HTML – Ausgewählte Referenz (XHTML)

HTML-Grundgerüst

Der gesamte Inhalt einer HTML-Datei wird in die Tags <html> bzw. </html> eingeschlossen. Das html-Element wird auch als Wurzelelement einer HTML-Datei bezeichnet. Hinter dem einleitenden HTML-Tag folgt das einleitende Tag für den Kopf <head>. Zwischen diesem Tag und seinem Gegenstück </head> werden Tags notiert. Die wichtigste dieser Angaben ist der Seitentitel, markiert durch <title> bzw. </title>. Unterhalb davon folgt der Textkörper, markiert durch <body> bzw. </body>. Dazwischen wird dann der eigentliche Inhalt der Datei notiert, also das, was im Anzeigefenster des WWW-Browsers angezeigt werden soll.

HTML Grundgerüst

```
<html>
<head>
<title>Titel</title>
</head>
<body>
 <!-- Inhalt der Datei -->
</body>
</html>
```

Für Titel einen aussagekräftigen Titel vergeben! Den gesamten sichtbaren Inhalt der Datei zwischen <body> und </body> notieren.

Dokumenttyp-Angabe – neuester Standard XHTML

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

Die letzte Zeile ersetzt den normalen öffnenden <html> - Tag.

Mit der Dokumenttyp-Angabe bestimmen Sie, welche Auszeichnungssprache in welcher Version Sie verwenden. Eine auslesende Software, etwa ein Web-Browser, kann sich an dieser Angabe orientieren. Die Regeln für HTML sind mit Hilfe von SGML formuliert, die Regeln für XHTML mit Hilfe von XML. Nach den Regeln einer SGML- oder XML-basierten Auszeichnungssprache ist eine HTML-Datei oder eine XHTML-Datei erst dann eine gültige (valide) Datei, wenn sie einen bestimmten Dokumenttyp angibt und sich dann innerhalb des restlichen Quelltextes genau an die Regeln hält, die in diesem Dokumenttyp definiert sind. Nur so lässt sich das Konzept der software-unabhängigen, aber regelgerechten Dateiformate konsequent durchsetzen.

Auch für XHTML 1.0 gibt es die drei Varianten "Strict", "Transitional" und "Frameset". Im Mittelteil der Dokumenttyp-Angabe muss jedoch bei Version 1.0 von XHTML XHTML 1.0 notiert werden. Auch die Web-Adressen für den Bezug sind andere als bei HTML. Es gibt auch eine Sprachversion 1.1 von XHTML, die intern anders organisiert ist als XHTML 1.0. In XHTML 1.1 gibt es keine Sprachvarianten mehr. XHTML 1.1 entspricht nur noch der Sprachvariante "Strict" von XHTML 1.0.

Kommentare

```
<!-- Dies ist ein Kommentar -->
```

Zum internen Auskommentieren von Inhalten oder Befehlen. Wird im Browser nicht angezeigt.

HTML-Kopfdaten

Meta-Angaben

Allgemeines zu Meta-Angaben
In Meta-Angaben können Sie verschiedene nützliche Anweisungen für Web-Server, Web-Browser und automatische Suchprogramme im Internet ("Robots") notieren. Meta-Angaben können Angaben zum Autor und zum Inhalt der Datei enthalten. Sie können aber auch HTTP-Befehle absetzen, zum Beispiel zum automatischen Weiterleiten des Web-Browsers zu einer anderen Adresse.

Für jede Meta-Angabe notieren Sie ein Meta-Tag im HTML-Dateikopf. Es ist also kein Problem, mehrere Meta-Tags zu notieren.

Meta-Angabe: Autor

<meta name="author" content="Name" />

Für Name den Namen des Urhebers notieren.

Meta-Angabe: Beschreibung

<meta name="description" content="Text" />

Für Text eine Kurzbeschreibung des Inhalts notieren.

Meta-Angabe: Stichwörter

<meta name="keywords" content="Wort, Wort, Wort" />

Für Wort je ein Stichwort zum Inhalt notieren. Anzahl ist flexibel. Stichwörter durch Kommata trennen.

Meta-Angabe: Zeitstempel

<meta name="date" content="yyyy-MM-ddThh:mm:ss+hh:mm" />

yyyy = Jahr, MM = Monat, dd = Tag, T = fixer Buchstabe, hh = Stunden, mm = Minuten, ss = Sekunden, +/- = Abweichung gegenüber Greenwich in Stunden und Minuten.

Meta-Angabe: Robots

<meta name="robots" content="index|noindex|follow|nofollow" />

Anweisung an Suchmaschinen:
index = Auslesen erlaubt,
noindex = Auslesen nicht erlaubt,
follow = Verweisen folgen erlaubt,
nofollow = Verweisen folgen nicht erlaubt.

Meta-Angabe: Zeichensatz

<meta http-equiv="content-type" content="Mime-Type; charset=Zeichensatz" />

Für Mime-Type so etwas wie text/html und für Zeichensatz so etwas wie ISO-8859-1 - das entspricht dem westeuropäischen Zeichensatz - notieren.

Meta-Angabe: Originaladresse

<meta http-equiv="expires" content="0" />

Zeitpunkt angeben, ab dem von Originaladresse geladen werden soll. 0 bedeutet: immer von Originaladresse laden (Proxy-Server ignorieren).

Häufig abgerufene Web-Seiten werden im Web auf so genannten Proxy-Servern zwischengespeichert. Das ist dann ein so genannter Proxy-Cache. Auch Browser speichern aufgerufene Seiten, und zwar lokal auf dem Rechner des Anwenders. Dabei spricht man vom Browser-Cache. Die Cache-Speicher sparen in vielen Fällen Leitungswege und Ressourcen. Ein Nachteil ist jedoch, dass dem Anwender möglicherweise Daten angezeigt werden, die gar nicht mehr aktuell sind, weil auf der Original-Adresse mittlerweile neue Daten liegen. Sie können mit Hilfe einer Meta-Angabe erzwingen, dass die Daten nicht aus einem Cache-Speicher serviert werden, sondern vom Original-Server. Zu empfehlen ist diese Angabe, wenn Sie die Daten einer HTML-Datei häufig ändern und neu ins Web hochladen.

Meta-Angabe: Weiterleitung

<meta http-equiv="refresh" content="0; URL=URI" />

Angeben, wann und welche Adresse automatisch geladen werden soll. 0 bedeutet: sofort (Angabe bedeutet Verzögerung in Sekunden).
Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad angeben, das nach der angegebenen Zeit automatisch geladen werden soll.

Verlassen Sie sich nicht darauf, dass diese Angabe funktioniert. Wenn Sie etwa eine Weiterleitung zu einer neuen Adresse eingeben, notieren Sie am besten auf der gleichen Seite zur Sicherheit noch einen normalen Verweis zu der neuen Adresse. So finden auch Anwender, bei denen die Meta-Angabe nicht funktioniert, auch über den Verweis den Weg zu Ihrer neuen Heimat.

Dateiweite Einstellungen

Farben für Text, Hintergrund und Verweise

<body bgcolor="#XXXXXX" text="#XXXXXX" link="#XXXXXX" vlink="#XXXXXX" alink="#XXXXXX">

bgcolor = dateiweite Hintergrundfarbe,
text = dateiweite Schriftfarbe,
link = dateiweite Farbe für Verweise zu noch nicht besuchten Zielen,
vlink = dateiweite Farbe für Verweise zu bereits besuchten Zielen,
alink = dateiweite Farbe für aktivierte Verweise.
Für #XXXXXX eine hexadezimal notierte RGB-Farbe angeben oder einen erlaubten Farbnamen.

Hintergrundbild (Wallpaper)

<body background="URI">

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Grafikdatei angeben.
Das Attribut background= ist als veraltet eingestuft und soll künftig vermieden werden. Den gleichen Effekt erreichen Sie nämlich auch mit Hilfe von CSS , z.B. so:
<body style="background-image:url(background.jpg)">

Elemente zur Textstrukturierung

Überschriften

<h[1-6]>Text</h[1-6]>

Sechs Überschriftenebenen sind erlaubt, also z.B. <h1>Text</h1>.

Überschriften ausrichten

<h[1-6] align="Ausrichtung">Text</h[1-6]>

Für Ausrichtung einen der folgenden Werte notieren:
left = linksbündig | center = zentriert | right = rechtsbündig | justify = Blocksatz

Nicht alle Browser beherrschen den Blocksatz.
align ist als veraltet (missbilligt) eingestuft. Statt dessen wird empfohlen, CSS Stylesheets zu benutzen, z.B.: ... .

Textabsätze

Text

Alleinstehende -Tags, wie sie früher mal zulässig waren, sind mittlerweile nicht mehr HTML-gerecht. Notieren Sie Textabsätze immer mit einleitendem und abschließendem Tag.
Das -Element darf keine anderen blockerzeugenden Elemente wie z.B. Überschriften, Textabsätze, Listen, Zitate oder Adressen enthalten.

Textabsätze ausrichten

Text

Siehe „Überschriften ausrichten“

Zeilenumbruch erzwingen

Text alte Zeile
Text neue Zeile

Text innerhalb von normalen Absätzen, Listen, sowie in Überschriften oder Tabellenzellen wird vom Web-Browser bei der Anzeige automatisch umgebrochen. Sie können jedoch an einer gewünschten Stelle einen Zeilenumbruch erzwingen.

Aufzählungslisten

Listeneintrag
Listeneintrag

 leitet eine Aufzählungsliste ein (ul = unordered list = unsortierte Liste). Mit <li> beginnt ein neuer Punkt innerhalb der Liste (li = list item = Listeneintrag). beendet den Listeneintrag. Nach XHTML-Standard ist abschließende -Tag erforderlich. beendet die Liste.
Wie das Bullet dargestellt wird, bestimmt dabei der Web-Browser.
Das Verschachteln von Listen ist ebenfalls möglich.

Bullet-Typ für Aufzählungslisten

<ul type="Typ">
Listeneintrag
Listeneintrag

Für Typ einen der folgenden Werte notieren:
circle = Kreis/Punkt | disc = Scheibe/Dateisymbol | square = Quadrat

Diese HTML-Attribute sind mittlerweile als veraltet gekennzeichnet und sollen künftig aus dem HTML-Standard entfallen. Statt dessen sollten Sie CSS verwenden.

Nummerierte Listen

Listeneintrag
Listeneintrag

 leitet eine nummerierte Liste ein (ol = ordered list = nummerierte Liste). Mit <li> beginnt ein neuer Punkt innerhalb der Liste (li = list item = Listeneintrag). beendet den Listeneintrag. Nach XHTML-Standard ist abschließende -Tag erforderlich. beendet die Liste.

Automatische Nummerierungshierarchien wie 1, 1.1, 1.1.1, sind in HTML nicht möglich.

Nummerierte Listen alphabetisch oder römisch

<ol type="Typ">
Listeneintrag
Listeneintrag

Für Typ einen der folgenden Werte notieren:
a = für klein alphabetisch | A = für groß alphabetisch | i = für klein römisch | I = für groß römisch.

Diese HTML-Attribute sind mittlerweile als veraltet gekennzeichnet und sollen künftig aus dem HTML-Standard entfallen. Statt dessen sollten Sie CSS verwenden .

Nummerierung beeinflussen

<ol start="Startwert">
Listeneintrag
<li value="Fortsetzungswert">Listeneintrag

Für Startwert die Zahl mit dem gewünschten Startwert notieren, für Fortsetzungswert die Zahl mit dem gewünschten Fortsetzungswert innerhalb der Liste.

Definitionslisten (Glossarlisten)

<dl>
<dt>Zu definierender Ausdruck</dt>
<dd>Definition des Ausdrucks</dd>
</dl>

Definitionslisten sind für Glossare gedacht. Glossare bestehen aus einer Liste von Einträgen. Die Einträge eines Glossars bestehen aus einem zu definierenden Ausdruck (z.B. ein Fachbegriff) und der zugehörigen Definition.
<dl> leitet eine Definitionsliste ein (dl = definition list = Definitionsliste).
<dt> leitet einen zu definierenden Ausdruck ein (dt = definition (list) term = Ausdruck in der Definitionsliste).
<dd> leitet eine Definition eines Ausdrucks ein (dd = definition (list) definition = Definition in der Definitionsliste). </dl> beendet die Liste.

Zitate

<blockquote>Text</blockquote>

Sie können Zitate von Fremdautoren in einem eigenen, anders formatierten (zumeist eingerückten) Absatz hervorheben. Es handelt sich dabei jedoch um eine logische, inhaltliche Auszeichnung. Wie diese Absätze genau formatiert werden, bestimmt letztlich der Web-Browser.

Adressen

<address>Text</address>

Sie können Internet-Adressen von Personen oder Dateien in einem eigenen, anders formatierten (zumeist kursiv dargestellten, eingerückten) Absatz hervorheben. Auch dies ist eine logische Textauszeichnung, für deren tatsächliche Formatierung bei der Ausgabe es keine festen Vorschriften gibt.

Präformatierter Text

<pre>Text</pre>

Beispielsweise für Programmlistings ist es wichtig, dass sie in dicktengleicher Schrift dargestellt werden, und dass Einrückungen so wiedergegeben werden, wie sie beim Editieren eingegeben wurden. Zu diesem Zweck bietet HTML die Möglichkeit der "präformatierten Textabschnitte" an. Auch wenn Sie tabellarische Daten darstellen müssen und auf Tabellen verzichten wollen, können Sie präformatierten Text benutzen. Und dann ist präformatierter Text auch noch dazu geeignet, um andere Elemente, beispielsweise Grafiken, auszurichten.

In welcher Schriftart und Schriftgröße präformatierter Text genau dargestellt werden, darauf haben Sie mit HTML keinen Einfluss. Die Browser benutzen Default-Formatierungen. Mit CSS können Sie ein solches Element jedoch nach Wunsch formatieren.

Logische Textauszeichnung (inline)

In HTML gibt es logische und physische Elemente zur Auszeichnung von Text. Bei logischen Elementen entscheidet der Web-Browser, wie ein solcher Text hervorgehoben wird (z.B. fett, kursiv oder andersfarbig). In Verbindung mit CSS Stylesheets können Sie logische Textauszeichnungen allerdings nach Wunsch formatieren. Im Unterschied zu Elementen wie Überschriften, Textabsätzen, Listen usw., die ja auch der logischen Strukturierung von Text dienen, sind die hier beschriebenen Elemente jedoch so genannte Inline-Elemente , während Überschriften, Textabsätze, Listen usw. als Block-Elemente bezeichnet werden. Inline-Elemente erzeugen keinen Absatz (genauer: keine neue Zeile) im Textfluss.

betont : zeichnet einen Text aus mit der Bedeutung "emphatisch"

stark betont : zeichnet einen Text aus mit der Bedeutung "stark betont"

<code>Quellcode</code> : zeichnet einen Text aus mit der Bedeutung "dies ist Quelltext"

<samp>Beispiel</samp> : zeichnet einen Text aus mit der Bedeutung "Dies ist ein Beispiel"

<kbd>Tastatureingabe</kbd> : zeichnet einen Text aus mit der Bedeutung "das ist eine Tastatureingabe"

<var>Variable</var> : zeichnet einen Text aus mit der Bedeutung "dies ist eine Variable oder ein variabler Name"

<cite>Zitat</cite> : zeichnet einen Text aus mit der Bedeutung "dies ist ein Zitat von einer anderen Quelle"

<dfn>Definition</dfn> : zeichnet einen Text aus mit der Bedeutung "dies ist eine Definition"

<acronym>Abkürzung</acronym> : zeichnet einen Text aus mit der Bedeutung "dies ist eine Abkürzung" (z.B. "z.B.")

<abbr>abgekürzte Schreibweise</abbr> : zeichnet einen Text aus mit der Bedeutung "dies ist eine abgekürzte Schreibweise" (z.B. "WWW")

Physische Textauszeichnung (inline)

In HTML gibt es physische und logische Elemente zur Auszeichnung von Text. Physische Textauszeichnungen haben Bedeutungen wie "fett" oder "kursiv", stellen also direkte Angaben zur gewünschten Schriftformatierung dar. Bei physischen Elementen sollte der Web-Browser eine Möglichkeit finden, den so ausgezeichneten Text entsprechend darzustellen. Ebenso wie die logischen Elemente zur Auszeichnung von Text Inline-Elemente sind, gehören die hier beschriebenen Elemente zu den Inline-Elementen.

fett
kursiv
<tt>dicktengleich (Tele-Typer)</tt>
<u>unterstrichen</u>
<strike>durchgestrichen</strike>
<s>durchgestrichen</s>
<big>größer als normal</big>
<small>kleiner als normal</small>
^{hochgestellt}
_{tiefgestellt}

Allgemeines Block-Element

verschiedene HTML-Elemente

z.B. für CSS-Formatierungen.

Sie können mehrere Elemente wie Text, Grafiken, Tabellen usw., in einen gemeinsamen Bereich einschließen. Dieses allgemeine Element bewirkt nichts weiter als dass es in einer neuen Zeile des Fließtextes beginnt. Ansonsten hat es keine Eigenschaften. Es ist dazu gedacht, um mit Hilfe von CSS formatiert zu werden.

Allgemeines Block-Element mit Ausrichtung

verschiedene HTML-Elemente

Siehe „Überschriften ausrichten“

Allgemeines Inline-Element

Text

z.B. für für CSS-Formatierungen.

Analog zum div-Element, das andere Block-Elemente enthalten kann, gibt es ein Element, das Text und andere Inline-Elemente enthalten kann, selbst aber keinerlei Eigenschaften hat und nichts bewirkt. Es ist dazu gedacht, um mit Hilfe von CSS formatiert zu werden. In der "Strict"-Variante dürfen Inline-Elemente nur innerhalb von Block-Elementen notiert werden.

Trennlinien

Text davor
<hr />
Text danach

Trennlinien dienen der optischen Abgrenzung von nicht unmittelbar zusammengehörigen Textabschnitten oder allgemein zur Auflockerung. Eine Trennlinie erzeugt einen eigenen Absatz.

Trennlinien-Eigenschaften

<hr width="Breite" size="Höhe" align="Ausrichtung" />

Für Breite eine Zahl wie z.B. 300 für Anzahl Pixel oder einen Prozentwert wie z.B. 50% angeben.
Für Höhe eine Zahl wie z.B. 5 für Anzahl Pixel angeben.
Für Ausrichtung einen der folgenden Werte notieren:
left = linksbündig | center = zentriert | right = rechtsbündig.

Diese Attribute sind allerdings allesamt als veraltet eingestuft und sollen künftig aus dem HTML-Standard entfallen. Empfohlen wird die Gestaltung von Trennlinien mit CSS.

Schriftformatierung

Text
Text
Text

<basefont size="Größe">
nachfolgender Inhalt
<basefont color="#XXXXXX">
nachfolgender Inhalt
<basefont face="Schriftart">
nachfolgender Inhalt

font für Inline-Auszeichnung verwenden,
basefont als Auszeichnung für alle nachfolgenden Inhalte.

size = Schriftgröße: Für Größe einen Wert von 1 (sehr klein) bis 7 (sehr groß) notieren. 3 ist normal. Oder einen relativen Wert wie z.B. +1, +2 usw. für größer, oder -1, -2 usw. für kleiner.

color = Schriftfarbe: Für #XXXXXX eine hexadezimal notierte RGB-Farbe angeben oder einen erlaubten Farbnamen.

face = Schriftart : Für Schriftart den Namen einer oder mehrerer (alternativer) Schriftarten angeben, mehrere Schriftarten durch Kommata trennen.

Die hier beschriebenen HTML-Elemente sind als veraltet eingestuft, das heißt, sie sollen in Zukunft aus dem HTML-Standard entfernt werden. Da es sich bei diesen Elementen um die schlimmste "Verunreinigung" von HTML als reiner Strukturs prache handelt, die bisher passiert ist, gibt es richtige Kampfschriften und Hetzkampagnen gegen die weitere Verwendung dieser Elemente im Web. Diese Elemente stellten eine Notlösung in jener Zeit dar, als HTML noch keine Formatiersprache wie CSS zur Seite stand. Im Hinblick auf die Rückwärtskompatibilität zu Netscape 3.x kann der Gebrauch der hier beschriebenen Elemente noch nicht grundsätzlich verdammt werden.

Tabellen

Tabellenaufbau

```
<table border="Dicke">
<tr>
<td>Datenzelle</td>
<td>Datenzelle</td>
</tr>
</table>
```

<table> leitet eine Tabelle ein (table = Tabelle). Wenn die Tabelle sichtbare Gitternetzlinien enthalten soll, müssen Sie im einleitenden <table>-Tag das Attribut border= notieren und ihm einen Wert größer 0 zuweisen. Der angegebene Wert ist dann die Breite des Rahmens in Pixeln. Um eine blinde Tabelle ohne sichtbaren Rahmen und Gitternetzlinien zu erzeugen, lassen Sie die Angabe zu border= entweder weg, oder - was sauberer ist - Sie notieren border="0".
<tr> leitet eine neue Tabellenzeile ein (tr = table row = Tabellenzeile). Im Anschluss daran werden die Zellen (Spalten) der betreffenden Reihe definiert. Am Ende einer Tabellenzeile wird ein abschließendes Tag </tr> notiert.
Eine Tabelle kann Kopfzellen und gewöhnliche Datenzellen enthalten. Text in Kopfzellen wird hervorgehoben (meist fett und zentriert ausgerichtet). <th> leitet eine Kopfzelle ein, <td> eine normale Datenzelle (th = table header = Tabellenkopf, td = table data = Tabellendaten). Der Inhalt einer Zelle wird jeweils hinter dem Tag notiert. Obwohl die zugehörigen End-Tags </th> bzw. </td> offiziell optional sind, ist dringend zu empfehlen, sie immer und in jedem Fall zu notieren.

In einer Tabellenzelle können beliebige Elemente stehen, d.h. außer normalem Text z.B. auch andere Block- und Inline-Elemente. Sogar eine weitere Tabelle können Sie innerhalb einer Zelle definieren.

Beachten Sie:

Die Anzahl der Zellen sollte bei jeder Zeile gleich sein, so dass die Tabelle durchweg die gleiche Anzahl Spalten pro Zeile hat. In der ersten Zeile, die Sie definieren, legen Sie deshalb durch die Anzahl der dort definierten Zellen die Anzahl der Spalten Ihrer Tabelle fest.

Tabellenzellen dürfen auch leer sein. Wenn Sie in einer Zeile für eine Spalte keine Daten eingeben wollen, notieren Sie ein einfaches <td></td>. Beachten Sie dabei jedoch, dass viele Web-Browser die Zelle in diesem Fall als "nicht vorhanden" darstellt. Probieren Sie deshalb auch mal die Notation <td> </td> für leere Tabellenzellen aus.

Tabelle mit sichtbaren Gitternetzlinien durch Angabe von border= im einleitenden Tabellen-Tag, durch Weglassen wird die Tabelle eine blinde Tabelle. Für Dicke eine Zahl wie z.B. 3 für Anzahl Pixel der Dicke des Außenrahmens angeben.

Zellenabstand und Zelleninnenabstand

```
<table cellspacing="Abstand" cellpadding="Innenabstand">
<!-- Tabelleninhalt -->
</table>
```

cellspacing = Abstand der Zellen untereinander,
cellpadding = Innenabstand von Zellenrand zu Zelleninhalt.
Für Abstand und Innenabstand Zahlen wie z.B. 5 und 10 für die Anzahl der gewünschten Pixel angeben.

Regeln für Außenrahmen

```
<table border="Dicke" frame="Typ">
<!-- Tabelleninhalt -->
</table>
```

Wenn Sie mit border= einen sichtbaren Tabellenrahmen erzeugen, erhalten automatisch alle Seiten einen Rahmen. Sie können aber auch genau bestimmen, welche Seiten eines Außenrahmens angezeigt werden und welche nicht.

Rahmendicke angeben, damit ein Außenrahmen sichtbar ist. Für Dicke eine Zahl wie z.B. 3 für Anzahl Pixel der Dicke des Außenrahmens angeben. Für Typ einen der folgenden Werte notieren:

box = Rahmen rundum,
above = Rahmen nur oben,
below = Rahmen nur unten,
hsides = Rahmen nur oben und unten.
vsides = Rahmen nur links und rechts.
lhs = Rahmen nur links.
rhs = Rahmen nur rechts.

Regeln für Gitternetzlinien

```
<table border="Dicke" rules="Typ">
<!-- Tabelleninhalt -->
</table>
```

Sie können Regeln aufstellen, welche Gitternetzlinien einer Tabelle angezeigt werden sollen und welche nicht. Rahmendicke angeben, damit Außenrahmen und Gitternetz sichtbar sind.
Für Dicke eine Zahl wie z.B. 3 für Anzahl Pixel der Dicke des Außenrahmens angeben.
Für Typ einen der folgenden Werte notieren:

none = keine Gitternetzlinien,
rows = Linien nur zwischen Tabellenzeilen,
cols = Linien nur zwischen Tabellenspalten,
groups = Linien nur zwischen Kopf, Körper und Fuß,
all = komplettes Gitternetz (Normaleinstellung).

Breiten- und Höhenangaben

```
<table width="Breite">
<tr>
<th width="Breite" height="Höhe">Kopfzelle</th>
<th>Kopfzelle</th>
</tr>
<tr>
<td>Datenzelle</td>
<td width="Breite" height="Höhe">Datenzelle</td>
</tr>
</table>
```

width = Breite, height = Höhe.

Breite der gesamten Tabelle im einleitenden <table>-Tag angeben, Breite einer Spalte in einem der einleitenden <th>- oder <td>-Tags der Spalte, und Höhe einer Tabellenzeile in einem der einleitenden <th>- oder <td>-Tags der Zeile.

Für Breite eine Zahl wie z.B. 100 für Pixel angeben, oder einen Prozentwert wie z.B. 20%.
Für Höhe eine Zahl wie z.B. 50 für Pixel angeben. Das Attribut height ist jedoch nicht XHTML-konform.

Zelleninhalt horizontal ausrichten

<th align="Ausrichtung">Kopfzelle</th>
<td align="Ausrichtung">Datenzelle</td>

Siehe „Überschriften ausrichten“

Zelleninhalt vertikal ausrichten

<th valign="Ausrichtung">Kopfzelle</th>
<td valign="Ausrichtung">Datenzelle</td>

Für Ausrichtung einen der folgenden Werte notieren:
top = obenbündig | middle = mittig | bottom = untenbündig | baseline = an gemeinsamer Basislinie, so dass erste Textzeile immer auf gleicher Höhe beginnt.

Hintergrundfarbe

<table bgcolor="#XXXXXX">
<tr bgcolor="#XXXXXX">
<th bgcolor="#XXXXXX">Kopfzelle</th>
<td bgcolor="#XXXXXX">Datenzelle</td>
</tr>
</table>

Für #XXXXXX eine hexadezimal notierte RGB-Farbe angeben oder einen erlaubten Farbnamen.

Hintergrundbild (Wallpaper)

<table background="URI">
<tr background="URI">
<th background="URI">Kopfzelle</th>
<td background="URI">Datenzelle</td>
</tr>
</table>

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Grafikdatei angeben.

Zellen verbinden

<th rowspan="Zeilen">Kopfzelle</th>
<td rowspan="Zeilen">Datenzelle</td>
<th colspan="Spalten">Kopfzelle</th>
<td colspan="Spalten">Datenzelle</td>
<th rowspan="Zeilen" colspan="Spalten">Kopfzelle</th>
<td rowspan="Zeilen" colspan="Spalten">Datenzelle</td>

Für Zeilen eine Zahl wie z.B. 3 angeben, um festzulegen, über wie viele Zeilen sich die Zelle erstrecken soll.
Für Spalten eine Zahl wie z.B. 3 angeben, um festzulegen, über wie viele Spalten sich die Zelle erstrecken soll.

Tabellen-Beschriftung

```
<table>
<caption align="Ausrichtung">Text</caption>
<tr>
<td>Zelle</td>
<td>Zelle</td>
</tr>
</table>
```

<caption>...</caption> definiert eine Tabellenüberschrift. Das Element muss unmittelbar nach dem einleitenden <table>-Tag notiert werden. Es bewirkt normalen Fließtext. Um den enthaltenen Text auffällig zu gestalten, darf das caption-Element CSS-Anweisungen enthalten.

Wenn Sie nichts anderes angeben, bewirkt das caption-Element eine Tabellenüberschrift. Um eine Tabellenunterschrift zu erzwingen, können Sie <caption align="bottom"> notieren. Ferner sind die Angaben align="left" für seitliche Überschrift links und align="right" für seitliche Überschrift rechts erlaubt. Während align="bottom" von den Browsern interpretiert wird, werden align="left" und align="right" meistens nicht oder nicht korrekt unterstützt.

Das Attribut align= ist darüber hinaus als veraltet gekennzeichnet, soll also künftig aus dem HTML-Standard entfallen. Alternativangaben in CSS werden von den weit verbreiteten Browsern aber im Zusammenhang mit dem caption-Element bislang nicht unterstützt.

Tabellen ausrichten

```
<table align="Ausrichtung">
<!-- Tabelleninhalt -->
</table>
```

Für Ausrichtung einen der folgenden Werte notieren:
left = Tabelle links ausrichten (nachfolgende Inhalte fließen rechts vorbei),
right = Tabelle rechts ausrichten (nachfolgende Inhalte fließen links vorbei),
center = Tabelle zentriert ausrichten.

Zusammenfassung des Tabelleninhalts

```
<table summary="Text">
<!-- Tabelleninhalt -->
</table>
```

Sie können einen zusammenfassenden Kurztext für die Tabelle definieren. Ein Sprachausgabesystem könnte diesen Text als Einleitung zur Tabelle ausgeben.

Verweise (Links)

Verweis definieren

```
<a href="URI">Verweistext</a>
```

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der Datei oder Quelle angeben, das verlinkt werden soll.
URI kann sein:
Anker in der gleichen Datei (z.B. #Ankername)
Andere Datei (z.B. datei.htm)

Andere Datei in anderem Verzeichnis (z.B. ../verzeichnis/datei.htm)
Anker in der anderer Datei (z.B. datei.htm#Ankername)
WWW-Adresse (http://...)
Gopher-Adresse (gopher://...)
FTP-Adresse (ftp://...)
Telnet-Adresse (telnet://...)
Newsgroup-Adresse (news:...)
Absolute lokale Adresse (file://...)

Für Verweise in HTML gibt es das a-Element (a = anchor = Anker). Damit jedoch ein Verweis aus diesem Element wird, ist das Attribut <a href= erforderlich (href = hyper reference = Hyper(text)-Referenz). Als Wert an das href-Attribut weisen Sie das gewünschte Verweisziel zu. Als Inhalt des a-Elements, also zwischen <a> und , notieren Sie den Text, der dem Anwender als Verweis angeboten wird (bei den meisten Web-Browsern andersfarbig, meist unterstrichen).

Als Inhalt des a-Elements, also bei Verweisen der Verweistext, ist nicht nur reiner Text erlaubt. Sie können im Verweistext auch andere notieren. Unter anderem können Sie anstelle von Text auch eine Grafik referenzieren und auf diese Weise fungieren lassen, was in der Praxis des Web-Designs recht häufig vorkommt.

Anker definieren

[Inhalt]

Für Ankername einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Sie können innerhalb einer HTML-Datei Anker definieren. Dann können Sie Verweise zu solchen Ankern setzen, um einen Sprung genau an die Ankerstelle innerhalb der Datei zu veranlassen. Der Verweis kann in der gleichen Datei stehen. Dann wird einfach ein Sprung innerhalb der angezeigten Seite ausgeführt. Der Verweis kann aber auch in einer anderen Datei stehen. Dann wird die Zieldatei geladen, und der Browser springt, sobald er die Stelle mit dem Anker geladen hat, an die entsprechende Stelle innerhalb der Datei.

Zielfenster für Verweise

Verweistext

Für Zielfenster den Namen eines definierten Framefensters angeben oder einen der folgenden reservierten Namen:
_blank = Verweis in neuem Fenster öffnen,
_self = Verweis im gleichen Fenster öffnen,
_parent = aktuelles Frameset beim Ausführen des Verweises sprengen,
_top = alle Framesets beim Ausführen des Verweises sprengen.

E-Mail-Verweise

Verweistext
Verweistext
Verweistext
Verweistext
Verweistext
Verweistext

Auf Wunsch vorbelegtes Thema (subject=), sichtbaren Kopienempfänger (cc=), unsichtbaren Kopienempfänger (bcc=) oder vorbelegten Text (body=) angeben. Mehrere solcher Zusatzangaben kombinieren wie im Schema mit ?Zusatz1&Zusatz2.

In den Angaben Zeichen mit Zeichenwerten größer 127 hexadezimal maskieren, z.B. %FC für 252 ("ü"). Ebenso einige Standardzeichen, die in URIs Bedeutung haben können, maskieren.

Grafiken

Um Grafiken in Ihre HTML-Dateien einzubinden, referenzieren Sie die Grafikdateien an gewünschten Stellen im HTML-Quelltext. Geeignete Dateiformate für Web-gerechte Grafiken sind vor allem GIF und JPEG, allmählich aber auch PNG.
Wenn Sie HTML-Dateien fürs Web erstellen, sollten Sie darauf achten, dass die darin referenzierten Grafiken nicht zu groß sind, denn aufwendige Grafiken verursachen lange Ladezeiten und Missmut beim Anwender. Reduzieren Sie in Ihren Grafiken gegebenenfalls die Anzahl der Farben, verringern Sie die Bildgröße und stopfen Sie nicht zu viele Grafik-Referenzen in eine einzige HTML-Datei. In jedem Fall sollten Sie mit angeben.
Grafiken sind aus HTML-Sicht Inline-Elemente. In der "Strict"-Variante von HTML müssen solche Elemente innerhalb von Block-Elementen vorkommen, etwa in einem Textabsatz oder einem allgemeinen Bereich oder auch einer Tabellenzelle.

Grafiken einbinden

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Grafikdatei angeben.
Für Alternativtext einen Kurztext notieren für den Fall, dass die Grafik nicht angezeigt werden kann.

Das alt-Attribut ist nach HTML-Standard Pflicht. Wenn Sie aus irgendwelchen Gründen keinen Alternativtext notieren wollen, z.B. weil es sich bei der Grafik um einen so genannten "blinden Pixel" handelt, dann notieren Sie alt="" - also eine leere Zeichenkette als Zuweisung.

Achten Sie bei der Wertzuweisung an das src-Attribut unbedingt auf Groß-/Kleinschreibung von Dateinamen und Verzeichnisnamen. Die meisten Server-Rechner laufen mit Betriebssystemen, bei denen streng zwischen Groß- und Kleinschreibung unterschieden wird. Am einfachsten und sichersten ist es, wenn Sie alle Datei- und Verzeichnisnamen kleinschreiben - sowohl beim Vergeben der Namen als auch beim Referenzieren in der HTML-Datei.

Breite und Höhe der Grafik

Für Breite und Höhe eine Zahl wie z.B. 100 für Pixel angeben,
oder einen Prozentwert wie z.B. 60% für Größe in Bezug auf Umgebung.

Wenn Sie Grafiken in HTML-Dateien einbinden, die Sie im Web anbieten wollen, sollten Sie stets die Breite und Höhe der Grafik mit angeben. Dadurch entnimmt der Web-Browser bereits der HTML-Datei, wie groß die Grafik ist, und muss nicht warten, bis er die entsprechende Header-Information der Grafikdatei ausgelesen hat. So kann er die gesamte Web-Seite bereits am Bildschirm aufbauen und bei noch nicht eingelesenen Grafiken erst mal eine entsprechend große Freifläche anzeigen. Wenn Sie Breite und Höhe nicht angeben, wartet der Browser dagegen mit der Anzeige der Web-Seite, bis er alle nötigen Größenangaben aus eingebundenen Grafikdateien eingelesen hat, oder er muss den Bildschirmaufbau korrigieren, was nicht sehr schön aussieht.

Rahmen um Grafik

Für Dicke eine Zahl wie z.B. 3 für die Rahmendicke in Pixeln notieren.

Das Attribut border="" ist als veraltet gekennzeichnet und soll künftig aus dem HTML-Standard entfallen. Umrahmung ist auch mit Hilfe von CSS Stylesheets möglich, und dort mit viel mehr Möglichkeiten.

Namen für Grafik

Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.). Das Vergeben eines Namens ist in Verbindung mit JavaScript sinnvoll. Der Name einer Grafik kann zum Beispiel beim JavaScript-Objekt von Bedeutung sein.

Grafiken im Text ausrichten

Für Ausrichtung einen der folgenden Werte notieren:
top = obenbündig mit Text davor oder danach,
middle = mittig zu Text davor oder danach,
bottom = untenbündig mit Text davor oder danach.

Da das img-Element ein Inline-Element ist, können Grafiken mitten in einem Text platziert werden. Wenn nun aber die Grafik höher ist als die Zeilenhöhe, dann muss der Text der gleichen Zeile in irgendeiner Weise zur Grafik ausgerichtet werden. Wenn Sie nichts anderes angeben, wird der Text untenbündig zur Grafik ausgerichtet. Sie können jedoch mit Hilfe eines Attributs selbst bestimmen, wie der Text zur Grafik ausgerichtet werden soll. Dieses Attribut ist allerdings als veraltet gekennzeichnet und soll künftig aus dem HTML-Standard entfallen. Die gleiche Wirkung lässt sich nämlich auch mit CSS erzielen.

Text um Grafik fließen lassen

<img src="URI" alt="Alternativtext" align="Ausrichtung" hspace="LinksRechts"
vspace="ObenUnten" />

Für Ausrichtung einen der folgenden Werte notieren:
left = Grafik links ausrichten (nachfolgende Inhalte fließen rechts vorbei),
right = Grafik rechts ausrichten (nachfolgende Inhalte fließen links vorbei),
Für LinksRechts eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Grafik zur Umgebung links und rechts zu bestimmen.
Für ObenUnten eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Grafik zur Umgebung oben und unten zu bestimmen.

Grafiken, die Sie mit dem <img>-Tag referenzieren, können Sie links oder rechts ausrichten. Der umgebende Text fließt dabei um die Grafik. Mit zwei weiteren Attributen können Sie Abstand zum umfließenden Text erzeugen, damit der Text nicht direkt an der Grafik "klebt". Alle dazu notwendigen Attribute sind allerdings als veraltet gekennzeichnet und sollen künftig aus dem HTML-Standard entfallen. Die gleiche Wirkung lässt sich nämlich auch mit CSS erzielen.

Verweissensitive Grafik (Image-Map)

<map name="Mapname">
<area shape="Typ" coords="Koordinaten" href="URI" />
</map>

Bei Mapname beim usemap-Attribut des <img>-Tags und beim name-Attribut des <map>-Tags den gleichen Namen vergeben. Beim usemap-Attribut das Gatterzeichen # voranstellen.
Bei Typ einen der folgenden Werte notieren:
rect = viereckiger Bereich innerhalb der Grafik,

circle = Kreisbereich,
poly = vieleckiger Bereich (Polygon).

Bei Koordinaten bei rect so etwas angeben wie 10,20,90,100 (10 ist Anfang Pixel von links, 20 Anfang oben, 90 Ende links, 100 Ende unten),
bei Koordinaten bei circle so etwas angeben wie 100,150,50 (100 ist Kreismittelpunkt in Pixeln von links, 150 Kreismittelpunkt von oben, 50 der Kreisradius),
und bei Koordinaten bei poly so etwas angeben wie 10,20,40,30,300,200 (10 ist erste Ecke links in Pixeln, 20 erste Ecke oben, 40 zweite Ecke links, 30 zweite Ecke oben, 300 dritte Ecke links, 200 dritte Ecke oben). Usw. bebliebig viele Ecken.
Bei URI beim href-Atribut das Verweisziel notieren, das mit dem verweis-sensitiven Bereich verknüpft sein soll.

Um die gewünschten Pixelkoordinaten für verweis-sensitive Flächen einer Grafik zu erhalten, können Sie beispielsweise ein Grafikprogramm benutzen, bei dem Sie mit der Maus in der angezeigten Grafik herumfahren können und dabei die genauen Pixelkoordinaten des Mauszeigers angezeigt bekommen.

Formulare

HTML stellt die Möglichkeit zur Verfügung, Formulare zu erstellen. In Formularen kann der Anwender Eingabefelder ausfüllen, in mehrzeiligen Textfeldern Text eingeben, aus Listen Einträge auswählen usw. Wenn das Formular fertig ausgefüllt ist, kann der Anwender auf einen Button klicken, um das Formular abzusenden.

Dazu geben Sie beim Erstellen eines Formulars an, was mit den Daten des ausgefüllten Formulars passieren soll. Sie können sich die ausgefüllten Daten beispielsweise per E-Mail zuschicken lassen oder von einem CGI-Programm auf dem Server-Rechner weiterverarbeiten lassen.

Formulare können sehr unterschiedliche Aufgaben haben. So werden sie zum Beispiel eingesetzt:

- um bestimmte, gleichartig strukturierte Auskünfte von Anwendern einzuholen,
- um Anwendern das Suchen in Datenbeständen zu ermöglichen,
- um Anwendern die Möglichkeit zu geben, selbst Daten für einen Datenbestand beizusteuern,
- um dem Anwender die Möglichkeit individueller Interaktion zu bieten, etwa um aus einer Produktpalette etwas Bestimmtes zu bestellen.
- Ein Software-Hersteller könnte z.B. ein Formular zur Verfügung stellen, in dem der Anwender angeben kann, welche Produkte der Firma er besitzt, wie er Kenntnis von den Produkten erhalten hat, welchen Beruf er ausübt, auf welchem Rechnertyp die Software bei ihm läuft usw. Auf diese Weise könnte er gleichartig strukturiertes und daher gut vergleichbares Feedback von Anwendern einholen.

Viele Suchdienste im Internet bieten dem aufrufenden Web-Browser Eingabe-Formulare an, in denen der Anwender seinen Suchwunsch spezifizieren kann. Ohne solche Formulare wäre das Durchsuchen gar nicht möglich. Die meisten Suchdienste bieten darüber hinaus auch die Möglichkeit an, eigene Internet-Adressen in die Datenbank einzuspeisen. Das Einholen der dazu nötigen Information geschieht ebenfalls mit Hilfe von Formularen.

Immer zahlreicher werden auch die Online-Shops im Internet. Egal ob Tickets, Pizza oder Unterwäsche - um solche Bestell-Services zu realisieren, sind Formulare erforderlich, in denen der Anwender seine Bestellwünsche genau angeben kann.

Formular definieren

```
<form action="URI" method="Methode" enctype="Mime-Type">
<!-- Formularelemente und andere Elemente innerhalb des Formulars -->
</form>
```

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der Datei oder Quelle angeben, das die Daten verarbeiten soll.
URI kann z.B. ein CGI-Script, eine HTML-Datei mit PHP-Code oder eine E-Mail-Adresse (mailto:jemand@irgendwo.xy) sein.
Für Methode entweder get oder post notieren (bei mailto-Formularen immer post)
Für Mime-Type bei mailto-Formularen text/plain angeben. Bei E-Mail-Formularen besteht keine Garantie auf Erfolg. Es hängt vom Browser und anderen Einstellungen auf dem Rechner des Anwenders ab, ob der Formularversand klappt. E-Mail-Formulare gelten deshalb mittlerweile als unsauber, zumal es Alternativen gibt.

Zielfenster für Server-Antwort

```
<form action="URI" method="Methode" target="Zielfenster">
<!-- Formularelemente und andere Elemente innerhalb des Formulars -->
</form>
```

Für Zielfenster den Namen eines definierten Framefensters angeben oder einen der folgenden reservierten Namen:
_self = Antwort im gleichen Fenster ausgeben wie das Formular,
_parent = aktuelles Frameset für Antwort sprengen,
_top = alle Framesets für Antwort sprengen.

Das target-Attribut ist zwar nicht als veraltet gekennzeichnet, doch um es einzusetzen, müssen Sie die HTML-Variante "Transitional" verwenden. Der Grund ist, dass dieses Attribut vorwiegend für Verweise bei Verwendung von Frames gedacht ist und Frames eine eigene HTML-Variante haben, die von der Einstufung her der Variante "Transitional" entspricht (auf gut Deutsch: nicht der "reinen Lehre" entspricht).

Einzeilige Eingabefelder

```
<input type="text" size="Länge" maxlength="MaxLänge" name="Name" />
```

Für Länge die Anzeigebreite des Feldes in Zeichen notieren (optional).
Für MaxLänge die maximal eingebbare Anzahl Zeichen notieren (optional).
Für Name einen Namen notieren (optional). Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Jedes Eingabefeld sollte einen internen Bezeichnernamen erhalten, und zwar mit der Angabe name=. Der vergebene Name wird bei CGI-Scripts benötigt, um auf die Daten des Eingabefeldes zugreifen zu können. Bei E-Mail-Formularen taucht der Name als Feldname auf. Und auch bei dem Formularfeldzugriff mit JavaScript ist der Name von Bedeutung. Groß- und Kleinschreibung werden bei den meisten Programmiersprachen ebenfalls unterschieden.

Einzeilige Eingabefelder Textvorbelegung

```
<input type="text" value="Wert" />
```

Für Wert den Text der Vorbelegung notieren.

Eingabefelder für Passwort

```
<input type="password" size="Länge" maxlength="MaxLänge" name="Name" />
```

Für die Eingabe von Geheimnummern, Passwörtern usw. gibt es einen speziellen Typ von Eingabefeld. Die eingegebenen Zeichen werden dabei durch Platzhalter (meistens Sternchen) dargestellt, so dass Personen im Raum des Anwenders nicht zufällig das eingegebene Passwort mitlesen können.

Mehrzeilige Eingabefelder

<textarea cols="Spalten" rows="Reihen" name="Elementname">
Optionale Textvorbelegung
</textarea>

Für Spalten die Anzeigebreite des in Anzahl Zeichen pro Zeile notieren.
Für Reihen die Anzeigehöhe in Zeilen notieren.
Für Name einen Namen notieren (optional). Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Die Attribute rows= und cols= bestimmen lediglich die Anzeigegröße des Eingabebereichs, nicht die Länge des erlaubten Textes. Die ist theoretisch unbegrenzt. Web-Browser statten die mehrzeiligen Eingabefelder bei der Anzeige üblicherweise mit Scrollbalken aus, so dass der Anwender bei längeren Eingaben scrollen kann.

Mehrzeilige Eingabefelder Umbruchkontrolle

<textarea cols="Spalten" rows="Reihen" wrap="Umbruch">
</textarea>

Kein HTML-Standard!
Für Umbruch einen der folgenden Werte notieren:
soft = automatischer Zeilenumbruch nur bei Eingabe,
hard = automatischer Zeilenumbruch bei Eingabe wird zu Zeilenumbrüchen bei Formularversand,
virtual = wie soft.
physical = wie hard.
off = kein automatischer Zeilenumbruch bei Eingabe.

Eingabefelder nur lesen

<input type="text" readonly="readonly" value="Text nur lesbar" />

<textarea cols="Spalten" rows="Reihen" readonly="readonly">
Text nur lesbar
</textarea>

Bei älteren Browsern funktioniert diese Angabe nicht, und der Feldeintrag ist für den Anwender überschreibbar!

Auswahlliste

<select size="Höhe" name="Name">
<option>Eintrag</option>
<option>anderer Eintrag</option>
</select>

Für Höhe eine Zahl wie 1 oder 10 notieren, um die Anzeigehöhe der Liste (Anzahl gleichzeitig angezeigter Einträge) zu bestimmen.
Für Name einen Namen notieren (optional). Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Mit dem Attribut size= bestimmen Sie die Anzeigegröße der Liste, d.h. wie viele Einträge angezeigt werden sollen. Wenn die Liste mehr Einträge enthält als angezeigt werden, kann der Anwender in der Liste scrollen. Wenn Sie size="1" angeben, definieren Sie eine so genannte "Dropdown-Liste".

Auswahlliste Mehrfachauswahl

<select multiple="multiple" size="Höhe">
<option>Eintrag</option>
<option>anderer Eintrag</option>
</select>

Eine Mehrfachauswahl ist für Anwender nicht unmittelbar erkennbar. Deshalb sollten Sie darauf hinweisen, wenn mehrere Einträge auswählbar sind. Auch ist nicht allen Anwendern klar, wie sie mehrere Einträge selektieren können. Auf modernen PC-Tastaturen geschieht das normalerweise durch Halten der [Strg]-Taste bei gleichzeitigem Anklicken der gewünschten Listeneinträge.

Auswahlliste mit Vorauswahl

<select multiple="multiple" size="Höhe">
<option>Eintrag</option>
<option selected="selected">anderer Eintrag</option>
</select>
Wenn Sie nichts anderes angeben, ist zunächst kein Eintrag einer Auswahlliste vorselektiert. Sie können einen Eintrag vorselektieren. Um einen Eintrag der Auswahlliste vorzuselektieren, geben Sie im einleitenden <option>-Tag des betreffenden Eintrags das Attribut selected mit gleichem Wert an.

Absendewert von Listeneinträgen

<select size="Höhe">
<option value="Wert">Eintrag</option>
<option value="Wert">anderer Eintrag</option>
</select>

Für Wert bei jedem Eintrag einen passenden Absendewert notieren.

Normalerweise wird beim Absenden des Formulars der Text eines ausgewählten Listeneintrags übertragen, der zwischen <option> und </option> notiert ist. Sie können jedoch bestimmen, dass intern ein anderer Text übertragen wird.

Radiobuttons

<input type="radio" name="Name" value="Wert" /> Text

Für Name einen Namen notieren. Zusammengehörige Radiobuttons erhalten den gleichen Namen. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Wert einen Text notieren, der beim Formularversand diesen Button bezeichnet.
Für Text einen sichtbaren Beschriftungstext notieren.

Radiobuttons sind eine Gruppe von beschrifteten Knöpfen, von denen der Anwender einen auswählen kann. Es kann immer nur einer der Radiobuttons ausgewählt sein. Der Wert des ausgewählten Radiobuttons wird beim Absenden des Formulars mit übertragen.

Checkboxen

<input type="checkbox" name="Name" value="Wert" /> Text

Für Name einen Namen notieren. Zusammengehörige Checkboxen erhalten den gleichen Namen. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Wert einen Text notieren, der beim Formularversand diese Checkbox bezeichnet.
Für Text einen sichtbaren Beschriftungstext notieren.

Checkboxen sind eine Gruppe von ankreuzbaren Quadraten, bei denen der Anwender keine, eins oder mehrere auswählen kann. Die Werte von ausgewählten Checkboxen werden beim Absenden des Formulars mit übertragen.

Einträge vorselektieren

<input type="radio" checked="checked" name="Name" value="Wert" /> Text
<input type="checkbox" checked="checked" name="Name" value="Wert" /> Text

Klickbuttons (1)

<input type="button" name="Name" value="Beschriftung" onClick="Aktion" />

Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Beschriftung den Text notieren, der als Button-Beschriftung angezeigt wird.
Für Aktion z.B. eine JavaScript-Anweisung notieren.

Sie können Schaltflächen definieren, die keine spezielle Bedeutung haben. Sinnvoll sind solche frei definierten Klickbuttons in Verbindung mit Scriptsprachen wie JavaScript. In der Regel werden sie dazu verwendet, Verweise oder andere JavaScript-gesteuerte Anweisungen auszuführen.
Die hier beschriebene, herkömmliche Methode hat den Vorteil, dass sie auch von älteren Browsern (Netscape seit Version 2.x, MS Internet Explorer seit Version 3.x) interpretiert wird.

Klickbuttons (2)

<button type="button" name="Name" value="Alternativbeschriftung" onClick="...">
Beschriftung
</button>

Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Alternativbeschriftung einen Beschriftungstext notieren, falls kein Elementinhalt notiert wird.
Für Beschriftung die Anzeigefläche des Buttons gestalten, z.B. auch mit Grafiken und anderen HTML-Elementen.

Ab HTML 4.0 dürfen anklickbare Buttons endlich so heißen wie sie heißen: nämlich Button. Solche Buttons sind flexibler als herkömmliche Buttons, denn sie dürfen auch einen definierten Inhalt haben, etwa eine Grafik. Der Nachteil ist, dass diese Sorte Buttons nur von neueren Browsern unterstützt wird - vor allem von Netscape 4.x noch nicht.

Formularfeld für Datei-Upload

<input type="file" name="Name" maxlength="Größe" accept="Mime-Type" />

Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Wenn Sie das Attribut maxlength= angeben, sollte der Web-Browser die dahinter notierte Zahl als maximal erlaubte Dateigröße in Bytes interpretieren (maxlength = maximal length = maximale Länge). Im obigen Beispiel wird auf diese Weise die Bytezahl auf 100000 Byte begrenzt. Wenn Sie maxlength= weglassen, kann der Anwender beliebig große Dateien senden. Beachten Sie jedoch, dass diese Angabe mit Vorsicht zu genießen ist. In der HTML-Version 3.2 wurde es so bestimmt, in der Version 4.0 wird diese Funktionalität beim maxlength-Attribut dagegen nicht mehr erwähnt. Verlassen Sie sich also nicht auf diese Angabe. Sicherer ist es, bei der Weiterverarbeitung mit einem CGI-Script im Script die Dateigröße zu

ermitteln und das Script davon abhängig entscheiden zu lassen, ob die Datei akzeptiert oder verworfen wird.

Wenn Sie nur bestimmte Dateitypen zulassen wollen, können Sie mit der Angabe accept= die erlaubten Dateitypen eingrenzen (accept = akzeptieren). Wichtig: im <form>-Tag enctype="multipart/form-data" notieren.

Diese Sorte Formularelement erlaubt dem Anwender, eine Datei von seinem lokalen Rechner zusammen mit dem Formular zu übertragen. Wenn ein CGI-Script die ankommenden Formulardaten auf dem Server-Rechner verarbeitet, ist es dadurch möglich, dem Anwender das Uploaden (Hochladen) von Dateien auf den Server-Rechner zu ermöglichen.

Versteckte Elemente

<input type="hidden" name="Name" value="Wert" />

Für Name einen Namen notieren. Zusammengehörige Checkboxen erhalten den gleichen Namen. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Wert einen Text notieren, der beim Formularversand mit übertragen, aber nicht angezeigt wird.

Sie können Felder in einem Formular definieren, die dem Anwender nicht angezeigt werden. Versteckte Felder können Daten enthalten. Beim Absenden des Formulars werden die Daten versteckter Felder mit übertragen. Auf diese Weise können Sie beispielsweise zusätzliche Informationen an übergeben oder erläuternden Text einfügen, der bei der E-Mail-Übertragung der Formulardaten in der E-Mail mit enthalten ist.

Elemente gruppieren

<fieldset>
<legend>Gruppenüberschrift</legend>
<!-- Formularelemente -->
</fieldset>

Größere Formulare bestehen häufig aus Gruppen von Elementen. Ein typisches Bestellformular besteht beispielsweise aus Elementgruppen wie "Absender", "bestellte Produkte" und "Formular absenden/abbrechen". Solche Elementgruppen können Sie ab HTML 4.0 eigens auszeichnen. Ein Web-Browser kann Elementgruppen durch Linien oder ähnliche Effekte optisch sichtbar machen.

Label für Elemente

<label for="idName">Beschriftung:</label>
<[Formularelement] id="idName">

idName müssen beim for-Attribut des <label>-Tags und beim id-Attribut des zugehörigen Formular-Elements übereinstimmen. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Funktioniert erst ab Netscape 6 und IE4.

Buttons zum Absenden/Abbrechen

<input type="submit" value="Beschriftung" />
<input type="reset" value="Beschriftung" />
Mit <input type="submit"> definieren Sie einen Absende-Button (input = Eingabe, submit = bestätigen). Beim Anklicken dieses Buttons werden die Formulardaten abgeschickt, und es wird die Adresse aufgerufen, die im einleitenden <form>-Tag beim Attribut action= angegeben ist.

Mit <input type="reset"> definieren Sie einen Abbrechen-Button (reset = zurücksetzen). Eingegebene Daten werden verworfen und nicht abgeschickt.
Für Beschriftung den jeweiligen Beschriftungstext des Buttons notieren.

Zwei Standard-Buttons stellt HTML zur Verfügung, um Formulareingaben zu handhaben: einen Button zum "Absenden" und einen zum "Abbrechen". Mit dem Absende-Button kann der Anwender das ausgefüllte Formular losschicken. Mit den Formulardaten geschieht dann das, was im einleitenden <form>-Tag bei action= festgelegt wurde (siehe hierzu). Mit dem Abbrechen-Button kann der Anwender alle Eingaben verwerfen. Das Formular wird nicht abgeschickt, Eingaben werden gelöscht.

Grafischer Absendebutton

<input type="image" src="URI" />

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Grafikdatei angeben.

Mit <input type="image"> definieren Sie einen grafischen Button (input = Eingabe).

Frames

Mit Hilfe von Frames können Sie den Anzeigebereich des Browsers in verschiedene, frei definierbare Segmente aufteilen. Jedes Segment kann eigene Inhalte enthalten. Die einzelnen Anzeigesegmente (also die Frames) können wahlweise einen statischen Inhalt (= "non scrolling regions") oder einen wechselnden Inhalt haben. Verweise in einem Frame können Dateien aufrufen, die dann in einem anderen Frame angezeigt werden.

Framesets und Frames

```
<html><head><title>Titel</title></head>
<frameset cols="Spalten" rows="Reihen">
<frame src="URI" name="Name">
<noframes>
Wird angezeigt, wenn der Browser keine Frames anzeigen kann
</noframes>
</frameset>
</html>
```

Anzeigefenster in Spalten (cols=) oder Reihen (rows=) aufteilen.
Zwei oder mehrere Teile in Pixeln oder Prozent angeben, durch Kommata getrennt, z.B. rows="60%,40%".
Wildcard * erlaubt, z.B. cols="200,*".
Verschachtelung von Framesets möglich.
Für jedes entstehende Segment ein <frame>-Tag notieren.
Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Datei angeben, die im Framefenster angezeigt werden soll.
Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Bereich mit noframes-Element notieren wie gezeigt. Inhalt wird nur angezeigt, wenn Browser keine Frames anzeigen kann. Inhalt kann HTML-Elemente enthalten.

Ein wichtiger Unterschied zu anderen HTML-Dateien ist, dass Dateien mit Frameset-Definitionen kein body-Element besitzen. Anstelle des body-Elements, also nach dem abschließenden </head>-Tag für den Dateikopf, werden die Frames definiert. Mit Framesets bestimmen Sie die Aufteilung der Framefenster, mit Frames die Datenquellen der einzelnen Framefenster, und der Noframes-Bereich ist für Browser gedacht, die keine Frames anzeigen können, oder bei denen die Anzeige von Frames deaktivierbar ist und vom Anwender deaktiviert wurde.
Der Titel (<title>...</title>), den Sie in der Datei mit der Frameset-Definition angeben, wird während der gesamten Dauer des Frame-Sets angezeigt, auch wenn andere Dateien innerhalb des Frame-Sets

aufgerufen werden. Leider gibt es bislang keine Möglichkeit, den Titel zu aktualisieren. Wählen Sie in der Datei, die die Frame-Set-Definitionen enthält, deshalb einen allgemeinen, aussagekräftigen Titel, der für das gesamte Projekt Gültigkeit hat.

Der URI der Datei mit der Frameset-Definition, also beispielsweise *http://www.ihr-guter-name.de/*, bleibt in der gleichen Form in der Adresszeile des Browsers stehen, auch wenn der Anwender durch Navigieren innerhalb des Framesets andere Seiten des Projekts in eines der Framefenster lädt.

Scrollbars in Frames

<frame src="URI" name="Name" scrolling="Option" />

Für Option einen der folgenden Werte angeben:
yes = Scrollbars anzeigen,
no = keine Scrollbars anzeigen (kein Scrollen möglich),
auto = Scrollbars bei Bedarf anzeigen (Normaleinstellung).

Wenn Sie das Scrollen des Fensterinhalts verhindern, können Inhalte, die größer sind als das Fenster, nicht vollständig angezeigt werden. Verwenden Sie scrolling="no" daher nur, wenn Sie sicher sind, dass der Fensterinhalt vollständig in das Frame-Fenster passt. Die Angabe ist zum Beispiel sinnvoll, wenn in einem Frame-Fenster dauerhaft ein Logo angezeigt werden soll und sonst nichts. Bei größeren Inhalten, vor allem in Framefenstern mit relativer Größe, sollten Sie mit dem Ausschalten der Scrollbars vorsichtig sein - bedenken Sie, dass nicht alle Anwender die gleiche Bildschirmauflösung wie Sie haben.

Abstand/Rand zu Fensterinhalt

<frame src="URI" name="Name" marginwidth="LinksRechts" marginheight="ObenUnten" />

Für LinksRechts eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Frame-Rand zum Frame-Inhalt links und rechts zu bestimmen.
Für ObenUnten eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Frame-Rand zu Frame-Inhalt oben und unten zu bestimmen.

Mit den Angaben marginwidth="0" marginheight="0" sollte der Fensterinhalt exakt in der linken oberen Ecke beginnen. Leider fügt Netscape 4.x bei der Anzeige doch immer noch ein Pixel "Seitenrand" ein. Als Alternative dazu bleibt nur, den Inhalt mit Hilfe von CSS Stylesheets mit top:0px und left:0px.

Unveränderbare Fenstergröße

<frame src="URI" name="Name" noresize="noresize" />

Alle benachbarten Frames sind durch noresize mit betroffen.

Rahmendicke / unsichtbare Rahmen

<frame src="URI" name="Name" frameborder="Dicke" />
<frameset [...] border="Dicke" frameborder="Dicke" framespacing="Abstand">

HTML-Standard erlaubt nur das border-Attribut im <frame>-Tag. Browser interpretieren diese Angaben jedoch anders als HTML-Standard, nämlich im <frameset>-Tag. Für unsichtbare Rahmen alle drei Werte für Dicke auf 0 setzen.

Nur mit frameborder="0" framespacing="0" (also mit beiden Angaben) unterdrücken Sie beim MS Internet Explorer die Rahmen konsequent. Wenn Sie nur frameborder="0" angeben, zeigt der Internet Explorer sichtbare Abstände zwischen den Frame-Fenstern an, allerdings nicht die üblichen 3D-Rahmen, sondern flache Zwischenräume. Um die Rahmen beim Internet Explorer und bei Netscape zu unterdrücken, müssen Sie alle drei Angaben frameborder="0" framespacing="0" border="0" notieren.

Wenn Sie die Rahmen zwischen den Frame-Fenstern unterdrücken, hat der Anwender keine Möglichkeit, die Größe der Frame-Fenster zu verändern. Testen Sie Projekte mit rahmenlosen Frames deshalb möglichst unter mehreren Bildschirmauflösungen und mit unterschiedlich großem Anzeigefenster.

Eingebettete Frames

<iframe src="URI" name="Name" width="Breite" height="Höhe">
Inhalt bei Nichtanzeige
</iframe>

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Datei angeben, die im Framefenster angezeigt werden soll.
Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).
Für Breite und Höhe eine Zahl wie z.B. 100 für Pixel angeben,
oder einen Prozentwert wie z.B. 60% für Größe in Bezug auf Umgebung.

Eingebettete Frames werden vom MS Internet-Explorer ab Version 3.x interpretiert, von Netscape bis einschließlich Version 4.x dagegen nicht. Seit HTML 4.0 gehören sie zum HTML-Standard. Eingebettete Frames sind ein eigenständiges Gestaltungsmittel zur Informationsaufbereitung, das anders funktioniert als "normale" Frames. Eingebettete Frames erzeugen keine Aufteilung des Bildschirms, sondern sind ähnlich wie Grafiken Bereiche innerhalb einer HTML-Datei, in denen fremde Quellen, vor allem andere HTML-Dateien angezeigt werden können.

Wenn Sie das iframe-Element in einer HTML-Datei notieren, müssen Sie in dieser Datei die HTML-Variante "Transitional" wählen. Eingebettete Frames gehören also auch nicht zur "reinen Lehre".

Eigenschaften eingebetteter Frames

```
 <iframe src="URI" name="Name"
   width="Breite" height="Höhe" scrolling="Option"
   align="Ausrichtung" hspace="LinksRechts" vspace="ObenUnten"
   marginwidth="RandLinksRechts" marginheight="RandObenUnten">
Inhalt bei Nichtanzeige
</iframe>
```

Für Option einen der folgenden Werte angeben:
yes = Scrollbars anzeigen,
no = keine Scrollbars anzeigen (kein Scrollen möglich),
auto = Scrollbars bei Bedarf anzeigen (Normaleinstellung).
Für Ausrichtung einen der folgenden Werte notieren:
left = Fenster links ausrichten (nachfolgende Inhalte fließen rechts vorbei),
right = Fenster rechts ausrichten (nachfolgende Inhalte fließen links vorbei),
Für LinksRechts eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Fenster zur Umgebung links und rechts zu bestimmen.
Für ObenUnten eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Fenster zur Umgebung oben und unten zu bestimmen.
Für RandLinksRechts eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Fensterrand zum Fensterinhalt links und rechts zu bestimmen.
Für RandObenUnten eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Fensterrand zum Fensterinhalt oben und unten zu bestimmen.

Multimedia-Objekte

Daten als Objekt einbinden

<object data="URI" type="Mime-Type" width="Breite" height="Höhe">
Alternativer Inhalt
</object>

Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Datei angeben, die angezeigt werden soll.
Für Mime-Type so etwas wie image/svg+xml notieren, je nachdem, welcher Dateityp referenziert wird.
Für Breite und Höhe eine Zahl wie z.B. 100 für Pixel angeben,
oder einen Prozentwert wie z.B. 60% für Größe in Bezug auf Umgebung.

Sie können eine beliebige Datendatei als Objekt in eine HTML-Datei einbinden, also z.B. ein Video, eine Konstruktionszeichnung, eine als fertige Datei vorliegende 3D-Welt, eine Musikdatei oder dergleichen. Auch einfache Textdateien und andere HTML-Dateien können Sie auf diese Weise einbinden. Ein Web-Browser kann solche Dateien anzeigen, wenn er entweder selber in der Lage ist, das Dateiformat anzuzeigen, oder wenn der Anwender ein entsprechendes Plugin installiert hat. Wenn das Plugin installiert ist, kann der Web-Browser die Datei so in seinem Anzeigefenster präsentieren, wie sie von dem Ursprungsprogramm erstellt wurde. Bei Abspielvorgängen, etwa von Videos oder Sound, wird ein entsprechender Player angezeigt - je nachdem, wie das Plugin beschaffen ist.
Wenn dem Browser eine Verknüpfung zwischen dem Datentyp und einem Fremdprogramm bekannt ist, kann er das Fremdprogramm mit der betreffenden Datei starten. Ob die Daten dann jedoch innerhalb des Bereichs auf der Web-Seite angezeigt werden, der für das Objekt definiert ist, hängt davon ab, ob der Browser und das andere Programm entsprechende Kommunikationsmöglichkeiten, z.B. vom Betriebssystem bereitgestellte Kommunikationsschnittstellen, nutzen.

Java-Applets als Objekt

<object width="Breite" height="Höhe" classid="java:datei.class" codebase="URI"
codetyte="application/java-vm">
<param name="Name" value="Wert" />
</object>

codebase-Attribut angeben, wenn Applet nicht im aktuellen Verzeichnis liegt. Mit URI die Adresse des Verzeichnisses angeben.
Parameter mit name- und value-Attribut angeben wenn es das Applet erfordert.

Java-Applets sind ausführbare Programme, deren Bildschirmausgaben ein Web-Browser innerhalb seines Anzeigefensters darstellen kann. Applets können z.B. bewegte Animationen (Tricksequenzen) enthalten, Echtzeitabläufe in bewegten Grafiken darstellen (Stichwort: Börsenticker), oder Interaktionen mit dem Anwender austauschen. So werden Java-Applets etwa häufig bei Online-Banking eingesetzt.
Java-Applets müssen in compilierter Form vorliegen, um bei der Referenzierung in einer HTML-Datei ausgeführt werden zu können. Normalerweise haben compilierte Java-Applets die Dateinamenerweiterung *.class*.

ActiveX als Objekt

```
<object width="Breite" height="Höhe"
  classid="CLSID:XXXXXX-XXXX-XXXXXX">
<param name="Name" value="Wert" />
</object>
```

Für XXXXXX-XXXX-XXXXXX die ID des ActiveX-Controls notieren.

ActiveX-Controls können ähnliche Aufgaben wahrnehmen wie Java-Applets. Jedoch sind sie stärker als Java-Applets in der Windows-Welt und der Welt von Microsoft verankert und werden nur vom Microsoft Internet Explorer interpretiert.

Flash-Movies als Objekt

```
<object width="Breite" height="Höhe" classid="CLSID:CLSID:D27CDB6E-AE6D-11cf-96B8-
444553540000" codebase="http://active.macromedia.com/flash5/cabs/swflash.cab#version=5,0,0,0">
<param name="movie" value="datei.swf" />
<param name="Name" value="Wert" />
</object>
```

classid-Attribut erhält immer diesen Wert.
Beim codebase-Attribut Flash-Version angeben.
Beim einem der Parameter name-Attribut movie beim value-Attribut den Dateinamen der swf-Datei angeben.
Parameter mit name- und value-Attribut ansonsten angeben wenn es die Anwendung erfordert.

Namen für Objekt

```
<object data="URI" type="Mime-Type" name="Name"></object>
```

Für Name einen Namen notieren. Keine Leerzeichen und Umlaute, erstes Zeichen ein Buchstabe, sonst auch Ziffern, Unterstrich (_), Bindestrich (-), Doppelpunkt (:) oder Punkt (.).

Java-Applets einbinden

```
<applet code="datei.class" codebase="URI" alt="Text" width="Breite" height="Höhe">
<param name="Name" value="Wert" />
</applet>
```

codebase-Attribut angeben, wenn Applet nicht im aktuellen Verzeichnis liegt. Mit URI die Adresse des Verzeichnisses angeben.
Für Breite und Höhe eine Zahl wie z.B. 100 für Pixel angeben,
oder einen Prozentwert wie z.B. 60% für Größe in Bezug auf Umgebung.
Parameter mit name- und value-Attribut angeben wenn es das Applet erfordert.

Java hat - zumindest im Web - nicht ganz das gebracht, was die Marketingstrategen anfangs prophezeiht hatten. Aber eines kann man sagen: es hat eine Menge Staub aufgewirbelt. Sun, der Entwickler von Java, hatte in den wilden Anfangsjahren von HTML zunächst eine eigene Syntax zum Einbinden von Java-Applets in seinen WWW-Browser "HotJava" integriert. Doch Netscape implementierte eine andere Syntax. Diese Syntax wurde dann auch seinerzeit in den HTML-Standard 3.2 übernommen.

Java-Applets-Eigenschaften

```
 <applet code="datei.class" codebase="URI" alt="Text"
   width="Breite" height="Höhe"
   align="Ausrichtung" hspace="LinksRechts" vspace="ObenUnten">
<param name="Name" value="Wert" />
</applet>
```

Für Ausrichtung einen der folgenden Werte notieren:
top = obenbündig mit Text davor oder danach,
middle = mittig zu Text davor oder danach,
bottom = untenbündig mit Text davor oder danach.
left = Applet links ausrichten (nachfolgende Inhalte fließen rechts vorbei),
right = Applet rechts ausrichten (nachfolgende Inhalte fließen links vorbei),
Für LinksRechts eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Applet zur Umgebung links und rechts zu bestimmen.
Für ObenUnten eine Zahl wie z.B. 10 notieren, um den Pixelabstand von Applet zur Umgebung oben und unten zu bestimmen.

Multimedia (Netscape)

```
<embed src="URI" width="Breite" height="Höhe">
```

Gehört nicht zum HTML-Standard!
Für URI eine Web-Adresse oder ein Ziel mit oder ohne Pfad der gewünschten Datei angeben, die angezeigt werden soll.
Für Breite und Höhe eine Zahl wie z.B. 100 für Pixel angeben,
oder einen Prozentwert wie z.B. 60% für Größe in Bezug auf Umgebung.

Einbindung CSS und JavaScript

Stylesheet-Bereich für CSS

```
<head>
<style type="text/css">
<!--
[ Zentrale Formate ]
-->
</style></head>
```

Mit <style> leiten Sie innerhalb des HTML-Dateikopfs, also zwischen <head> und </head>, einen Bereich zum Definieren von Formaten ein (style = Stil), mit </style> beenden Sie den Abschnitt. Die Formate, die Sie innerhalb des Bereichs nach der Syntax der Stylesheet-Sprache definieren, gelten dann für die gesamte Datei.

Innerhalb des einleitenden <style>-Tags müssen Sie mit dem Attribut type= angeben, welche Stylesheet-Sprache Sie innerhalb des Bereichs zum Definieren der Formate benutzen möchten.

Mit dem Attribut media=, das im Gegensatz zu type= kein Pflichtattribut ist, können Sie den Typ des Ausgabemediums bestimmen. Neben der Wertzuweisung screen (für das Medium "Bildschirm") gibt es weitere mögliche Ausgabemedien. Sie können, indem Sie mehrere style-Bereiche definieren, die jeweils unterschiedliche media-Attribute haben, Stylesheets für verschiedene Ausgabemedien definieren.

Es ist empfehlenswert, den Inhalt von Style-Sheet-Bereichen in einen (<!-- bis -->) zu setzen, so wie im obigen Beispiel. Dadurch verhindern Sie, dass ältere Browser die Inhalte irrtümlich als Text anzeigen.

CSS-Formatierungen im Text

<[Elementname] style="[CSS-Eigenschaften]">...</[Elementname]>

Script-Bereich

```
 <script language="text/javascript">
<!--
/* JavaScript-Code */
//-->
</script>
<noscript>
Inhalt, wenn kein JavaScript möglich ist
</noscript>
```

Mit <script> leiten Sie einen Script-Bereich ein, mit </script> beenden Sie den Bereich. Innerhalb des Bereichs können Sie Anweisungen der verwendeten Script-Sprache notieren. Aus Sicht von HTML wird alles, was innerhalb des Bereichs steht, als "nackter Text" betrachtet. Dabei kann es allerdings zu Konflikten zwischen einer Sprache wie JavaScript und einem streng SGML-konformen Parser kommen. Ein solcher Parser betrachtet den Script-Bereich als beendet, sobald er auf die nächste Zeichenfolge </ stößt. Wenn Sie innerhalb von JavaScript diese Zeichenfolge benötigen, wie im obigen Beispiel in der letzten Anweisung, wo mit Hilfe von document.write eine Überschrift erster Ordnung ins Dokument geschrieben wird. Am Ende des ausgegebenen Textes ist dort <\/h1> notiert statt </h1>. Der Backslash dient in JavaScript dazu, ein nachfolgendes Zeichen zu maskieren, was im Beispiel an der Stelle aus JavaScript-Sicht aber keine weitere Bedeutung hat und nichts weiter bewirkt. Der Backslash ist auch nicht für den JavaScript-Interpreter gedacht, sondern für penible HTML-Parser.

Um zu verhindern, dass ältere Browser den Inhalt des Script-Bereichs irrtümlich als Text ausgeben, empfiehlt es sich, den Inhalt in einen Kommentar-Tag zu setzen, so wie im obigen Beispiel. In der letzten Zeile des Scriptbereichs notieren Sie dann am besten einen einzeiligen Kommentar, eingeleitet durch //, gefolgt vom schließenden HTML-Kommentar -->. Der JavaScript-Kommentar ist erforderlich, um Fehlermeldungen in scriptfähigen Browsern zu unterdrücken. Berücksichtigen Sie, dass vor und nach dem HTML-Kommentar ein Zeilenumbruch zwingend erforderlich ist.

Innerhalb des einleitenden <script>-Tags geben Sie mit dem Pflichtattribut type= an, welche Script-Sprache Sie innerhalb des Bereichs benutzen möchten. Als Wert weisen Sie den der Script-Sprache zu. Die gängigste Angabe ist dabei text/javascript für JavaScript.

Das immer noch sehr vielgenutzte Attribut language=, mit dem viele Web-Designer im einleitenden <script>-Tag die Script-Sprache angeben (z.B. language="JavaScript" oder language="JavaScript1.2"), gehört nicht zum HTML-Standard und erzeugt ungültiges HTML. Die Browser kennen das language-Attribut zwar und werten es aus, aber für modernere Browser reicht auch das Attribut type= zur Erkennung der Sprache aus.