sammeln, ob es überhaupt Sinn macht, an diesem Tutorial weiter zu arbeiten. Also schreibt fleißig was ihr zu den gelesen Zeilen denkt und ob sie euch geholfen haben. Auch Korrekturen für entdeckte inhaltliche Fehler und Rechtschreibefehler sind immer sehr hilfreich ein gutes Tutorial zu entwickeln.

In weiteren Tutorials widme ich mich dann dem Thema XHTML, wobei ich dies mit diesem Tutorial schon zu 90% beschrieben habe und nur noch ein paar Details fehlen. Wer also nach meiner Methode HTML lernt, ist auch für die Zukunft gewappnet, da XHTML mit seiner XML Kompatibilität fortgeschritten ist.
Außerdem geht es dann um das Designen einer Webseite. Dies fängt bei der Einfärbung von Text an und endet dabei, Elemente auf der Webseite auszurichten. Das Ganze ohne unseren Quellcode den wir erstellt haben zu ändern. Das Wunder heißt Cascading Style Sheets (CSS) und bietet ungeahnte Möglichkeiten.

Wenn das Interesse an den Tutorials so riesig sein sollte, so werde ich mich auch hinreißen lassen, ein Tutorial über die Gestaltung von Webseiten mittels PHP zu schreiben, um damit den Grundstein für Webseiten in einem Content Management System (CMS) zu berichten.