

Kapitel 3

HTML- Wiederholung

In diesem Kapitel wiederholen Sie die wichtigsten HTML-Grundbegriffe, die Sie später für die Erstellung von CGI-Programmen brauchen werden. HTML-Codierung ist ein hilfreicher Einstieg in die Programmierung. Sie lernen bereits wichtige Eigenschaften von Programmiersprachen kennen und können dann auf Ihr schon vorhandenes Wissen aufbauen.

Da wir Perl im Laufe dieses Buches in Zusammenarbeit mit Webseiten verwenden wollen, ist ein HTML-Grundverständnis unbedingt notwendig. Vielleicht besitzen Sie bereits ein solides Wissen über HTML-Code, vielleicht haben Sie bisher Ihre Webseiten aber auch nur mit einem WYSIWYG-Editor geschrieben.

WAS IST DAS?

Ein **WYSIWYG**-Editor ist ein Editor, der wie ein modernes Textverarbeitungsprogramm funktioniert: Die Anzeige am Bildschirm entspricht bereits dem Endergebnis (auf Papier oder als Webseite). WYSIWYG ist eine Abkürzung für »What you see is what you get« – Sie bekommen, was Sie sehen – und wird »wi-si-wig« ausgesprochen.

Bei Textverarbeitungsprogrammen ist WYSIWYG heute eine Selbstverständlichkeit. Bei Web-Editoren hat sich das WYSIWYG-Prinzip erst in den letzten Jahren durchgesetzt. Mit einem guten Editor wie Dreamweaver können Sie etwa 90 Prozent der Zeit im WYSIWYG-Modus arbeiten und nur in der verbleibenden Zeit müssen Sie den Code direkt bearbeiten. Ganz ohne HTML-Code geht es aber meistens trotzdem nicht.

HTML mit dem Texteditor

Wir begeben uns nun auf eine Zeitreise zurück in die Anfänge des Webdesigns und arbeiten nur mit einem Texteditor und dem Webbrowser.

Unter Windows können Sie immer den *Editor* verwenden. Sie finden ihn im Start-Menü unter PROGRAMME/ZUBEHÖR/EDITOR oder im Windows Explorer unter `c:\windows\notepad.exe`.



Editor

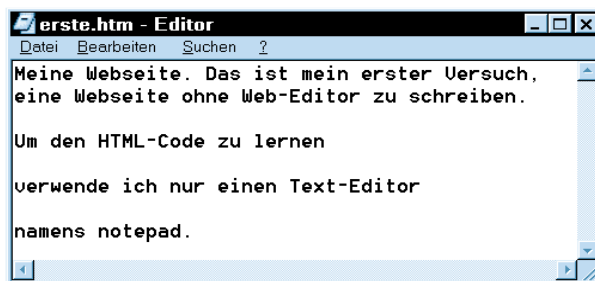


Abbildung 3.1: Editor/Notepad

- 1 Starten Sie den Editor.
- 2 Tippen Sie ein paar Zeilen Text ein.
- 3 Speichern Sie den Text in einer Datei mit dem Namen *erste.htm* im Verzeichnis *C:\EasyPerl* ab. (DATEI/SPEICHERN UNTER). Setzen Sie beim ersten Speichern den Dateityp auf ALLE DATEIEN.
- 4 Starten Sie den *Windows-Explorer* und öffnen Sie den Ordner *C:\EasyPerl*. Dort finden Sie die Datei, die Sie gerade gespeichert haben.
- 5 Überprüfen Sie, ob der Dateiname passt. Der Editor hängt manchmal die Endung *.txt* an den Dateinamen an. Falls das passiert ist, benennen Sie die Datei um.
- 6 Starten Sie den Webbrowser mit einem Doppelklick auf die Datei.

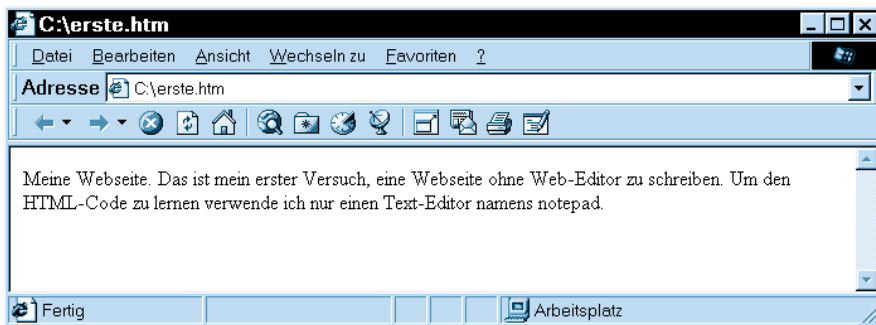


Abbildung 3.2: HTML-Datei im Browser (hier: Internet Explorer)

Wie Sie in Abbildung 3.2 sehen können, wird der Text völlig unformatiert angezeigt. Zeilenumbrüche, Tabulatoren und mehrere Leerzeichen werden völlig ignoriert. Wenn Sie das Fenster des Webbrowsers kleiner oder größer machen, ändert sich der Zeilenumbruch.

Sie werden nun lernen, wie Sie diesen Text mit HTML-Kommandos formatieren können. Dabei müssen Sie immer wieder Änderungen in der HTML-Datei vornehmen und das Ergebnis im Webbrowser betrachten.

Tipp: Sie können den Editor und den Webbrowser gleichzeitig benutzen. Nach jeder Änderung im Editor müssen Sie die Datei speichern. Mit der Tastenkombination **ALT** + **F5** wechseln Sie schnell zum Webbrowser. Dort müssen Sie die Ansicht aktualisieren und können dann die Auswirkungen Ihrer letzten Änderung sehen.

HTML-Tags

HTML (HyperText Markup Language) ist eine einfache Programmiersprache. Seit Ihrer Entstehung 1990 ist sie gewachsen und hat sich immer wieder verändert. Sie werden jetzt HTML in der Version XHTML kennen lernen.

HTML-Befehle beginnen immer mit einem mathematischen »kleiner«-Zeichen und enden mit einem mathematischen »größer«-Zeichen.

```
<br>
```

Wir werden diese beiden Zeichen »spitze Klammern« nennen. Die HTML-Befehle werden auch Tags genannt (englisch ausgesprochen »Tägs«). Der `br`-Tag bewirkt einen Zeilenumbruch (`br` steht für line **break**).

In XHTML werden bei allein stehenden Tags am Ende ein Leerzeichen und ein Schrägstrich eingefügt. Der `br`-Tag wird also in XHTML folgendermaßen geschrieben:

```
<br />
```

Die meisten HTML-Befehle treten paarweise auf:

Ein Stück ``fett gedruckter`` Text

Der `b`-Tag bewirkt Fettdruck (`b` steht für **bold**). Die Wirkung des Befehls beginnt mit dem Start-Tag `` und endet mit dem Ende-Tag ``.

In der Tabelle 3.1 sehen Sie weitere Format-Tags, die ebenfalls alle paarweise auftreten.

| Tag | Erklärung | Beschreibung |
|-------------------------|-------------|--|
| <code>i</code> | italic | kursiv |
| <code>sub</code> | subscript | tiefgestellt |
| <code>sup</code> | superscript | hochgestellt |
| <code>kbd</code> | keyboard | Computertext (Nicht-Proportionalschrift) |
| <code>p</code> | paragraph | Absatz |
| <code>blockquote</code> | block quote | Zitat als eigener Absatz, mit Einzug |
| <code>h1</code> | heading 1 | Haupt-Überschrift |
| <code>h2</code> | heading 2 | Überschrift 2. Ordnung |

Tabelle 3.1: Diverse Format-Tags

Formatieren Sie jetzt mit den Tags aus Tabelle 3.1 Ihren Text aus dem letzten Kapitel und sehen Sie sich das Ergebnis in Ihrem Webbrowser an.

Sie können verschiedene Tags auch miteinander kombinieren. Dabei müssen Sie darauf achten, die Tags richtig ineinander zu verschachteln.

Zuerst `dick`, dann auch noch `<i>schräg</i>`.

Wichtig ist, dass zuerst der innere `i`-Tag geschlossen wird und dann erst der `b`-Tag.

Zeilenumbrüche, Leerzeichen und Tabulatoren werden vom Webbrowser ignoriert. Wenn Sie also auf Ihrer Webseite einen Zeilenumbruch brauchen, müssen Sie den `br`-Tag verwenden. Normale Zeilenumbrüche in Ihrer HTML-Datei werden vom Webbrowser ignoriert, egal wie viele davon Sie einfügen.

WAS IST DAS?

Als **Whitespace** wird in der Programmierung ein Stück Text bezeichnet, das nur aus unsichtbaren Zeichen besteht: Leerzeichen, Tabulatoren und Zeilenumbrüche.

In den meisten Programmiersprachen wird – wie in HTML – Whitespace einfach ignoriert. Ein Leerzeichen bedeutet daher dasselbe wie zehn Leerzeichen oder 40 Zeilenumbrüche.

Für Sonderzeichen gibt es in HTML die Schreibweise `&sonderzeichen;`. In Tabelle 3.2 sehen Sie ein paar Beispiele.

| HTML Schreibweise | Zeichen | HTML Schreibweise | Zeichen |
|-------------------------|---------|---|-------------------------|
| <code>&copy;</code> | © | <code>&uuml;</code> <code>&ouml;</code> <code>&auml;</code> | ü, ö, ä |
| <code>&lt;</code> | < | <code>&Uuml;</code> <code>&Ouml;</code> <code>&Auml;</code> | Ü, Ö, Ä |
| <code>&gt;</code> | > | <code>&nbsp;</code> | erzwungenes Leerzeichen |
| <code>&amp;</code> | & | <code>&szlig;</code> | ß |

Tabelle 3.2: HTML-Sonderzeichen

Sie können die Umlaute und das scharfe ß auch ganz normal in den HTML-Code eintippen. Die Sonderzeichen-Schreibweise ist für Leute gedacht, die

keine deutsche Tastatur haben. Umgekehrt können Sie mit `ø` ein `ø` erzeugen, auch wenn es auf Ihrer Tastatur nicht vorgesehen ist.

Viele Web-Editoren verwenden immer diese besondere Schreibweise für Sonderzeichen. Das kann die Lesbarkeit des HTML-Codes sehr erschweren.

Es `grü`nt so `grü`n, wenn Spaniens `Blü`ten `blü`hen.

An diesem Beispiel können Sie erkennen, dass HTML von einem Engländer erfunden wurde. Das englische Original sieht wesentlich einfacher aus:

The rain in spain falls mainly in the plain.

Fehler im HTML-Code

Wenn Sie einen Web-Editor zur Erstellung Ihrer Webseiten verwenden, ist dieser dafür verantwortlich, korrekten HTML-Code zu erzeugen. Wenn Sie den Code aber selber schreiben, werden sich hin und wieder Fehler einschleichen. Was passiert also, wenn Sie einen Tag falsch eintippen? Stürzt Ihr Webbrowser ab oder wird gar die automatische Selbstzerstörung des Computers eingeleitet? Nein, natürlich nicht. Das Ganze ist harmlos. Meist erscheint nur ein Teil der Webseite nicht. Sehen wir uns ein paar häufig vorkommende Fehler an:

Ich vergesse immer die ``Türe hinter mir zu schließen.

In diesem HTML-Code fehlt der schließende `b`-Tag. Somit wird alles nach dem beginnenden `b`-Tag fett gedruckt – bis zum Ende der Webseite.

Auch die `<em Kaffeemaschine vergesse ich abzuschalten`.

Hier fehlt die schließende spitze Klammer. Der Webbrowser, der die Seite anzeigen soll, glaubt nun, dass das Wort `Kaffeemaschine` noch Teil des HTML-Befehls ist. Deswegen wird das Wort (und der Rest der Webseite) nicht am Bildschirm angezeigt.

Ich bin ja `<wackelt>`soooo`</wackelt>` kreativ.

Nehmen wir an, Sie sind ein sehr kreativer Mensch und erfinden gerne neue Dinge, in diesem Beispiel einen neuen HTML-Tag. Was macht nun der Webbrowser mit diesem ihm unbekanntem Tag? Er wird ihn einfach ignorieren und den Text so behandeln, als ob der Tag nicht vorhanden wäre.

Ich bin ja soooo kreativ.

Dieses Ignorieren von unbekanntem Tags war sehr wichtig für die Weiterentwicklung des World Wide Web. So konnten immer wieder neue Tags einge-

führt werden, ohne dass ältere Webbrowser Seiten mit diesen neuen Tags nicht mehr darstellen konnten oder ständig Fehlermeldungen produzierten.

Attribute

Manche HTML-Tags enthalten zusätzliche Information in so genannten Attributen. Sie werden innerhalb des HTML-Tags eingefügt und haben immer die Form `name="wert"`.

```

```

Der `img`-Tag fügt ein Bild in die Seite ein (**image**). Mit dem Attribut `src` (**Source**, Quelle) können Sie angeben, in welcher Datei das Bild gespeichert ist. Im `img`-Tag können Sie noch diverse andere Attribute angeben.

```

```

`width` und `height` (Breite und Höhe), in Pixel gemessen, geben die Größe des Platzes an, der in der Webseite für das Bild reserviert werden soll. Falls das Bild größer oder kleiner ist als hier angegeben, wird es vom Webbrowser verzerrt dargestellt. Da diese Verzerrung die Bilder meist recht unansehnlich macht, sollten Sie darauf achten, immer die korrekte Größe des Bildes anzugeben.

Die Reihenfolge der Attribute ist egal. Whitespace wird auch hier ignoriert. Sie können obigen `img`-Tag mit Attributen also genauso gut so schreiben:

```

```

Das World Wide Web wird erst durch Links zu anderen Webseiten zum Web (Spinnennetz). Links sind daher sehr wichtig. Vielleicht wurde deshalb für den entsprechenden Tag der erste Buchstabe des Alphabets verwendet. (Die Erklärung ist zwar gut, aber leider falsch. Das `a` in diesem Tag steht für das englische Wort **anchor** und bedeutet Anker.)

Ich empfehle `Perl`.

Der Text zwischen Start- und Ende-Tag (hier also Perl) wird auf der Webseite angezeigt, meist blau und unterstrichen. Das Attribut `href` gibt an, wohin der Link zeigen soll.

Bei einem Link kann entweder auf die ganze Seite (wie oben) oder auf eine bestimmte Textmarke in der Seite verwiesen werden.

```
<a href="http://webwelt.horus.at/a.html#ort">am Ort</a>
```

Dieser Link verweist auf eine Webseite mit dem Dateinamen *a.html* und in dieser Webseite auf die Textmarke mit dem Namen *ort*. Die Textmarke muss in der Webseite vorhanden sein. Sie wird ebenfalls mit dem *a*-Tag, aber mit dem *name*-Attribut gesetzt:

Bildung und Kultur am `Ort`

ACHTUNG

Die Textmarke ist unsichtbar. Das Wort »Ort«, das sich innerhalb des *a*-Tags befindet, wird nicht anders formatiert als die anderen Wörter. Ein Text innerhalb des *a*-Tags wird nur blau und unterstrichen, wenn das *href*-Attribut gesetzt ist. Ein Link hin zu einer Textmarke wird auch als interner Link bezeichnet.

Mit dem *font*-Tag können Sie Schriftart, Schriftgröße und Farbe des Textes beeinflussen. Die Farbe wird mit einem komplizierten Code angegeben:

```
<font face="Arial" color="#FF0000" size="+6">Hi!</font>
```

Welche Attribute ein bestimmter Tag haben kann, ist festgelegt. Sie können nicht einfach ein Attribut im falschen Tag verwenden, solche Attribute werden ignoriert.

```
<font family="Arial" size="+6">Hi!</font>
```

Es gibt kein Attribut *family* im *font*-Tag, dieses Attribut wird ignoriert. Die Wirkung des anderen Attributs bleibt bestehen:

```
<font size="+6">Hi!</font>
```

Gesamtstruktur

In diesem Kapitel haben Sie bisher nur unvollständige Webseiten erstellt. Der Webbrowser ist aber so tolerant, diese Seiten trotzdem korrekt darzustellen. Von nun an sollten Sie immer vollständige HTML-Dokumente erstellen.

Listing 3.1: Gesamtstruktur eines HTML-Dokuments

```
<html>
<head>
  <title>HTML Übungsseite</title>
</head>
<body>
  <p>Hier ist der Inhalt der Webseite... </p>
</body>
</html>
```

Ein HTML-Dokument beginnt und endet mit dem `html`-Tag. Innerhalb des `html`-Tags gibt es einen `head`- und einen `body`-Tag. Alle Tags, die Sie bisher kennen gelernt haben, können Sie (nur) innerhalb des `body`-Tags verwenden. Der `Body` entspricht dem »sichtbaren« Bereich einer Webseite.

Der »Head« des Dokumentes enthält zusätzliche Informationen, die oft nicht direkt angezeigt werden. Den Titel des Dokumentes können Sie in der Titelzeile des Webbrowsers sehen.

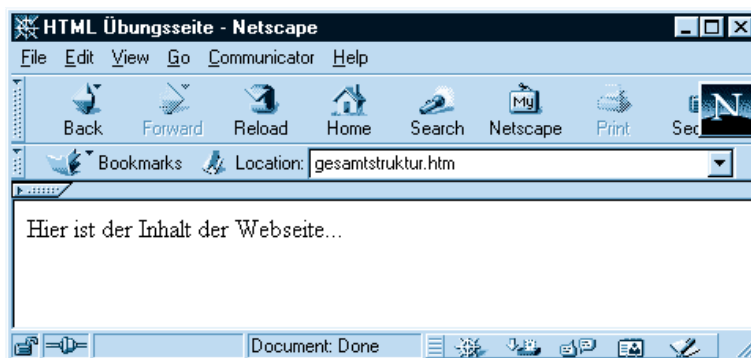


Abbildung 3.3: Webseite mit Titel, angezeigt in Netscape

In den `Head` des HTML-Dokumentes gehören auch die Tags mit Informationen für Suchmaschinen.

Listen

In HTML gibt es zwei einfache Listen, um Aufzählungen und nummerierte Listen zu erzeugen. Sie brauchen jeweils zwei verschiedene Arten von HTML-Tags. Ein Tag bezeichnet den Anfang und das Ende der Liste, der andere Tag kennzeichnet die einzelnen Listenelemente.

Der HTML-Code für die Aufzählung

- Rot
- Grün
- Blau

würde folgendermaßen aussehen:

```
<ul>
  <li>Rot</li>
  <li>Grün</li>
  <li>Blau</li>
</ul>
```

Für jeden Listenpunkt gibt es einen `li`-Tag. Der `ul`-Tag umspannt alle Listenpunkte. `ul` steht für »unorderd list« – ungeordnete Liste. Wenn Sie statt des `ul`-Tags den `ol`-Tag verwenden und sonst alles gleich lassen, erhalten Sie eine »ordered list« – eine geordnete Liste:

1. Rot
2. Grün
3. Blau

```
<ol>
  <li>Rot</li>
  <li>Grün</li>
  <li>Blau</li>
</ol>
```

Tabellen

Eine HTML-Tabelle wird aus drei verschiedenen verschachtelten HTML-Tags erstellt. Der `table`-Tag bezeichnet die ganze Tabelle, der `tr`-Tag (**table row**) eine einzelne Zeile und der `td`-Tag (**table data**) eine einzelne Zelle:

```
<table border="1">
  <tr>
    <td>Rot</td>
    <td>Grün</td>
    <td>Blau</td>
  </tr>
  <tr>
    <td>30</td>
    <td>10</td>
  </tr>
</table>
```

```
<td>25</td>  
</tr>  
</table>
```

Diese Tabelle hat zwei Zeilen und drei Spalten. Für die Spalten gibt es keinen eigenen Tag. Diese ergeben sich aus den td-Tags der einzelnen Zeilen. Um eine Spalte einzufügen, müssen Sie daher in jeder Zeile einen td-Tag einfügen.

```
<table border="1">  
  <tr>  
    <td>Rot</td>  
    <td>Grün</td>  
    <td>Blau</td>  
    <td>Gelb</td>  
  </tr>  
  <tr>  
    <td>30</td>  
    <td>10</td>  
    <td>25</td>  
    <td>110</td>  
  </tr>  
</table>
```

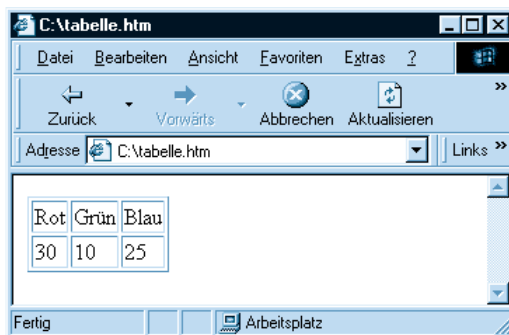


Abbildung 3.4: Tabelle mit zwei Zeilen und drei Spalten

Die Schreibweise mit vielen Zeilenumbrüchen und den zusätzlichen Leerzeichen vor den »inneren« Tags bei Verschachtelungen ist nicht zwingend vorgeschrieben. Sie machen aber das Verstehen des Codes wesentlich einfacher. Dieselbe Tabelle könnten Sie auch so schreiben:

```
<table border="1"> <tr>           <td>Rot</td>  
<td>Grün</td>           <td>Blau</td>           </tr>   <tr>  
<td>30</td>           <td>10</td>           <td>25</td>           </<br>  
<tr> </table>
```

Bei dieser Schreibweise ist das Einfügen einer zusätzlichen Spalte allerdings eine Herausforderung.

Formulare

Vielleicht haben Sie ja bereits viele Webseiten mit einem Formular erstellt. In diesem Buch werden Sie lernen, wie Sie die Daten aus dem Formular weiterverarbeiten können. Zuerst aber werden wir die Erstellung eines Formulars im HTML-Code genauer betrachten.

Die beiden wichtigsten Tags für Formulare sind der `form`-Tag, der das ganze Formular umspannt, und der `input`-Tag, der für fast alle Eingabefelder verwendet wird.

```
<form action="http://irgendwo.im.net/cgi-bin/tuwas.cgi">
  Vorname: <input type="text" name="vorname" />
  <br />Beilage:
  <input type="checkbox" name="kartoffel" /> Kartoffel
  <input type="checkbox" name="Brokkoli" /> Brokkoli
  <input type="checkbox" name="Erbsen" /> Erbsen
  <br />Bezahlung:
  <input type="radio" name="zahl" value="bar" />
    Bargeld
  <input type="radio" name="zahl" value="wasch" />
    Abwasch
  <br /><input type="submit" value="Abschicken" />
</form>
```

Das `action`-Attribut des `form`-Tags enthält die Adresse des Programms, das die Daten weiterverarbeitet wird. Bald werden Sie dieses Programm selbst schreiben können.

Für die diversen Eingabefelder müssen Sie `input`-Tags verwenden. Diese besitzen je nach gewähltem `type`-Attribut unterschiedliches Aussehen und Funktionalität. Abbildung 3.5 zeigt das vorher gezeigte HTML-Formular im Webbrowser, das Formular ist schon ausgefüllt.

Textfelder (`<input type="text" />`) sind – wie der Name schon sagt – für die Eingabe von Text gedacht. In unserem Beispielformular muss der Vorname in ein Textfeld eingegeben werden.

Eine **Checkbox** (`<input type="checkbox" />`) wird für einzelne Fragen verwendet, die mit Ja oder Nein zu beantworten sind, hier etwa die Wahl

der Beilagen. Die einzelnen Punkte können unabhängig voneinander gewählt oder abbestellt werden.

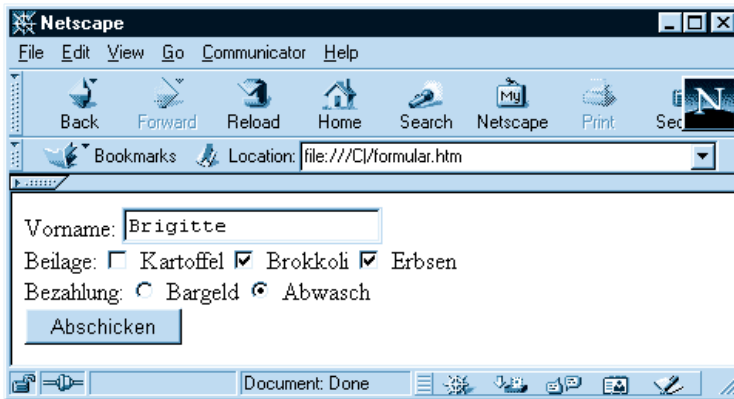


Abbildung 3.5: Web-Formular

Sowohl Textfelder als auch Checkboxes haben jeweils einen eigenen Namen, der mittels des Attributs `name` eingestellt wird.

Radiobuttons (`<input type="radio" />`) – auch Optionsschalter genannt – treten immer in Gruppen auf. Es kann jeweils nur eine Option aus der Gruppe ausgewählt werden, also entweder Bargeld oder Abwasch, nicht aber beides. Alle Radiobuttons in der Gruppe haben denselben Namen aber verschiedene Werte im Attribut `value`.

Die **Schaltfläche zum Einsenden** (`<input type="submit" />`) des Formulars ist einfach, aber sehr wichtig. Diese Schaltfläche braucht keinen Namen. Sie müssen das `value`-Attribut zur Beschriftung verwenden.

Für die Auswahl aus einer Liste können Sie ein in Abbildung 3.6 gezeigtes **Dropdown-Menü** verwenden.



Abbildung 3.6: Web-Formular: Dropdown-Menü

Bei einem Dropdown-Menü können Sie aus vorgegebenen Antworten auswählen, es ist keine andere Eingabe möglich. Der HTML-Code erfordert zwei Tags: `select` und `option`.

```
<select name="zahl">
  <option>Bargeld</option>
  <option>Mastercard</option>
  <option>Visacard</option>
  <option>Eurocheque</option>
  <option>Abwaschen</option>
</select>
```

Das Dropdown-Menü entspricht in seiner Funktionalität einer Gruppe von Radiobuttons. Das Dropdown-Menü braucht allerdings weniger Platz auf der Webseite. Bei den Radiobuttons sehen Sie alle Auswahlmöglichkeiten sofort – ohne zusätzlichen Klick. Welche der beiden Möglichkeiten gerade besser ist, müssen Sie im Einzelfall entscheiden.

Auch ein Bild kann als Eingabefeld verwendet werden.

```
<input type="image" name="wo" src="karte.gif" />
```

Das Bild wird auf der Webseite angezeigt. Wenn Sie auf eine bestimmte Stelle des Bildes klicken, werden die **Koordinaten** des Klicks an das Programm geliefert.

Ebenfalls sehr selten werden unsichtbare Formular-Felder verwendet. Von diesem Feld gibt es keine Abbildung.

```
<input type="hidden" name="passwort" value="rosebud" />
```

Damit haben Sie alle Formularelemente kennen gelernt, die wir in den Programmierbeispielen verwenden werden.

Imagemap

In einem Link kann statt des sichtbaren Textes auch ein Bild verwendet werden. Das wird gerne für Navigationselemente verwendet:

```
<a href="home.html">Zurück zur Homepage</a>
<a href="home.html"></a>
```

Wenn es wichtig ist, **wo** auf das Bild geklickt wurde, dann müssen Sie statt eines einfachen Links eine so genannte *Imagemap* verwenden. Imagemaps sind mit WYSIWIG-Editoren sehr einfach zu erstellen, in Abbildung 3.7 sehen Sie die Erstellung einer Imagemap im Programm *Dreamweaver*.

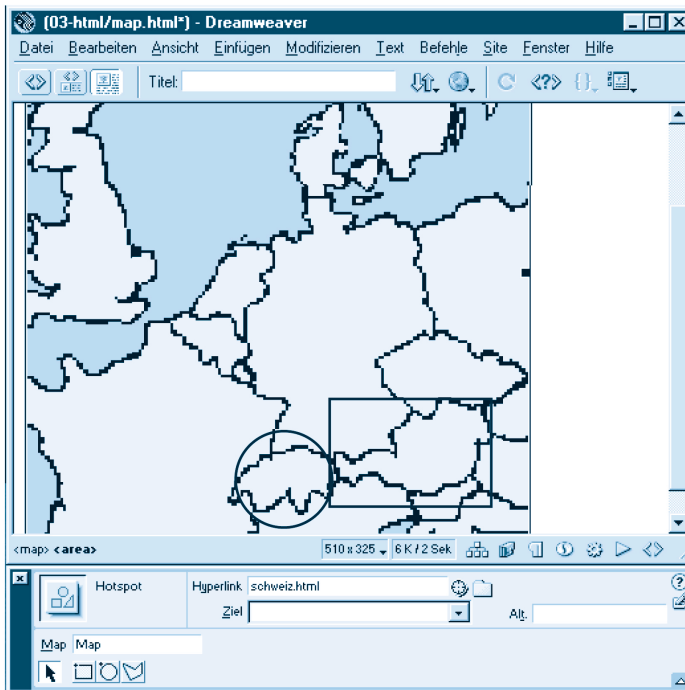


Abbildung 3.7: Imagemap in Dreamweaver

Der HTML-Code einer Imagemap ist relativ kompliziert, da mit Koordinaten gearbeitet wird:

```

<map name="europa">
  <area shape="rect" coords="146,333,232,396" href="#">
  <area shape="circle" coords="291,343,58" href="#">
</map>
```

Im `img`-Tag wird mit dem `usemap`-Attribut der Name der Imagemap angegeben. Die Imagemap besteht aus mehreren `area`-Tags, die von einem `map`-Tag zusammengehalten werden.

Kommentare

Ein Kommentar ist ein Text im HTML-Code, der nichts zur Darstellung der Seite beiträgt. Ein Beispiel:

```
<!-- hier wollt ich noch mehr schreiben ... -->
```

Dieser Text erscheint nicht auf der Webseite und beeinflusst auch nicht die Formatierung der Seite. Die Kommentare sehen also nur Leute, die den HTML-Code lesen.

Mit Kommentaren können Sie Teile der Seite vorübergehend verschwinden lassen, ohne sie wirklich zu löschen. Das nennt man *auskommentieren*.

```
<p>Danke für Ihren Besuch!</p>
<!--
<p>Schreiben Sie etwas in mein
<a href="gaestebuch.pl">Gästebuch</a></p>
-->
```

Der Link zum Gästebuch ist spurlos verschwunden. Das ist praktisch, wenn Sie gerade das Gästebuch umbauen. Wenn Sie mit dem Umbau fertig sind, brauchen Sie nur die Kommentar-Zeichen zu entfernen und der Link erscheint wieder:

```
<p>Danke für Ihren Besuch!</p>
<p>Schreiben Sie etwas in mein
<a href="gaestebuch.pl">Gästebuch</a></p>
```

Lehren aus HTML

Vieles, was Sie anhand von HTML gelernt haben, wird Ihnen bei der Perl-Programmierung weiterhelfen. Es gibt aber auch Unterschiede. Hier sind noch einmal die wichtigsten Punkte kurz zusammengefasst:

- Tags/Befehle müssen genau richtig geschrieben werden:
Falsch geschriebene HTML-Tags werden stillschweigend ignoriert. Ein falscher Befehl in einem Perl-Programm führt zu einer Fehlermeldung des Perl-Interpreters.
- Whitespace:
In HTML und in Perl wird Whitespace (fast immer) wie ein einzelnes Leerzeichen behandelt. Zusätzliche Leerzeichen und Zeilenumbrüche werden nicht verwendet, um die Webseite/das Programm zu verändern, sondern um diese besser lesbar zu machen.
- Groß-/Kleinschreibung:
Im HTML-Code können Sie Tags groß oder klein schreiben, das Ergebnis ist dasselbe. In Perl ist das nicht so! Der Perl-Interpreter unterscheidet zwischen Groß- und Kleinschreibung.

- Kommentare

Kommentare haben keine Auswirkung auf die Darstellung der Webseite/ die Ausführung des Programms. Sie werden verwendet für Erläuterungen oder um Teile vorübergehend auszukommentieren (verschwinden zu lassen).

- Ineinander verschachteln:

In HTML haben Sie gelernt, Tags korrekt ineinander zu verschachteln. Es ist wichtig, zuerst den inneren Tag zu schließen und dann erst den äußeren. Dasselbe gilt für Perl. Auch hier müssen Anführungszeichen und Klammern in der richtigen Reihenfolge geschlossen und Befehle richtig verschachtelt werden.

In HTML und in Perl ist es üblich, alle Zeilen verschachtelter Befehle innerhalb der Verschachtelung einzurücken.

- Plattform-unabhängig:

HTML und Perl funktionieren auf UNIX, Windows, Apple und noch einigen anderen Systemen gleich.

Mit so viel Vorwissen wird Ihnen der Einstieg ins Programmieren leicht fallen! Im nächsten Kapitel geht es jetzt endlich richtig los.