

# HTML - Einführung

Hypertext Markup Language  
die Sprache des World Wide Web

Version 2001.03

Hubert Partl  
partl@mail.boku.ac.at  
Zentraler Informatikdienst  
Universität für Bodenkultur Wien

## Vorwort zur gedruckten Version

Ich habe meine "HTML-Einführung" in der Sprache geschrieben, die dafür die natürlichste ist, nämlich in HTML selbst, und ich habe dabei auch viele der Möglichkeiten verwendet, die HTML und Hypertext-Links bieten und die in der auf Papier ausgedruckten Version zwangsläufig verloren gehen. Die eigentliche Version ist daher nicht dieses gedruckte Buch, sondern die Online-Hypertext-Version, die ich unter dem URL "<http://www.boku.ac.at/html Einf/>" über das Internet verfügbar gemacht habe.

Die gedruckte Version, die Sie in Händen halten, ist nur ein primitives Abbild der Online-Version.

Mit →Pfeilen sind hier die Stellen bezeichnet, wo der Leser der Online-Version per Mausklick zu weiteren Informationen springen kann (→Hypertext-Links). Die Möglichkeit, solche Hypertext-Links sofort auszuprobieren, sowie die Darstellung von Bildern, das Ausfüllen von Formularen, die Suche nach bestimmten Wörtern oder das Absenden von Briefen fehlen in der Papier-Version.

Die einzelnen Kapitel sind hier in einer von mehreren empfohlenen Lese-Reihenfolgen angeordnet. Welche anderen Lese-Möglichkeiten und Navigations-Hilfen und welche Verweise zu weiteren, von anderen Autoren stammenden Informationen die Online-Version bietet, können Sie in →Kapitel 9 sehen.

## Vorwort zur Hypertext-Online-Version

In der vorliegenden "HTML-Einführung" werden die wichtigsten Elemente und Eigenschaften der →Hypertext Markup Language HTML kurz beschrieben. Dies sollte für die meisten Anwendungen ausreichend sein (siehe den →Wegweiser). Für genauere und vollständigere Informationen wird auf die →Referenzen verwiesen.

Die HTML-Elemente werden an Hand von einfachen **Beispielen** erläutert. Wie im Abschnitt →"Inhalt und Form" beschrieben, wird mit →HTML nur die logische Struktur der Informationen festgelegt, und das Layout hängt vom jeweils verwendeten →Web-Browser (→Client) ab. Die Ausgabe, die Sie bei den Beispielen sehen, gilt daher nur für *Ihren* Web-Browser und nur mit *Ihren* persönlichen Einstellungen. Ich empfehle Ihnen, die Probe aufs Exempel zu machen, indem Sie die Einführung und die darin enthaltenen Beispiele mit verschiedenen Web-Browsern auf verschiedenen Bildschirmen oder zumindest mit verschiedenen Fenstergrößen und persönlichen Einstellungen (Optionen, Präferenzen) ansehen.

Hinweise, in welcher Reihenfolge Sie die einzelnen Abschnitte dieser Einführung am besten lesen, finden Sie im →Wegweiser.

Ich lade Sie ein, Anregungen und Verbesserungsvorschläge zu dieser Einführung an meine Mail-Adresse →partl@mail.boku.ac.at oder an meine Post-Adresse an der →BOKU Wien zu senden, und ich bedanke mich bei allen "Internet-Surfern", die mir schon beim Korrekturlesen geholfen haben.

*Hubert Partl  
Wien, im Juli 1995*

## Copyright

Diese Dokumentation, die "HTML-Einführung" von Hubert Partl, ist im World-Wide-Web für den Online-Zugriff veröffentlicht, das **Urheberrecht** und die **Nutzungsrechte** (Copyright) liegen aber trotzdem beim Autor.

Das Lesen über das →World Wide Web und das Abspeichern und Ausdrucken für den eigenen Gebrauch ist für jedermann erlaubt.

Die Veröffentlichung von Zitaten (kurzen Ausschnitten) mit Angabe des Autors und der Quelle ist erlaubt.

Die Erstellung, die Verwendung und die nicht kommerzielle Weitergabe von Kopien der kompletten Dokumentation in elektronischer oder ausgedruckter Form sind erlaubt, wenn der Inhalt (einschließlich der Markup-Befehle, der Autoren-Angabe und dieser Copyright-Information) unverändert bleibt. Die kommerzielle Weitergabe ist nach Rücksprache mit dem Autor ebenfalls erlaubt.

Die Erstellung und Verbreitung von Bearbeitungen (veränderten, erweiterten, gekürzten oder übersetzten Versionen) ist nur nach Rücksprache mit dem Autor erlaubt und nur, wenn in der Autorenliste der neuen Version sowohl der Autor der Bearbeitung als auch der Autor der ursprünglichen Version angegeben werden.

Der Autor kann keine Garantie für die Aktualität und Richtigkeit der Dokumentation übernehmen.

## Zugriff über das Internet

Die Online-Hypertext-Version der "HTML-Einführung" ist unter dem folgenden →URL verfügbar:

→<http://www.boku.ac.at/html Einf/>

Wenn Sie eine **lokale Kopie** der Files anlegen wollen, haben Sie zwei Möglichkeiten:

- Entweder Sie speichern die einzelnen HTML-Files (siehe →komplettes Handbuch lesen) und die darin verwendeten Bilder und Style-Sheets mit dem Save- oder Download-Befehl Ihres →Web-Browsers,
- oder Sie übertragen die folgenden zwei mit Gnuzip komprimierten Unix-Files vom anonymen →FTP-Server der BOKU:  
→<ftp://mail.boku.ac.at/www/hein.tar.gz> (ca. 100 kBytes) und  
→<ftp://mail.boku.ac.at/www/html Einf.ps.gz> (ca. 300 kBytes)  
oder die mit PKZIP komprimierten MS-DOS-Files  
→<ftp://mail.boku.ac.at/www/hein.zip> (ca. 100 kBytes, ohne PostScript)

---

# Inhaltsverzeichnis

---

<b>1</b>	<b>Die Grundlagen</b>	<b>1</b>
1.1	WWW – Was ist das? . . . . .	2
1.2	Inhalt des HTML-Files . . . . .	2
1.2.1	Was darf ich im WWW veröffentlichen? . . . . .	2
1.2.2	Was soll ich im WWW veröffentlichen? . . . . .	2
1.2.3	Frisch geplant ist halb gewonnen! . . . . .	3
1.3	Inhalt und Form . . . . .	4
1.4	Richtige HTML . . . . .	5
1.4.1	Was ist richtig? . . . . .	5
1.4.2	Weltweite Zusammenarbeit oder Firmenabhängigkeit . . . . .	5
1.5	Format der Markup-Befehle (HTML-Tags) . . . . .	7
1.6	Aufbau eines HTML-Files <code>&lt;html&gt; &lt;head&gt; &lt;body&gt;</code> . . . . .	8
1.7	Organisation der HTML-Files . . . . .	9
1.7.1	Aufteilung der Information auf einzelne HTML-Files . . . . .	10
1.7.2	Filenamen und Directories . . . . .	10
1.8	Wie kann ich meine Web-Pages erstellen? . . . . .	11
1.8.1	Statisch oder dynamisch . . . . .	11
1.8.2	Methoden und Software zum Erstellen von HTML-Files . . . . .	12
1.8.3	Integrierte Systeme und Umwandlungsprogramme . . . . .	12
1.8.4	Hinweise zur Verwendung von MS-Word . . . . .	14
1.8.5	Direktes Editieren oder Nachbearbeitung von HTML-Files . . . . .	15
1.8.6	HTML-Editoren . . . . .	16
1.9	Wie kann ich meine HTML-Files im WWW veröffentlichen? . . . . .	17
1.9.1	Testen und Validieren der HTML-Files . . . . .	17
1.9.2	Abspeichern der HTML-Files . . . . .	18
1.9.3	Bekanntmachen der HTML-Files . . . . .	19
1.9.4	Aktualisieren der Informationen . . . . .	19
1.9.5	Löschen von HTML-Files . . . . .	20
<b>2</b>	<b>Text-Elemente</b>	<b>21</b>
2.1	Aufbau des HTML-Files <code>&lt;head&gt; &lt;title&gt; &lt;body&gt;</code> . . . . .	21
2.2	Absatz (paragraph) <code>&lt;p&gt;</code> und Zeilenumbruch . . . . .	22
2.3	Zeilenwechsel (line break) <code>&lt;br&gt;</code> . . . . .	23
2.4	Seitenwechsel (page break) . . . . .	23
2.5	Buchstaben und Sonderzeichen (entity) . . . . .	24
2.6	Hervorgehobene Wörter (emphasis) <code>&lt;em&gt; &lt;strong&gt;</code> . . . . .	26
2.7	Hervorgehobene Absätze und Zitate (quote) <code>&lt;blockquote&gt;</code> . . . . .	26
2.8	Überschriften (heading) <code>&lt;h1&gt; &lt;h2&gt; &lt;h3&gt;</code> . . . . .	27

2.9	Listen und Aufzählungen . . . . .	27
2.9.1	Nicht numerierte Liste (unordered list) <ul> . . . . .	27
2.9.2	Numerierte Liste (ordered list) <ol> . . . . .	28
2.9.3	Liste von Beschreibungen (definition list) <dl> . . . . .	29
2.10	Formatierte Texteingabe (preformatted) <pre> . . . . .	30
2.11	Tabelle (table) <table> . . . . .	31
2.12	Mathematik und Chemie <sub> <sup> . . . . .	34
<b>3</b>	<b>Hypertext-Links</b>	<b>35</b>
3.1	Verweise zu anderen Informationen <a href> . . . . .	36
3.2	URL (Uniform Resource Locator) . . . . .	37
3.2.1	Absolute URLs im WWW . . . . .	38
3.2.2	Relative URLs im WWW . . . . .	39
3.2.3	URLs für andere Internet-Services . . . . .	39
3.3	Listen von Verweisen . . . . .	40
3.4	Markierungen innerhalb eines HTML-Files <a name> . . . . .	40
3.5	Inhaltsverzeichnisse . . . . .	41
<b>4</b>	<b>Bilder und Töne</b>	<b>43</b>
4.1	Bilder – ja oder nein? . . . . .	43
4.2	Die Wirkung auf die Menschen . . . . .	44
4.3	Inline-Bilder <img> <object> . . . . .	45
4.4	Inline-Objekte <object> . . . . .	47
4.5	Externe Bilder, Töne, Filme . . . . .	50
4.6	Kleine und große Bilder (thumbnails) . . . . .	51
<b>5</b>	<b>Layout und Spezialeffekte</b>	<b>52</b>
5.1	Schönes Layout mit HTML – wie geht das? . . . . .	53
5.1.1	Logisches Markup und Layout-Hinweise . . . . .	53
5.1.2	Wenn die Glocken locken... . . . . .	54
5.1.3	Norm oder nicht Norm, das ist hier die Frage . . . . .	54
5.2	Klassen (class) und Style-Sheets <style> . . . . .	55
5.3	Schrift . . . . .	57
5.3.1	Schriftarten . . . . .	57
5.3.2	Schriftgrößen . . . . .	58
5.4	Farben . . . . .	58
5.4.1	Wie werden Farben sichtbar? . . . . .	58
5.4.2	Farben mit <em> <strong> class <style> . . . . .	60
5.4.3	Farben mit <body> <em> <strong> <font> . . . . .	61
5.5	Anordnung (align) . . . . .	62
5.5.1	Linksbündig, rechtsbündig, zentriert . . . . .	63
5.5.2	Unten, oben, neben Bildern . . . . .	65
5.6	Abstände . . . . .	67
5.6.1	Horizontale Abstände . . . . .	67

5.6.2	Einrückungen . . . . .	68
5.6.3	Vertikale Abstände . . . . .	69
5.7	Trennlinien (horizontal rule) <hr> . . . . .	69
5.8	Numerierungen in Listen <ol> . . . . .	70
5.9	Frames <frameset> <frame> <noframes> . . . . .	72
5.10	Navigationshilfen <link> . . . . .	74
5.11	Schlagwörter für Suchhilfen eintragen <meta> . . . . .	76
5.12	Interaktion mit dem Benutzer . . . . .	76
5.12.1	Aktionen am Server (CGI, SSI, ASP, PHP) . . . . .	77
5.12.2	Zugriffe zählen . . . . .	79
5.12.3	Formulare <form> . . . . .	80
5.12.4	Image-Maps (ismap <map> usemap) . . . . .	82
5.12.5	Aktionen am Client (Java, JavaScript, Active-X, DHTML, XSL) . . . . .	84
5.12.6	Java-Applets <applet> <param> <object> . . . . .	85
5.12.7	JavaScript, <script> <noscript> . . . . .	87
5.12.8	Paßwort-Schutz und Sicherheit (SSL, https) . . . . .	89
5.12.9	Dynamische Web-Pages und Datenbanken . . . . .	89
5.12.10	Electronic Mail (mailto) . . . . .	90
<b>6</b>	<b>Geschichte und Geschichten</b>	<b>91</b>
6.1	Vom Elektronengehirn zum World Wide Web . . . . .	91
6.2	Von der Textverarbeitung zur Hypertext Markup Language . . . . .	93
6.3	Entwicklungen für die Zukunft . . . . .	94
<b>7</b>	<b>Entwicklungen für die Zukunft</b>	<b>95</b>
7.1	XML – die einfachere SGML . . . . .	95
7.2	XHTML – die neue HTML . . . . .	97
7.3	MathML für Mathematik . . . . .	98
7.4	CML für Chemie . . . . .	98
7.5	WAP und WML für Handys . . . . .	98
<b>8</b>	<b>Glossar</b>	<b>100</b>
8.1	WWW – Was ist das? . . . . .	100
<b>9</b>	<b>Die roten Fäden der Ariadne</b>	<b>110</b>
9.1	Wegweiser . . . . .	110
9.1.1	Ich möchte den kompletten Text lesen. . . . .	111
9.1.2	Ich möchte das komplette Handbuch auf Papier ausdrucken. . . . .	111
9.1.3	Ich möchte einen bestimmten HTML-Befehl nachschlagen. . . . .	111
9.1.4	Ich möchte kurz erfahren, wofür WWW und HTML überhaupt gut sind. . . . .	111
9.1.5	Ich kenne WWW und möchte alles über HTML erfahren. . . . .	112
9.1.6	Ich möchte rasch eine kurze Information im WWW veröffentlichen. . . . .	112

9.1.7	Ich möchte eine umfangreiche Information gut strukturiert im WWW veröffentlichen. . . . .	112
9.1.8	Ich möchte eine Liste von "heißen" Verbindungen zusammenstellen. . . . .	112
9.1.9	Ich möchte, daß meine Informationen bunt und modern aussehen	113
9.1.10	Ich möchte, daß meine Informationen von vielen Leuten gelesen werden. . . . .	113
9.1.11	Ich möchte mehr über die Hintergründe von WWW und HTML erfahren. . . . .	113
9.1.12	Ich möchte etwas anderes. . . . .	113
9.2	Referenzen . . . . .	114
9.3	Liste der HTML-Befehle . . . . .	116

---

# 1. Die Grundlagen

---

- →Vorwort
- →WWW – Was ist das?
- →Inhalt
  - →Was darf ich im WWW veröffentlichen?
  - →Was soll ich im WWW veröffentlichen?
  - →Frisch geplant ist halb gewonnen!
- →Inhalt und Form
- →Richtige HTML
  - →Was ist richtig?
  - →Weltweite Zusammenarbeit oder Firmenabhängigkeit
- →Format der Markup-Befehle (HTML-Tags)
- →Aufbau eines HTML-Files `<html> <head> <body>`
- →Organisation der HTML-Files
  - →Aufteilung der Information auf einzelne HTML-Files
  - →Filennamen und Directories
- →Wie kann ich meine Web-Pages erstellen?
  - →Statisch oder dynamisch
  - →Methoden und Software für HTML-Files
  - →Integrierte Systeme und Umwandlungsprogramme
  - →Verwendung von MS-Word
  - →Direktes Editieren, Muster-Files und Nachbearbeitung
  - →HTML-Editoren
- →Wie kann ich meine HTML-Files im WWW veröffentlichen?
  - →Erstellen der HTML-Files
  - →Testen und Validieren
  - →Abspeichern der HTML-Files
  - →Bekanntmachen der HTML-Files
  - →Aktualisieren der Informationen
  - →Löschen von HTML-Files
- siehe auch →Geschichte und →Referenzen

## 1.1 WWW – Was ist das?

Erklärung der wichtigsten Grundbegriffe im →"WWWörterbuch" (Glossar):

- →WWW . →Internet
- →Server . →Client . →Web-Browser
- →Hypertext . →HTML . →WML

## 1.2 Inhalt des HTML-Files

- →Was darf ich im WWW veröffentlichen?
- →Was soll ich im WWW veröffentlichen?
- →Frisch geplant ist halb gewonnen!

### 1.2.1 Was darf ich im WWW veröffentlichen?

Texte, Bilder oder Töne über das WWW verfügbar zu machen, bedeutet, sie weltweit zu veröffentlichen (es sei denn, der Zugriff auf die Files wird durch spezielle Vorkehrungen am →WWW-Server auf bestimmte Benutzer mit Userid und Paßwort eingeschränkt). Deshalb müssen alle für Veröffentlichungen gültigen nationalen und internationalen Gesetze und Regelungen eingehalten werden - also z.B. das Urheberrecht, der Datenschutz, das Strafrecht, verschiedene Geheimhaltungspflichten, firmeninterne Regeln und vieles andere mehr.

Die meisten Institutionen, Universitäten, Firmen u.dgl. legen außerdem großen Wert darauf, dass der Inhalt aller Veröffentlichungen dem Ansehen der Institution angemessen ist und mit ihrem Public-Relations-Konzept in Einklang steht, und dass das Layout ihrer "Corporate Identity" entspricht.

Das Speichern von Informationen im World Wide Web bedeutet *nicht* automatisch den Verzicht des Autors auf das Urheberrecht und die Nutzungsrechte ("Copyright"). Alle kreativen Werke (Texte, Töne, Bilder und dergleichen) sind weltweit urheberrechtlich geschützt, auch wenn sie in Büchern, in Rundfunksendungen oder im World Wide Web veröffentlicht wurden.

### 1.2.2 Was soll ich im WWW veröffentlichen?

Das WWW eignet sich *nicht* für Informationen, die man den Adressaten "aufdrängen" will, wie z.B. dringende Hinweise oder Werbungen. Das WWW eignet sich *nur* für Informationen, die von den Interessenten "freiwillig" aufgesucht und abgefragt werden – also für Informationen, die nützlich und interessant sind oder die unterhalten und Freude bereiten.

Außerdem ist es wichtig, die Informationen so zu strukturieren und →anzulegen, dass sie von den Interessenten auch →gefunden werden.

Bitte, bedenken Sie:

- Falsche oder nicht mehr aktuelle Informationen sind schlechter als gar keine Information.
- Eine nicht ganz so schön formatierte, aber verfügbare Information ist besser als gar keine Information.

### 1.2.3 Frisch geplant ist halb gewonnen!

Bevor Sie Ihre Web-Pages nach allen Regeln der Kunst erstellen, sollten Sie ein Konzept für den Inhalt, die Struktur und das Layout Ihrer Informationen erstellen. Zumindest müssen Sie die folgenden Punkte überlegen:

- Welchen Zweck und welches **Ziel** will ich erreichen? Welche Zielgruppe will ich erreichen?
- **Welche Informationen** will ich veröffentlichen? Welche Informationen kann und darf ich veröffentlichen? Wie kann ich sicherstellen, dass meine Informationen richtig sind und laufend aktualisiert werden?
- **Wie** kann ich meine Informationen am besten strukturieren und auf einzelne →Web-Pages aufteilen? Wie kann ich sie am besten und am verständlichsten formulieren? Wie kann ich sie am besten und am übersichtlichsten anordnen? Welches →Layout eignet sich dafür am besten? Wie kann ich erreichen, dass meine Informationen gut lesbar sind und schön aussehen?
- Mit welchen **anderen** Informationen hängen meine Informationen zusammen? Sind sie Teil eines umfassenderen Informationssystems? Wie kann ich Inhalt, Struktur und Layout meiner Informationen mit denen der anderen Informationen in Einklang bringen?
- Welche **Leser** will ich erreichen? Welche Leser werde ich erreichen? Welche Interessen, Vorkenntnisse und Erwartungen haben meine Leser? Welche Informationsstrukturen und Layouts sind sie gewohnt? Welche Sprachen sprechen sie? Welche →Client-Software, Hardware und Netzverbindungen werden sie für den Zugriff auf meine Informationen verwenden? Wie werden sie meine Informationen finden, welche →Zugriffswege (Links) und →Suchhilfen werden sie verwenden?
- Was **kostet** die Veröffentlichung meiner Informationen? Wieviel Speicherplatz und welche Software werde ich auf meinem →Server brauchen? Wie hoch wird die Netzbelastung durch den Zugriff auf meine Informationen sein? Wie kurz werden die Übertragungszeiten meiner Informationen zu den →Client-Rechnern meiner Leser sein?

Es ist wünschens- und empfehlenswert, beim Konzept für die Informationsstruktur und beim Layout-Design die Hilfe von professionellen Fachleuten in Anspruch zu nehmen – oder sich zumindest andere Informationssysteme zum Vorbild zu nehmen, die von solchen professionellen Designern entworfen wurden.

Es ist technisch sehr leicht, Web-Pages zu veröffentlichen, die nur geringen Informationsgehalt aufweisen, unlogisch strukturiert, verwirrend angeordnet, schlampig formuliert, schwer verständlich oder schwer auffindbar sind, sachliche oder sprachliche Fehler enthalten, unüberlegt und unprofessionell aussehen oder mit der Mehrzahl der Web-Browser gar nicht richtig gelesen werden können. Die vorliegende →HTML-Einführung soll Ihnen helfen, diesbezügliche Fehler zu vermeiden und es besser zu machen, und weitere Hinweise zu Web-Design und WWW-Projekten finden Sie bei →Stefan Münz.

### 1.3 Inhalt und Form

Wenn Sie einen Text mit einem Textverarbeitungsprogramm wie z.B. MS-Word für das Ausdrucken auf Ihrem lokalen Drucker erstellen, können Sie das Layout optimal für das verwendete Papierformat und für die auf Ihrem Drucker verfügbaren Schriftgrößen und Schriftarten einrichten.

Bei einem über das →World Wide Web veröffentlichten File hingegen können Sie nicht wissen, mit was für einer →Client-Software und auf was für einem Bildschirm, mit welchen Fenstergrößen und mit welchen Schriftarten und Schriftgrößen die verschiedenen Benutzer diese Information lesen werden.

Sie müssen deshalb den Inhalt ihrer Information in seiner *logischen Struktur* festlegen und nicht bloß in seinem sichtbaren Aussehen ("Layout").

HTML unterstützt zu diesem Zweck ein "logisches Markup", bei dem die logische Bedeutung der Text-Teile so festgelegt wird, dass sie vom jeweiligen →Web-Browser in der für den Benutzer (→Client) optimalen Form dargestellt werden können. Außerdem verwenden →Suchhilfen die HTML-Befehle dazu, um zu entscheiden, welche Wörter bei der Stichwort-Suche berücksichtigt werden sollen – auf Grund ihrer logischen Bedeutung.

Verschiedene →Web-Browser stellen die im Informationssystem gespeicherten Informationen in ganz verschiedener Art und Weise dar: Die meisten unterstützen Bilder und verschiedene Schriftarten und -größen, es sind auch Browser denkbar, die die Informationen nur zeilenweise in einer einheitlichen Schrift oder in Blindenschrift oder akustisch (als gesprochene Texte) oder in Form von dreidimensionalen Virtual-Reality-Szenen darstellen. Die Steuerung erfolgt je nach dem verwendeten Gerät mit Tastatur, Maus, Track-Ball oder Touch-Screen oder auch mit der Fernbedienung des Fernsehapparats.

Hinweise für die Verwendung Ihrer Informationen sollten Sie deshalb stets so formulieren, dass sie für alle verschiedenen Web-Browser gelten, also z.B. weder "Drücken Sie im Feld ... die Return-Taste" noch "Klicken Sie mit der Maus auf ..." sondern "Wählen Sie ... aus".

#### **Wie funktioniert nun dieses "logische Markup"?**

Sie geben in Ihrem HTML-File am →WWW-Server mit den in den Text eingestreuten →HTML-Befehlen ("Tags") an, dass ein bestimmter Textteil eine Überschrift, ein anderer eine Erklärung, eine Aufzählung, ein hervorgehobenes Wort und dergleichen oder ein Hyper-Link oder ein Bild ist (wie, das lernen Sie in der vorliegenden →HTML-Einführung).

Diese Elemente werden dann vom →Web-Browser auf dem Bildschirm eines jeden Benutzers so dargestellt, dass das Layout diese logische Bedeutung richtig wiedergibt, und zwar so, wie es der Benutzer haben möchte - in der Anordnung, die er gewöhnt ist und die er übersichtlich findet, und mit der Schriftart, Schriftgröße und Farbe, die er je nach der Auflösung seines Bildschirms und der Sehschärfe seiner Augen am besten lesen kann.

Wenn Ihr HTML-File auf Ihrem eigenen Browser mit Ihren eigenen persönlichen Einstellungen (Optionen, Präferenzen) gut aussieht, dann bedeutet das noch lange

nicht, dass es auch bei anderen Lesern mit den von ihnen verwendeten Browsern und persönlichen Einstellungen gut aussieht oder überhaupt sinnvoll lesbar ist.

Wie Sie als Autor das Layout trotzdem mit HTML-Befehlen festlegen oder zumindest beeinflussen können, ist im Kapitel →"Layout und Spezialeffekte" beschrieben. In Fällen, bei denen das *genaue* Layout wichtig ist (z.B. bei Formularen oder Diagrammen), kann es eventuell sinnvoller sein, die Information nicht als HTML-File sondern als →PostScript- oder →PDF- oder →RTF-File oder als →GIF-Bild oder →Image-Map abzuspeichern.

Ein Beispiel, wie man es *nicht* machen sollte, finden Sie im Kapitel →"Wenn die Glocken locken".

## 1.4 Richtige HTML

### 1.4.1 Was ist richtig?

Wenn Sie wollen, dass Ihre Information von *allen* Interessenten gelesen werden kann, dann müssen Sie darauf achten, dass Ihre HTML-Files von allen gängigen →Web-Browsern richtig verarbeitet werden ("Norm" im Sinne von "normales Verhalten" oder "De-facto-Standard"). Ein guter Anhaltspunkt dafür sind die vom →W3-Consortium zusammengestellten HTML-Spezifikationen und die von den →HTML-Prüfprogrammen (Validatoren) ausgegebenen Warnungen. Die De-facto-Norm umfaßt einerseits →HTML 3.2 und andererseits diejenigen Teile von →HTML 4.0 und →CSS1, die schon jetzt von den meisten Browsern unterstützt werden.

Viele, aber nicht alle Web-Browser tolerieren viele, aber nicht alle Abweichungen von den strikten HTML-Sprachregeln. Wenn Ihr HTML-File also von Ihrem eigenen Browser richtig verstanden wird, dann bedeutet das noch nicht, dass es sich tatsächlich um syntaktisch richtige HTML handelt und dass es auch von anderen Lesern mit den von ihnen verwendeten Browsern gelesen werden kann.

Um sicherzustellen, dass Ihre HTML-Files tatsächlich brauchbar sind, müssen Sie deswegen →HTML-Prüfprogramme (Validatoren) verwenden, die eigens für diesen Zweck eingerichtet wurden (siehe die →Referenzen).

### 1.4.2 Weltweite Zusammenarbeit oder Firmenabhängigkeit

Das →World Wide Web baut, wie schon der Name sagt, auf der Idee einer weltweiten Zusammenarbeit auf: *Alle* Benutzer mit *allen* →Clients sollen auf *alle* Informationen und Services auf *allen* →Servern zugreifen können.

Dies kann selbstverständlich nur dann funktionieren, wenn alle Beteiligten die offiziellen oder inoffiziellen Normen für die Datenübertragung und die Datenformate genau einhalten: →TCP/IP, →HTTP, →HTML, →CSS1, →Java ...

Manche kommerzielle Firmen wie z.B. Netscape oder Microsoft bemühen sich, WWW-Software zu vertreiben, die zusätzliche Features aufweist, die über die vom W3-Consortium zusammengestellten →HTML-Spezifikationen hinausgehen oder ihnen, noch schlimmer, sogar widersprechen. Damit werden alle Benutzer, die diese "Extras" ausnützen wollen, gezwungen, die Software dieser Firma zu kaufen.

Die Einengung auf ein bestimmtes Produkt oder eine Firma ist in diesem innovativen Bereich besonders ungünstig, wie die bisherigen Erfahrungen lehren:

Hardware:

- 1988 bis 1994 wurden Informationssysteme vor allem im Text-Modus an Großrechnern abgefragt.
- Seit 1994 sind vor allem graphisch orientierte Web-Browser auf PCs im Einsatz.
- Ab 2000 oder 2001 werden die Web-Browser in Fernsehapparaten, Handy-Telefonen, Palmtop-Organizern und anderen Haushaltsgeräten immer mehr an Bedeutung gewinnen und die klassischen, auf PCs laufenden Web-Browser zumindest teilweise verdrängen.

Software:

- 1993 wurde überwiegend Gopher verwendet.
- 1994 wurde Gopher durch Mosaic und Lynx verdrängt.
- 1995 wurde Mosaic durch Netscape verdrängt.
- Ab 1996 versuchen Netscape und Microsoft Internet Explorer, einander zu verdrängen, mit jedes Jahr neuen Features, die auf den anderen bzw. älteren Versionen nicht funktionieren: Laufschriften, spezielle Bildformate, Animationen, Töne, Video-Sequenzen, Tabellen, Frames, Mail-Formulare, JavaScript, Java-Applets, Active-X, CSS1 und CSS2 Style-Sheets, dynamic HTML, XML, Math-HTML, PDF, VRML u.v.a.
- Daneben gibt es andere Browser wie Hotjava, Opera u.a. sowie ab 1998 WebTV für Fernsehapparate und ab 1999 WAP und WML für Handy-Telefone und Palmtop-Geräte.
- 2000 wird Netscape durch den AOL-Browser iPlanet und durch das Open-Source-Projekt Mozilla ersetzt.

Wenn Ihre Web-Pages auch noch nächstes Jahr brauchbar sein sollen und wenn Sie nicht einen großen Teil der Interessenten vom Zugriff auf Ihre Informationen ausschließen wollen, dann sollten Sie als Autor sich nicht auf Firmen-, Hardware- und Software-spezifische "Extrawürste" verlassen, sondern diese, wenn überhaupt, nur als Alternativen und Zusätze zu den HTML-Befehlen einsetzen, die auf *allen* Browsern ein richtig lesbares Ergebnis bewirken (→De-facto-Norm).

Beispiele für "Extrawürste", die nicht von allen Web-Browsern und Suchhilfen richtig verarbeitet werden, sind →<animate>, →<applet>, →<blink>, →<bgsound>, →<center>, →<embed>, →<font>, →<frame>, →<frameset>, →<layer>, →<marquee>, →<object>, →<script>, →<spacer>, →<style>, u.a.

*Wenn* Sie solche neue oder nicht genormte oder nicht von allen Web-Browsern unterstützte Elemente verwenden, dann achten Sie bitte darauf, dass Ihre Informationen trotzdem von allen Web-Browsern wenigstens einigermaßen brauchbar dargestellt werden können, z.B. indem Sie "Alternativen" wie ALT= oder NOFRAMES und dergleichen einfügen. Hinweise für solche hilfreiche Tricks finden Sie in den Kapiteln über →Tabellen, →Zentrierung, →Farbe, →Bilder, →Frames, →Image-Maps und bei allen →Layout- und Spezialeffekten. Im Zweifelsfall halten Sie sich lieber an die alte Grundregel der Informationsverarbeitung:

Sei möglichst konservativ beim Schreiben  
und möglichst liberal beim Lesen!

## 1.5 Format der Markup-Befehle (HTML-Tags)

Die Sprache  $\rightarrow$ HTML baut auf der Syntax von  $\rightarrow$ SGML auf. Die Markup-Befehle ("Tags") werden vom normalen Text dadurch unterschieden, dass sie zwischen "spitzen Klammern", also zwischen Kleiner- und Größer-Zeichen eingeschlossen werden, in der Form

```
<xxx>
```

Manche dieser Befehle (Tags) haben auch Parameter (Attribute und Argumente), z.B. in einer Form wie

```
<xxx yyy=zzz>
```

Bei den Befehlen xxx und den Attributen yyy ist die Groß- oder Klein-Schreibung egal, bei den Argumenten zzz trifft dies nicht immer zu. *Bei  $\rightarrow$ XHTML müssen die Befehle und Attribute mit Kleinbuchstaben geschrieben werden.*

Wenn die Argumente Sonderzeichen enthalten, müssen sie in Anführungszeichen (Quotes) eingeschlossen werden:

```
<xxx yyy="zzz">
```

*Bei  $\rightarrow$ XHTML müssen alle Argumente in Quotes eingeschlossen werden.*

Einfache oder mehrfache Leerstellen oder Zeilenwechsel haben (von Spezialfällen abgesehen) dieselbe Bedeutung, d.h. sie wirken jeweils so wie eine Leerstelle.

Die meisten HTML-Befehle treten paarweise auf, mit einem "Start-Tag" der Form `<xxx>` und einem "End-Tag" der Form `</xxx>`. Diese Befehlspaare geben jeweils die Bedeutung des dazwischen liegenden Textes an. So ist z.B. der zwischen `<h1>` und `</h1>` stehende Text eine  $\rightarrow$ Überschrift.

Manche HTML-Befehle treten einzeln auf, also ohne einen End-Tag. Sie bezeichnen bestimmte Elemente, die zwischen dem Text stehen. So bedeutet z.B. `<hr>` eine  $\rightarrow$ Linie zwischen zwei Absätzen.

Die paarweisen HTML-Befehle müssen immer **richtig geschachtelt** werden. Dies soll an dem folgenden Beispiel durch das Einrücken der Eingabe-Zeilen veranschaulicht werden. (Was die einzelnen Befehle bedeuten, wird nicht hier sondern in den  $\rightarrow$ jeweiligen Kapiteln der HTML-Einführung beschrieben.)

```
<html>
  <head>
    <title> Der Titel des HTML-Files
  </title>
  </head>
  <body>
    <h1> Die Ueberschrift
  </h1>
    <p> Der erste Absatz.
  </p>
    <p> Ein
```

```

        <em> betontes
        </em>
        Wort im zweiten Absatz.
        Nach diesem Absatz kommt eine Linie:
    </p>
    <hr>
    <p> Das ist der letzte Absatz.
    </p>
</body>
</html>

```

Nicht erlaubt wäre also z.B. eine Konstruktion wie

```

<p>xxxxxx <em>xxxx</p>
<p>xxxx</em> xxxxxx</p>

```

Stattdessen müßte man es so schreiben:

```

<p>xxxxxx <em>xxxx</em></p>
<p><em>xxxx</em> xxxxxx</p>

```

In einigen Fällen darf der End-Tag weggelassen werden, so ist z.B. die explizite Angabe von `</p>` in den obigen HTML-Beispielen nicht unbedingt notwendig. *Bei →XHTML dürfen die End-Tags nicht weggelassen werden.*

Kommentare können in der Form

```

<!-- Kommentartext -->

```

eingefügt werden. Damit der Kommentartext tatsächlich von allen Browser-Versionen ignoriert wird, sollte er weder `--` noch `>` enthalten.

## 1.6 Aufbau eines HTML-Files `<html>` `<head>` `<body>`

Jedes HTML-File soll zumindest mit den folgenden Befehlen beginnen:

```

<html>
<head>
<title>Haupt-Ueberschrift</title>
<link rev=made href="mailto:username@hostname.domainname">
</head>
<body>
<h1>Haupt-&Uuml;berschrift</h1>

```

und mit den folgenden Befehlen enden:

```

</body>
</html>

```

Das gesamte HTML-File wird zwischen `<html>` und `</html>` eingeschlossen.

Zwischen `<head>` und `</head>` stehen die Angaben, die für die Verwaltung des HTML-Files notwendig sind (Titel, Autor und dergleichen).

Zwischen `<body>` und `</body>` steht die Information, die am Bildschirm des Benutzers dargestellt werden soll, also die  $\rightarrow$ Textelemente,  $\rightarrow$ Hypertext-Links,  $\rightarrow$ Bilder und Töne.

Zwischen `<title>` und `</title>` steht die Überschrift des HTML-Files, die in  $\rightarrow$ Bookmarks und  $\rightarrow$ Suchhilfen verwendet wird.

Zwischen `<h1>` und `</h1>` steht die  $\rightarrow$ Überschrift, die am Bildschirm zu Beginn des Textes erscheint.

Da sie nicht wissen können, über welche  $\rightarrow$ Hypertext-Links,  $\rightarrow$ Bookmarks oder  $\rightarrow$ Suchhilfen die Leser auf Ihre Information zugreifen werden, sollten Sie Titel, Überschriften und Inhalt Ihrer Files immer möglichst vollständig und auch ohne Kontext verständlich formulieren, also z.B. nicht bloß "Einleitung" sondern "Einleitung zur HTML-Einführung", und nicht bloß "Statistik-Abteilung" sondern "Abteilung für Statistik des Instituts für Mathematik an der Universität für Bodenkultur in Wien".

Zwischen `<title>` und `</title>` sollen (im Gegensatz zu `<h1>`) nur normale ASCII-Buchstaben stehen, keine  $\rightarrow$ Entities (deutsche Umlaute etc.) und keine HTML-Befehle.

Den Autor und andere Angaben, die sich auf die Web-Page beziehen, kann man innerhalb des `<head>` mit  $\rightarrow$ `<link>`,  $\rightarrow$ `<meta>` und  $\rightarrow$ `<style>` angeben.

Unabhängig davon ist es empfehlenswert, am Ende des lesbaren Textes im `<body>` den Namen und die  $\rightarrow$ Mail-Adresse des Autors, das Datum der Erstellung oder der letzten Aktualisierung, und ein Hypertext-Link auf die  $\rightarrow$ Startseite der Institution anzugeben.

Als erste Zeile des Files (vor dem Befehl `<html>`) kann man die  $\rightarrow$ HTML-Version mit einem  $\rightarrow$ SGML-Befehl wie z.B.

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

oder

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
```

angeben. Wenn diese Angabe fehlt, wird HTML 2.0 angenommen.

## 1.7 Organisation der HTML-Files

- $\rightarrow$ Aufteilung der Information auf einzelne HTML-Files
- $\rightarrow$ Filenamen und Directories

### 1.7.1 Aufteilung der Information auf einzelne HTML-Files

Bei der Aufteilung der Information auf eines oder mehrere HTML-Files müssen Sie einen Kompromiß zwischen den folgenden Vor- und Nachteilen wählen.

Für mehrere kleine Files und gegen wenige allzu große Files sprechen die folgenden Punkte:

- Große Files brauchen unter Umständen (z.B. über Telefonleitungen) sehr lange Übertragungszeiten zum Leser.
- Kleine Files erscheinen am Bildschirm übersichtlicher als lange Files.
- Die Leser können leichter einzelne Detail-Informationen abspeichern oder ausdrucken.

Für wenige größere Files und gegen allzu viele kleine Files sprechen die folgenden Punkte:

- Die Leser können mit den Textsuche-Befehl ihres Web-Browsers nach Stichworten innerhalb des gesamten Files suchen.
- Leser, die die gesamte Information abspeichern oder ausdrucken wollen, können dies einfach und mit nur wenigen "Maus-Klicks" erreichen.

Wann ein File als "klein" oder "groß" gilt, hängt von der Anwendung und vom Stand der Technik ab. Zur Orientierung: Ein File von 50 kBytes braucht für die Übertragung über eine Modem-Leitung mit 14.400 bit/s ungefähr eine Minute. Viele Benutzer werden bei Wartezeiten von mehr als 6 Sekunden ungeduldig...

### 1.7.2 Filenamen und Directories

Manche →Web-Browser verarbeiten HTML-Files nur dann richtig, wenn die Extension des Filenamens ".html" lautet. HTML-Files sollten daher immer unter **Filenamen** der Form "**xxxxx.html**" oder "**xxxxx.htm**" →abgespeichert werden.

Unter Betriebssystemen, die wie MS-DOS nur 3 Zeichen lange Extensions erlauben, können die Files *nur* unter Filenamen der Form "**xxxxx.htm**" abgespeichert werden. In manchen Fällen kann man dann trotzdem in den →URLs (→Hypertext-Links, →Bookmarks und →Suchhilfen) die Filenamen als "xxxxx.html" mit der kompletten Extension angeben, die Extension wird dann vom Betriebssystem automatisch richtig gekürzt; in vielen Fällen funktioniert das aber nicht.

Es wird empfohlen, zusammengehörende HTML-Files jeweils in einem **Directory** gemeinsam abzuspeichern und in Querverweisen zwischen ihnen nur →relative URLs zu verwenden, damit die Files auch nach einer eventuellen Übertragung auf einen anderen →WWW-Server oder in einen anderen Bereich verwendet werden können.

## 1.8 Wie kann ich meine Web-Pages erstellen?

- →Statisch oder dynamisch
- →Methoden und Software
- →Integrierte Systeme und Umwandlungsprogramme
- →Verwendung von MS-Word
- →Direktes Editieren, Muster-Files und Nachbearbeitung
- →HTML-Editoren

**Bevor** Sie Ihre HTML-Files erstellen und →veröffentlichen, müssen Sie sich ein →Konzept für den Inhalt, die Struktur und das Layout ihrer Informationen überlegen.

### 1.8.1 Statisch oder dynamisch

Es gibt drei Arten, Informationen über das →WWW zur Verfügung zu stellen:

- einzelne HTML-Files, die selbständig erstellt und gewartet werden. Dies wird als "**statische Web-Pages**" bezeichnet. Diese eher altmodische und mühsame Methode wird in den folgenden Abschnitten detailliert beschrieben.
- direkter Zugriff auf die in Datenbanken oder Workflow-Systemen gespeicherten Informationen über ein Web-Interface, das die aktuellen Inhalte im HTML-Format an den Web-Browser ausgibt. Dies wird als "**dynamische Web-Pages**" bezeichnet. Mehr darüber finden Sie im Kapitel über →Dynamische Web-Pages und Datenbanken.
- statische oder dynamische →XML-Files, die entweder direkt vom Web-Browser verstanden werden oder am →Web-Server automatisch in →HTML oder →XHTML umgewandelt werden. Mehr darüber finden Sie in den Kapiteln über →XML und →XHTML und in der →XML Kurz-Info.

Außerdem ist jeweils zu überlegen, ob die Informationen

- nur in →HTML für klassische Web-Browser oder
- auch in →WML für Handys

angeboten werden sollen. Mehr über WML finden Sie im Kapitel über →WAP und WML und in der →WML-Einführung.

In den folgenden Abschnitten wird das Erstellen von statischen HTML-Files detailliert beschrieben. Die darin angeführten Grundsätze gelten analog auch für die anderen Möglichkeiten.

### 1.8.2 Methoden und Software zum Erstellen von HTML-Files

Für das Erstellen von HTML-Files gibt es grundsätzlich zwei Wege:

- Die **einfachste** Möglichkeit ist es, die Inhalte mit der gewohnten Textverarbeitung wie z.B. →Word oder WordPerfect zu erstellen und mit →integrierten Systemen wie z.B. MS-Office97, Lotus Smartsuite, Lotus Domino Server, Oracle Web-Server u.a. oder mit →Umwandlungsprogrammen wie z.B. Word Internet Assistant, RTFtoHTML, LaTeXtoHTML u.a. in →HTML-Files umzuwandeln.

Damit können ohne großen Aufwand und ohne tiefere HTML-Kenntnisse *einfache* Web-Pages erstellt werden.

Spezielle Layouts oder auf allen Seiten wiederkehrende Kopf- und Fußzeilen, Logos, Links etc. können dabei durch die Verwendung von Muster-HTML-Files eingebaut werden.

- Der **aufwendigere** Weg ist es, die HTML-Files mit speziellen →HTML-Editoren oder einfach →als Text-File mit eingebetteten HTML-Befehlen →direkt zu erstellen.

Damit hat man die volle Kontrolle über *alle* von HTML gebotenen Möglichkeiten. Allerdings sind dafür etwas genauere HTML-Kenntnisse notwendig, wie sie in den weiteren Kapiteln der HTML-Einführung vermittelt werden.

In *allen* Fällen empfiehlt es sich, nicht für jede Web-Page "das Rad neu zu erfinden", sondern Muster-HTML-Files zu verwenden, die alle HTML-Befehle für den Aufbau und die Struktur der Web-Page, mit Überschrift und Fußzeile, Firmen-Logo, Link zur Home-Page, Mail-Adresse des Autors und dergleichen enthalten und dann nur mehr mit den jeweiligen Daten "gefüttert" werden müssen.

Außerdem kann man durch den Einsatz von →Style-Sheets eine möglichst einheitliche und der →Corporate Identity entsprechende Darstellung der Web-Pages erreichen.

### 1.8.3 Integrierte Systeme und Umwandlungsprogramme

Neuere Softwarepakete wie z.B. MS-Office97, Lotus Smartsuite, Lotus Domino, Oracle Web-Server u.a. haben Möglichkeiten für die Umwandlung von Texten, Spreadsheets und Datenbanken in HTML-Files bereits integriert.

Bei etwas älteren Versionen von Software-Produkten wie z.B. →Word oder Excel können solche Umwandlungsprogramme nachträglich installiert werden, z.B. der Word Internet Assistant.

Ähnliche Umwandlungsprogramme gibt es auch für andere Software-Produkte wie z.B. Excel und für andere File-Formate wie z.B. RTF (Rich Text Format) oder LaTeX.

Vorgangsweise:

- →wenn die Umwandlungs-Software sehr gut funktioniert
- →wenn die Umwandlungs-Software weniger gut funktioniert

Wenn die →Umwandlungs-Software *alle* benötigten HTML-Elemente richtig wiedergibt, läuft das Erstellen der Web-Pages typisch so ab (am Beispiel der Textverarbeitung mit MS-Word):

1. Ein Muster-HTML-File mit dem Filetyp "HTML" in Word öffnen (mit automatischer Umwandlung ins Word-Format),
2. den Inhalt mit den gewohnten →Word-Befehlen an den richtigen Stellen eingetippt oder ein bereits fertiges →Word-Dokument einfügen,
3. das Ergebnis als Word-Dokument `xxxx.doc` speichern (für künftige Weiterbearbeitung),
4. das Dokument als HTML-File `xxxx.htm` mit dem Datei-Typ "HTML" speichern. Dabei werden alle wichtigen Textelemente wie Absätze, Überschriften, Listen usw. und die deutschen Umlaute und scharfen s automatisch in die entsprechenden HTML-Befehle umgewandelt.

Damit die automatische Umwandlung richtig funktionieren kann, müssen allerdings gewisse →Grundregeln der Textverarbeitung eingehalten werden (siehe →Hinweise für Word).

Außerdem muß das HTML-File fast immer noch →kontrolliert und eventuell →nachbearbeitet werden, um sicherzustellen, dass es sich um ein →wirklich brauchbares HTML-File handelt und dass bei der automatischen Umwandlung nicht wichtige Eigenschaften verloren gegangen sind.

Eventuell kann man diese Nachbearbeitung auch mit einem Batch-Script unterstützen, das im HTML-File gewisse Korrekturen automatisch durchführt. Damit kann man

- einerseits den HTML-Code verbessern, indem man z.B. störende →width-Angaben oder →<font>-Tags entfernt und →Software-spezifische, nicht allgemein brauchbare Code-Elemente entfernt oder durch allgemein verwendbare →richtige HTML-Befehle ersetzt; und
- andererseits in das HTML-File der →Corporate Identity entsprechende →Style-Sheets, Kopf- und Fußzeilen, das Firmen-Logo, ein →Link auf die Startseite (Home-Page oder Inhaltsverzeichnis), den Namen und die →Mail-Adresse des Autors, das Datum der Erstellung bzw. letzten Änderung und dergleichen einfügen.

Wenn die →Umwandlungs-Software *nicht alle* benötigten HTML-Elemente richtig wiedergibt, was leider oft der Fall ist, dann empfiehlt sich eher die folgende Vorgangsweise:

- Nur der Textteil der Web-Page wird →wie oben mit Word erstellt und automatisch in ein "Rumpf-HTML-File" umgewandelt.

- Dieses "Rumpf-HTML-File" wird dann durch →direktes Editieren mit einem →Text-Editor (oder als →Text-File in Word) mit den Kopf- und Fuß-Teilen und allen sonstigen "schwierigen" Elementen des Muster-HTML-Files zu einem →richtigen HTML-File zusammengesetzt.

#### 1.8.4 Hinweise zur Verwendung von MS-Word

Diese Hinweise gelten konkret für MS-Word. Analoge Regeln gelten für die Verwendung anderer Textverarbeitungs-Programme wie WordPerfect, AmiPro usw.

Wenn Sie einen Text mit MS-Word für das Ausdrucken auf Ihrem **lokalen** Drucker erstellen, können Sie das Layout optimal für das verwendete Papierformat und für die auf Ihrem Drucker verfügbaren Schriftgrößen und Schriftarten einrichten.

Bei einem am Web-Server für das Internet gespeicherten File hingegen können Sie nicht wissen, mit was für einer Software und auf was für einem Bildschirm und mit welchen Fenstergrößen und Schriftarten und Schriftgrößen die verschiedenen Benutzer **in aller Welt** diese Information lesen werden. Sie müssen daher das Layout in seiner logischen Struktur und nicht bloß in seinem sichtbaren Aussehen definieren.

Die automatischen oder halbautomatischen →Umwandlungs-Programme wie Word Internet Assistant, Office 97 etc. funktionieren nur dann richtig, wenn Sie die Grundregel einhalten, immer den jeweils **logisch richtigen** Befehl zu verwenden:

- Verwenden Sie die Return-Taste nur für das Ende eines Absatzes, nicht für Zeilenwechsel innerhalb von Absätzen.
- Lassen Sie den Zeilenumbruch möglichst automatisch von MS-Word machen, und legen Sie in Ausnahmefällen Zeilenwechsel innerhalb von Absätzen mit Shift-Return (nicht mit Return) fest.
- Versuchen Sie Einrückungen, Zentrierungen usw. nicht durch eine bestimmte Anzahl von eingefügten Leerstellen oder Tabulatoren zu erreichen, sondern mit den entsprechenden Word-Befehlen für die Absatz-Formatierung.
- Stellen Sie Überschriften nicht durch Standard-Absätze mit größerer oder fetterer Schrift dar, sondern verwenden Sie dafür die Formatvorlagen "Überschrift1" usw.
- Erzeugen Sie Listen, Aufzählungen und Numerierungen nicht durch trickreich modifizierte Standard-Absätze, sondern verwenden Sie dafür die entsprechenden Word-Befehle.
- Versuchen Sie Tabellen nicht durch eine bestimmte Anzahl von eingefügten Leerstellen oder Tabulatoren zu erzeugen sondern mit den Word-Befehlen für Tabellen.
- Setzen Sie Seitennummern und sonstige Kopf- und Fußzeilen nicht in den normalen Textbereich sondern mit den Word-Befehlen für Kopf- und Fußzeilen.

Damit die Umwandlung von Word nach HTML erfolgreich funktionieren kann, sollen Sie im Word-Dokument möglichst nur die Elemente verwenden, für die es eine Entsprechung in HTML gibt:

- Verwenden Sie zur Hervorhebung von Textbestandteilen Fett- oder Kursivschrift und *nicht* andere Schriftarten.
- Verwenden Sie nur die Sonderzeichen, die im Iso-8859-1-Code definiert sind, und nicht irgendwelche für Macintosh, DOS oder MS-Windows spezifische sonstige Zeichen, also z.B. nur normale Doublequotes und Apostrophe, keine deutschen, englischen oder französischen einfachen oder doppelten Anführungszeichen, keine griechischen Buchstaben oder Mathematik-Symbole, und nur normale Minus-Zeichen, keine längeren Binde- oder Gedankenstriche.
- Verwenden Sie keine Word-Zeichenfunktionen.
- Wenn Sie Bilder einbinden wollen, lassen Sie sich von einem Graphik-Software-Experten "richtige" Web-Bilder erstellen (GIF- oder JPEG-Format, geeignete Farbpalette und Pixelgröße). Die von Word automatisch generierten Bilder haben oft eine zu große, am Druckformat orientierte Auflösung.
- Vermeiden Sie das "automatische" Inhaltsverzeichnis oder die automatische Numerierung der Überschriften. Manche Umwandlungsprogramme wie z.B. rtf2html erzeugen stattdessen automatisch ein "anklickbares" und richtig geschachteltes Inhaltsverzeichnis über alle mit den Formatvorlagen "Überschrift 1 bis 3" markierten Überschriften.
- Formulieren Sie Verweise auf andere Stellen im Dokument nicht mit Seiten- oder Kapitelnummern (wie "siehe Seite ..."), sondern schreiben Sie statt dessen "siehe Abschnitt *Titel des Abschnittes*" oder "siehe den Abschnitt über *Thema des Abschnittes*". Das kann dann bei der Nachbearbeitung in einen →Hyperlink verwandelt werden.
- Tragen Sie unter "Datei-Info" (im Menü "Datei") die Felder "Titel", "Autor" und "Stichwörter" richtig ein, die werden von manchen Umwandlungsprogrammen automatisch in den →<head> des HTML-Files eingetragen und dann von Bookmarks und Suchmaschinen verwendet.

In Fällen, bei denen das genaue Layout wichtig ist (z.B. bei Formularen), kann es sinnvoll sein, am Web-Server das RTF-File oder PostScript-File abzuspeichern, statt ein HTML-File zu erzeugen.

### 1.8.5 Direktes Editieren oder Nachbearbeitung von HTML-Files

HTML-Files sind Text-Files, die den Text mit eingestreuten →HTML-Befehlen enthalten. Diese Files können einfach mit einem normalen **Editor** wie z.B. Notepad oder als "Text-Files" mit Textverarbeitungsprogrammen wie z.B. Word erstellt werden.

Dies wird wesentlich vereinfacht, wenn Sie von Muster-Files ausgehen, die bereits die HTML-Befehle für Aufbau, Struktur und Layout der Web-Page enthalten, und dann nur mehr den Inhalt zwischen die HTML-Befehle "einfüllen" müssen.

Es gibt auch →HTML-Editoren, mit denen man Web-Pages durch Anklicken der Formatierungselemente in Pull-Down-Menüs direkt am Bildschirm erstellen kann.

Im Fall von Textverarbeitungsprogrammen wie Word müssen Sie darauf achten, dass das File mit dem Dateityp "nur Text" oder "HTML" gespeichert wird und nicht als binäres Word-File (DOC-File). Zu diesem Zweck müssen Sie je nachdem, welche

Version von Word Sie verwenden und ob eine →automatische HTML-Umwandlung installiert ist oder nicht,

- entweder das HTML-File in Word mit dem Dateityp "nur Text" öffnen und speichern (nicht mit dem Dateityp "HTML" oder "Word")
- oder das HTML-File in Word mit dem Dateityp "HTML" öffnen und die "Ansicht" auf "HTML-Source" ändern ("view source") und dann das File wieder mit dem Dateityp "HTML" speichern.

Das analoge gilt für andere Textverarbeitungs-Software.

Falls Ihre Word-Version diese Tricks nicht unterstützt, können Sie eventuell einen anderen Text-Editor verwenden wie z.B. den "Notepad".

Falls Ihr Windows-System so konfiguriert ist, dass es bei Text-Files automatisch eine Extension .txt anhängt, Sie also einen Filenamem der Form xxx.htm.txt erhalten würden, müssen Sie den Filenamem zwischen Doublequotes einschließen, also in der Form "xxx.htm" angeben.

### 1.8.6 HTML-Editoren

Mit HTML-Editoren kann man Web-Pages durch Anklicken der Formatierungselemente in Pull-Down-Menüs direkt am Bildschirm schreiben - analog zum Erstellen von Texten mit Word oder von Graphiken mit Corel Draw. Meist ist auch eine Möglichkeit eingebaut, das Aussehen der Web-Page zumindest mit einem lokalen Web-Browser zu kontrollieren ("Preview"), was aber natürlich noch nicht viel über das Aussehen bei anderen Leuten auf deren Bildschirmen und mit deren Software aussagt.

Damit kann man etwas bequemer als beim →direkten Editieren und ohne allzu genaue Kenntnisse der HTML-Befehle Web-Pages erstellen. Allerdings ist fast immer noch eine →Kontrolle oder Nachbearbeitung der HTML-Files notwendig.

Die meisten HTML-Editoren sind lizenz- und kostenpflichtig, aber meist nicht allzu teuer. Beispiele sind Homesite, Hot Dog, Microsoft Frontpage, HTMLed32 und viele andere (siehe auch →Referenzen).

**Nicht** empfehlenswert sind Editoren die, wie z.B. "Netscape Gold" und "Netscape Composer", aus firmenpolitischen Gründen keine →allgemein verwendbaren HTML-Files erzeugen, sondern spezielle Tricks in die HTML-Files einbauen, damit sie nur mit den neuesten →firmeneigenen Web-Browsern gelesen werden können und nicht von Lesern, die ältere Versionen oder Software von anderen Herstellern verwenden, oder von →Suchmaschinen.

In solchen und ähnlichen Fällen kann es hilfreich sein, mit einem Batch-Script nachträglich alle solchen Tricks automatisch aus dem HTML-File zu entfernen und durch "normale" (→richtige) HTML-Befehle zu ersetzen.

Hinweise zum →Testen und Validieren der HTML-Files finden Sie im →folgenden Abschnitt.

## 1.9 Wie kann ich meine HTML-Files im WWW veröffentlichen?

Für die Veröffentlichung von Informationen über das WWW sind die folgenden Voraussetzungen und die folgenden Schritte notwendig:

1. Sie müssen ein →Konzept für den Inhalt und die Struktur Ihrer Informationen erarbeiten (oder erarbeiten lassen).
2. Sie brauchen eine Benutzungsbewilligung und ausreichend Speicherplatz auf einem →Web-Server.
3. Sie müssen die HTML-Files und eventuell darin aufgerufene Bilder und Töne, Style-Sheets, Programme oder Applets →erstellen, korrekturlesen, →testen und validieren.
4. Sie müssen die HTML-Files und eventuell darin aufgerufene Bilder und Töne, Style-Sheets, Programme oder Applets auf einem →WWW-Server →abspeichern.
5. Sie müssen die Verfügbarkeit der HTML-Files →bekanntmachen.

Sobald das File öffentlich zugänglich ist, sind Sie dafür verantwortlich, es laufend zu →aktualisieren und, wenn es nicht mehr relevant ist, zu →löschen.

### 1.9.1 Testen und Validieren der HTML-Files

Wenn Ihr HTML-File auf Ihrem eigenen →Web-Browser mit Ihren eigenen persönlichen Einstellungen (Optionen, Präferenzen) gut aussieht, dann bedeutet das noch lange nicht, dass es auch bei anderen Lesern mit den von ihnen verwendeten Browsern und persönlichen Einstellungen gut aussieht oder überhaupt sinnvoll lesbar ist.

Viele, aber nicht alle Web-Browser tolerieren viele, aber nicht alle Abweichungen von den HTML-Sprachregeln. Wenn Ihr HTML-File also von Ihrem eigenen Browser richtig verstanden wird, dann bedeutet das noch nicht, dass es sich tatsächlich um →richtige HTML handelt und dass es auch von anderen Lesern mit den von ihnen verwendeten Browsern gelesen werden kann.

**Bevor** Sie Ihre Web-Page für alle Welt zugänglich machen, sollen Sie Ihr Werk deshalb mit mindestens zwei verschiedenen Web-Browsern als "lokales File" testen: einem im Line-Mode (z.B. Lynx) *und* einem im Graphik-Mode (z.B. Internet-Explorer oder Netscape), oder zumindest mit verschiedenen persönlichen Einstellungen (z.B. mit und ohne Bilder) und verschiedenen Fenstergrößen und Hintergrundfarben.

Außerdem sollten Sie sich nicht auf die Fehlertoleranz der eigenen Web-Browser verlassen, sondern **HTML-Prüfprogramme** ("Validatoren") verwenden, die sowohl die →HTML-Sprachregeln als auch die Eigenheiten von *allen* Web-Browsern berücksichtigen und damit die Brauchbarkeit Ihrer HTML-Files wirklich sicherstellen – so ähnlich wie Spell-Checker-Programme (Rechtschreibprüfung) in der Textverarbeitung.

Solche Programme können am eigenen PC installiert werden, wesentlich einfacher und bequemer ist es aber, die →öffentlichen HTML-Validatoren zu verwenden, denen man nur den →URL des HTML-Files angibt und die einem dann sofort alle in diesem File enthaltenen Fehler oder ungünstigen Konstruktionen anzeigen (siehe die →Referenzen).

### 1.9.2 Abspeichern der HTML-Files

Sie selbst können Ihr HTML-File als "lokales File" mit Ihrem →Web-Browser lesen. Damit auch andere Personen darauf zugreifen können, müssen Sie es aber auf einem an das →Internet angeschlossenen →WWW-Server speichern.

Dieser Server-Computer muß ständig (24 Stunden täglich) mit dem Internet verbunden sein. Günstig ist ein Rechner mit einem Multi-Tasking-Betriebssystem wie Unix, schnellen Platten und einem leistungsfähigen Internet-Anschluß. Als Server-Software kommen mehrere kostenlose sowie kostenpflichtige Software-Produkte in Frage (siehe die →Referenzen).

Wenn Sie keinen solchen Rechner besitzen (z.B. nur einen PC oder nur eine Telefon-Verbindung zum Internet) oder keine solche Software betreiben wollen, müssen Sie einen Bereich auf einem WWW-Server "ausborgen" oder mieten. In vielen Universitäten und größeren Firmen bietet das EDV-Zentrum allen Mitarbeitern eine Möglichkeit dafür, und kommerzielle Internet-Provider stellen privaten Kunden und Firmen WWW-Server-Bereiche gegen Bezahlung zur Verfügung.

HTML-Files müssen immer unter den →richtigen Filenamen abgespeichert werden.

**Bevor** Sie Ihre HTML-Seite für alle Welt zugänglich machen, müssen Sie Ihr Werk korrekturlesen und →testen.

Die **Vorgangsweise** beim Speichern eines HTML-Files auf einem WWW-Server sollte so ähnlich wie im folgenden Beispiel ablaufen:

1. Das HTML-File am PC →erstellen.
2. Das File am PC mit einem Graphik-Browser wie z.B. Internet-Explorer oder Netscape →testen.
3. Das HTML-File mit FTP auf Ihren privaten Plattenbereich auf dem Unix-Rechner übertragen, auf dem das WWW-Server-Programm läuft.
4. Das File unter Unix mit einem Text-Browser wie z.B. Lynx →testen.
5. Fehler im HTML-File korrigieren und den Zyklus wiederholen.  
Wenn das File okay ist:
6. Das HTML-File in den öffentlichen Plattenbereich des →WWW-Servers kopieren und mit weltweiter Leseberechtigung versehen.
7. Das öffentliche File →testen und validieren.

**Nachdem** Sie das File im WWW veröffentlicht haben, sind Sie dafür verantwortlich, den Inhalt laufend zu →aktualisieren – oder das File, wenn es nicht mehr relevant ist, zu →löschen.

### 1.9.3 Bekanntmachen der HTML-Files

Die schönste WWW-Information nützt nichts, wenn niemand erfährt, dass und wo es sie gibt. Nachdem Sie Ihre WWW-Informationen erfolgreich →verfügbar gemacht haben, sollten Sie ihre Existenz und ihren →URL in geeigneter Weise bekanntmachen.

Im einfachsten Fall lassen Sie ein Hypertext-Link auf Ihre WWW-Informationen vom "Webmaster" im EDV-Zentrum Ihrer Institution bzw. Ihres Internet-Providers in dessen WWW-Informationssystem einbauen. Viele Universitäten haben zum Beispiel auf ihrem WWW-Server so etwas wie eine Liste aller Informationen innerhalb der Universität.

Die Universitäten und Firmen selbst lassen Ihren WWW-Server am besten in einer Liste aller Informationssysteme ihres Landes eintragen, und diese Landeslisten sind ihrerseits in weltweiten Listen und Suchhilfen auffindbar (siehe die →Referenzen).

Zusätzlich kann es sinnvoll sein, WWW-Informationen über bestimmte Themen in fachspezifischen Listen und in →Suchhilfen einzutragen. Hinweise, wie man solche Eintragungen vornimmt und welche Informationen man dabei angeben muß, findet man meistens in den betreffenden WWW-Listen und Suchhilfen selbst.

Sehr interessante WWW-Informationen kann man außerdem in einschlägigen →Mailing-Listen oder →Usenet-Newsgruppen bekanntgeben – in weltweiten oder regionalen, allgemeinen oder fachspezifischen – aber nur, wenn die Information für die Mitglieder dieser Gruppen auch wirklich interessant und wichtig ist.

Eine einfache, aber durchaus wirksame Methode ist es, in der "E-Mail-Signature" nicht nur die eigene Mail-Adresse sondern auch den URL der eigenen WWW-Information anzugeben und damit eine nicht als lästig empfundene, aber bei allen fachlichen Kontakten sichtbare "Reklame" für diesen URL zu machen.

Natürlich sollte man in allen diesen Fällen nicht nur den URL sondern auch ein, zwei Stichwörter über den Inhalt der damit erreichbaren Informationen angeben. Dabei sind sachliche Hinweise wie z.B. "Spielpläne für Opernfreunde in Europa" erfolgreicher als ein nichtssagendes "Viele interessante Informationen" oder ein marktschreierisches "Lauter laute Links".

### 1.9.4 Aktualisieren der Informationen

Bitte, bedenken Sie:

Falsche oder nicht mehr aktuelle Informationen sind schlechter als gar keine Information.

Sie als Autor sind dafür verantwortlich, alle von Ihnen im WWW angelegten und damit weltweit veröffentlichten HTML-Files laufend aktuell zu halten und bei Bedarf zu korrigieren – oder das File, wenn es nicht mehr relevant ist oder Sie es nicht mehr weiter aktualisieren möchten, zu →löschen.

Es kann hilfreich sein, die Leser um Mithilfe zu bitten, etwa wie im folgenden Beispiel:

Verantwortlich für den Inhalt dieser Web-Page ist →Dr. Hubert Partl im zentralen Informatikdienst der →BOKU Wien. Bitte, senden Sie Fragen, Hinweise (z.B. auf nicht mehr aktuelle Informationen, fehlende Informationen oder ungünstige WWW-Links) und Verbesserungsvorschläge (Korrekturen, Ergänzungen, Hinzunahme von interessanten Links) per E-Mail an →partl@mail.boku.ac.at

### 1.9.5 Löschen von HTML-Files

Wenn ein im WWW veröffentlichtes File nicht mehr gültig ist oder nicht mehr aktualisiert wird, muß es am →WWW-Server gelöscht werden.

Das Löschen selbst kann einfach mit dem entsprechenden Systembefehl erfolgen (unter Unix z.B. mit rm). **Vorher** müssen aber alle →Hypertext-Links, die noch auf dieses File verweisen, gelöscht bzw. korrigiert werden.

Da Sie nicht wissen können, wo überall im World Wide Web andere Leute Links auf Ihr altes File in deren HTML-Files oder in →Bookmarks oder in →Suchhilfen angelegt haben, ist es empfehlenswert, das File nicht sofort zu löschen sondern während einer längeren Übergangszeit durch ein kleines HTML-File zu ersetzen, das angibt, warum die Information nicht mehr gültig ist, und ein →Hypertext-Link auf die neue, bessere Information enthält, die das alte HTML-File ersetzt. Eventuell lässt sich das auch mit einem "Redirect" in der Konfiguration des →Web-Servers, wo das alte HTML-File früher gespeichert war, lösen.

---

## 2. Text-Elemente

---

- →Vorwort
- →Aufbau des HTML-Files <head> <title> <body>
- →Absätze <p> und Zeilenumbruch
- →Zeilenwechsel <br>
- →Seitenwechsel
- →Buchstaben und Sonderzeichen
- →hervorgehobene Wörter <em> <strong>
- →hervorgehobene Absätze <blockquote>
- →Überschriften <h1> <h2> <h3>
- →Listen und Aufzählungen
  - →nicht numerierte Listen <ul>
  - →numerierte Listen <ol>
  - →Beschreibungen <dl>
- →formatierte Texteingabe <pre>
- →Tabellen <table>
- →Mathematik und Chemie <sub> <sup>
- Spezialeffekte:
  - →Klassen und Style-Sheets
  - →Inhaltsverzeichnisse
  - →Anordnung (linksbündig, rechtsbündig, zentriert)
  - →Abstände
  - →Trennlinien <hr>
  - →Schriftarten und Schriftgrößen
  - →Farben

### 2.1 Aufbau des HTML-Files <head> <title> <body>

- siehe →Grundlagen

## 2.2 Absatz (paragraph) <p> und Zeilenumbruch

Im einfachsten Fall besteht ein HTML-File zwischen  $\rightarrow\langle\text{body}\rangle$  und  $\rightarrow\langle/\text{body}\rangle$  nur aus laufendem Text, mit  $\rightarrow$ Entities für Umlaute und Sonderzeichen und mit eingestreuten Befehlen  $\langle\text{p}\rangle$  für den Beginn eines jeden Absatzes.

### Beispiel:

– – – *Die Eingabe von*

```
<p>
Das ist ein Absatz.
Der Zeilenumbruch erfolgt automatisch.
Zeilenumbruch,
einfache Leerstellen und      mehrfache      Leerstellen
bewirken alle die gleichen Wortabstände.
<p>
Das ist ein neuer Absatz...
```

– – – *bewirkt eine Darstellung wie*

Das ist ein Absatz. Der Zeilenumbruch erfolgt automatisch. Zeilenumbruch, einfache Leerstellen und mehrfache Leerstellen bewirken alle die gleichen Wortabstände.

Das ist ein neuer Absatz...

– – –

*In  $\rightarrow\text{XHTML}$  muss man  $\langle\text{p}\rangle$  immer mit  $\langle/\text{p}\rangle$  beenden.*

Die meisten  $\rightarrow$ Web-Browser stellen Absätze durch eine Leerzeile dar, seltener durch Einrücken der ersten Zeile wie im Buchdruck. Dies lässt sich aber mit  $\rightarrow$ Style-Sheets ändern. Manche Web-Browser erzeugen bei mehrfachen  $\langle\text{p}\rangle$ -Befehlen mehrfache Leerzeilen, die meisten ignorieren jedoch leere Absätze und stellen alle Absätze mit einheitlichen Abständen dar.

Mit  $\rightarrow$ Align-Parametern kann angegeben werden, wie der Absatz dargestellt werden soll (rechtsbündig, linksbündig, zentriert). Absätze *ohne* automatischen Zeilenumbruch kann man mit  $\rightarrow\langle\text{pre}\rangle$  erhalten.

Ein  $\langle/\text{p}\rangle$ -Befehl zum Beenden von Absätzen ist *nicht* nötig, jeder neue Absatz und jede neue  $\rightarrow$ Überschrift oder  $\rightarrow$ Liste und dergleichen beendet automatisch den vorherigen Absatz.

### 2.3 Zeilenwechsel (line break) <br>

Der Zeilenumbruch erfolgt im allgemeinen automatisch so, wie es der Fenstergröße des Benutzers auf seinem →Client-Bildschirm am besten entspricht.

Zusätzliche Zeilenwechsel innerhalb von →Absätzen kann man mit <br> erreichen.

#### Beispiel:

– – – *Die Eingabe von*

```
<p>
H&auml;nschen klein
<br>
ging allein
<br>
in das World Wide Web hinein.
<p>
```

– – – *bewirkt eine Darstellung wie*

```
H&auml;nschen klein
ging allein
in das World Wide Web hinein.
```

– – –

*In →XHTML muss man <br /> statt <br> schreiben.*

Manche →Web-Browser erzeugen bei mehrfachen <br>-Befehlen mehrfache Leerzeilen, die meisten ignorieren jedoch leere Zeilen und stellen alle Absätze mit einheitlichen Zeilenabständen dar.

### 2.4 Seitenwechsel (page break)

Im Gegensatz zur gedruckten Ausgabe, wo alle Informationen auf Papierblätter einer bestimmten Größe aufgeteilt werden müssen, erfolgt die Ausgabe der WWW-Informationen auf den Bildschirmen der Benutzer ohne solche Seitengrenzen. Der Benutzer kann mit den Funktionstasten oder dem Scrollbar seines →Web-Browsers fortlaufend und beliebig weit nach oben und unten lesen.

Es gibt deshalb in HTML *keinen* Befehl für einen Seitenwechsel. Neue Seiten (falls das HTML-File ausgedruckt wird) bzw. neue Fenster am Bildschirm des Benutzers (neue "Web-Pages") erreicht man nur durch den →Sprung auf ein neues HTML-File.

Innerhalb von HTML-Files ("Web-Pages") kann man für Trennlinien, die am Bildschirm eine ähnlich trennende Funktion wie das Umblättern auf eine neue Papierseite erfüllen, den Befehl →<hr> bzw. die Befehlsfolge </p><hr><p> verwenden.

In →Style-Sheets ist eine Möglichkeit vorgesehen, Stellen zu markieren, die beim Ausdrucken möglichst eine neue Seite beginnen sollen, bzw. umgekehrt Bereiche zu markieren, die möglichst nicht durch Seitenwechsel getrennt werden sollen. Diese Möglichkeiten werden aber von den meisten Web-Browsern noch nicht unterstützt.

## 2.5 Buchstaben und Sonderzeichen (entity)

Die Zeichen `<` und `>` und `&` haben eine Sonderbedeutung in HTML-Files. Wenn Sie die entsprechenden Zeichen im Text darstellen wollen, müssen Sie dafür eigene HTML-Befehle eingeben, die sogenannten "Entities" (Einheiten):

`&lt;`; für das Kleiner-Zeichen `<`

`&gt;`; für das Größer-Zeichen `>`

`&amp;`; für das Und-Zeichen `&`

Auch für nicht-amerikanische Buchstaben wie z.B. deutsche Umlaute und scharfes s, französische Akzente usw. müssen Sie solche "Entities" (oder die genormten ISO-8859-1-Codes) angeben, damit sie auf den verschiedenen →Web-Browsern so gut wie möglich dargestellt werden können. Andere Kodierungen, die nur auf bestimmten Rechnertypen gelten (z.B. nur auf PCs unter MS-DOS oder nur auf Apple Macintosh), dürfen dafür nicht verwendet werden.

Für deutschsprachige Texte sind die folgenden Entities wichtig:

`&auml;` für ä (Umlaut-a)

`&Auml;` für Ä (Umlaut-A)

`&ouml;` für ö (Umlaut-o)

`&Ouml;` für Ö (Umlaut-O)

`&uuml;` für ü (Umlaut-u)

`&Uuml;` für Ü (Umlaut-U)

`&szlig;` für ß (scharfes s, s-z-Ligatur)

Für andere europäische Sprachen gibt es Entities wie z.B.

`&eacute;` für é (e mit Acute-Akzent)

`&agrave;` für à (a mit Grave-Akzent)

`&ocirc;` für ô (o mit Circumflex-Akzent)

`&ccedil;` für ç (c mit Cedille)

`&ntilde;` für ñ (n mit Tilde)

`&aring;` für å (a mit Ring)

und weitere Entities mit analog gebildeten Namen.

Manche Web-Browser unterstützen auch weitere Sonderzeichen und Spezialfunktionen wie z.B.

`&deg;` für das Grad-Symbol ° (degree)

`&copy;` für das Copyright-Symbol ©

`&sect;` für das deutsche Paragraphen-Zeichen § (section)

`&para;` für das amerikanische Absatz-Zeichen ¶ (paragraph)

`&nbsp;` für eine Leerstelle, bei der kein Zeilenwechsel erfolgen darf (non breaking space)

`&shy;` für eine Stelle, an der ein Wort bei Bedarf abgeteilt werden darf (soft hyphen)

Es gibt auch Web-Browser die zwar nicht diese Entity-Namen für die Sonderzeichen unterstützen, aber die Angabe des ISO-Codes in der Form `&#123;` erlauben, wobei statt 123 der ISO-Code des gewünschten Zeichens anzugeben ist (dezimal).

Für vollständige Listen der Entities und ISO-Codes wird auf die →Referenzen verwiesen.

Der Strichpunkt am Ende der Entities darf *nicht* weggelassen werden, auch wenn manche (aber nicht alle) Web-Browser vielleicht in manchen (aber nicht allen) Fällen einen fehlenden Strichpunkt "erraten" können.

**Beispiel:**

– – – *Die Eingabe von*

ein Caf&eacute;; in der Sch&ouml;nbrunner Stra&szlig;e

– – – *bewirkt eine Darstellung wie*

ein Café in der Schönbrunner Straße

– – –

In →HTML 4 sind wesentliche Erweiterungen des Zeichensatzes vorgesehen:

- Umstellung vom westeuropäischen Zeichensatz Iso-Latin-1 (ISO-8559-1) auf den alle Schriften der Welt umfassenden Zeichensatz "Unicode" (UTF-8),
- Angabe der Sprache (mit `lang=`) und der Schreibrichtung (von links nach rechts oder von rechts nach links, mit `dir=`),
- `<q>` und `</q>` für richtige Anführungszeichen (je nach der Sprache),
- zahlreiche zusätzliche Entities, z.B. für verschiedene Arten von Anführungszeichen, Gedankenstriche, größere oder kleinere Abstände, mathematische Symbole, griechische Buchstaben, Pfeile, Spielkartensymbole und viele andere.

Diese Erweiterungen werden aber von den meisten Web-Browsern noch nicht unterstützt.

**Beispiel:**

– – – *Die Eingabe von*

mit &hearts;lichen Gr&uuml;&szlig;en

– – – *bewirkt eine Darstellung wie*

mit ♥lichen Grüßen

– – –

## 2.6 Hervorgehobene Wörter (emphasis) <em> <strong>

Zwischen <em> und </em> stehender Text wird *hervorgehoben* (emphasis = Betonung).

Zwischen <strong> und </strong> stehender Text wird **stärker** hervorgehoben (strong = stark).

<em> eignet sich für die Betonung einzelner Wörter *innerhalb von Sätzen*. Viele →Web-Browser verwenden dafür *kursive Schrift*, andere verwenden eine andere Farbe, Helligkeit oder (bei Sprachausgabe) Tonhöhe.

<strong> eignet sich für Texte, die wie →Überschriften **ins Auge springen** sollen. Viele →Web-Browser verwenden dafür **fette Schrift**, andere verwenden Farbe, Helligkeit oder Lautstärke.

### Beispiel:

– – – *Die Eingabe von*

In der <strong>Hypertext Markup Language</strong>  
wird die <em>logische</em> Bedeutung der Textelemente festgelegt,  
<em>nicht</em> das Aussehen.

– – – *bewirkt eine Darstellung wie*

In der **Hypertext Markup Language** wird die *logische* Bedeutung der Textelemente festgelegt, *nicht* das Aussehen.

– – –

## 2.7 Hervorgehobene Absätze und Zitate (quote) <blockquote>

Zwischen <blockquote> und </blockquote> stehende Absätze werden in einer speziellen Form dargestellt, die sich vor allem für Zitate (Quotes) eignet, aber auch für andere Texte verwendet werden kann, die "besonders" aussehen sollen, wie z.B. Beispiele oder Gedichte.

Viele (aber nicht alle) →Web-Browser verwenden dafür eingerückte Absätze, manche auch kursive Schrift oder eine andere Schriftart, wieder andere eine Randmarkierung oder dergleichen. Ein Beispiel dafür finden Sie in der Geschichte über das →Auto im Dschungel.

Durch die Angabe von →Klassen und durch die Verwendung von →Style-Sheets kann genauer spezifiziert werden, um was für Arten von Absätzen es sich handelt und wie sie am Bildschirm dargestellt werden sollen.

## 2.8 Überschriften (heading) <h1> <h2> <h3>

Zwischen <hx> und </hx> kann man Überschriften der Ebene **x** (1 bis 6) angeben. Der dazwischen stehende Text kann auch →Bilder oder →Links enthalten.

Es wird empfohlen, die Hierarchie der Überschriften genau einzuhalten und nicht mehr als 3 Ebenen zu verwenden:

<h1> für die Haupt-Überschriften (Kapitel),

<h2> für Abschnitte innerhalb der Kapitel,

<h3> für Unter-Abschnitte innerhalb der Abschnitte.

Neue Kapitel oder Abschnitte bedeuten automatisch neue Absätze, die Überschriften dürfen deshalb nicht innerhalb von →Absätzen oder →Listen und dergleichen stehen.

Die meisten →Web-Browser stellen die Überschriften durch fette und größere Schrift dar. Manche Web-Browser rücken den nachfolgenden Text entsprechend der Hierarchie-Ebene ein.

### Beispiele:

– – – *Die Eingabe von*

`<h2>Beispiel fuuml;r eine Uuml;berschrift der Ebene 2;/h2; Text auf Ebene 2...`

`<h3>Beispiel fuuml;r eine Uuml;berschrift der Ebene 3;/h3; Text auf Ebene 3...`

– – – *bewirkt eine Darstellung wie*

## 2.9 Listen und Aufzählungen

- →nicht numerierte Listen <ul>
- →numerierte Listen <ol>
- →Beschreibungen <dl>
- siehe auch →spezielle Numerierungen

### 2.9.1 Nicht numerierte Liste (unordered list) <ul>

Mit <ul> wird eine Liste oder Aufzählung begonnen.

Jedes Listenelement innerhalb der Liste beginnt mit <li> (list item).

Mit </ul> wird die Liste beendet.

Die meisten →Web-Browser stellen die Listenelemente durch eingerückte Absätze mit einem vorangestellten dicken schwarzen Punkt oder Sternchen oder anderen Symbolen dar. Bei manchen neueren Web-Browsern kann das Layout der Liste durch die Angabe von Parametern wie `type=circle` oder `src="xxx.gif"` in den Befehlen <ul> und <li> beeinflusst werden.

### Beispiel:

– – – *Die Eingabe von*

Eine unsortierte Liste:

```
<ul>
<li>Unsortierte Listen eignen sich für alle Arten
von Aufzählungen.
<li>Listen können auch geschachtelt werden:
<ul>
<li>Elemente der inneren Liste
werden meist weiter eingerückt und/oder
mit anderen Symbolen versehen.
<li>Hier ist das zweite Element der inneren Liste.
</ul>
<li>Hier ist ein weiteres (letztes) Element
der äußeren Liste.
</ul>
```

– – – bewirkt eine Darstellung wie

Eine unsortierte Liste:

- Unsortierte Listen eignen sich für alle Arten von Aufzählungen.
- Listen können auch geschachtelt werden:
  - Elemente der inneren Liste werden meist weiter eingerückt und/oder mit anderen Symbolen versehen.
  - Hier ist das zweite Element der inneren Liste.
- Hier ist ein weiteres (letztes) Element der äußeren Liste.

– – –

*In →XHTML muss man <li> immer mit </li> beenden.*

## 2.9.2 Numerierte Liste (ordered list) <ol>

Mit <ol> wird eine Liste oder Aufzählung begonnen.

Jedes Listenelement innerhalb der Liste beginnt mit <li> (list item).

Mit </ol> wird die Liste beendet.

Die meisten →Web-Browser stellen die Listenelemente durch eingerückte Absätze mit einer vorangestellten fortlaufenden Nummer dar. Die Numerierung erfolgt automatisch. Bei neueren Web-Browsern kann man die Numerierung durch →Parameter wie `type`, `start` und `value` in den Befehlen <ol> und <ol> beeinflussen (siehe →Spezialeffekte); diese Angaben werden aber von älteren Web-Browsern ignoriert.

### Beispiel:

– – – *Die Eingabe von*

Eine numerierte Liste:

```
<ol>
<li>Numerierte Listen eignen sich für Aufzählungen,
bei denen die genaue Reihenfolge der Elemente wichtig ist.
```

```

<li>Listen können auch geschachtelt werden:
<ol type=a>
<li>Elemente der inneren Liste
werden meist weiter eingerückt und
separat nummeriert.
<li>Hier ist das zweite Element der inneren Liste.
</ol>
<li>Hier ist ein weiteres (letztes) Element
der äußeren Liste.
</ol>

```

— — — bewirkt eine Darstellung wie

Eine nummerierte Liste:

1. Nummerierte Listen eignen sich für Aufzählungen, bei denen die genaue Reihenfolge der Elemente wichtig ist.
2. Listen können auch geschachtelt werden:
  - (a) Elemente der inneren Liste werden meist weiter eingerückt und separat nummeriert.
  - (b) Hier ist das zweite Element der inneren Liste.
3. Hier ist ein weiteres (letztes) Element der äußeren Liste.

— — —

In →XHTML muss man `<li>` immer mit `</li>` beenden.

### 2.9.3 Liste von Beschreibungen (definition list) `<dl>`

Mit `<dl>` wird eine Liste von Beschreibungen begonnen.

Jedes Listenelement innerhalb der Liste beginnt mit `<dt>` (definition term). Damit wird der Begriff angegeben, der beschrieben werden soll.

Anschließend wird mit `<dd>` (definition description) der Text angegeben, mit dem der Begriff beschrieben wird.

Mit `</dl>` wird die Liste beendet.

Viele (aber nicht alle) →Web-Browser stellen die Begriffe durch neue Absätze und ihre Beschreibungen durch eingerückte Absätze dar. Bei neueren Web-Browsern kann durch die Angabe von `<dl compact>` ein kompakteres Format erreicht werden.

#### Beispiel:

— — — Die Eingabe von

```

Kleine Tierkunde
<dl>
<dt>Gelse:
<dd>ein kleines Tier,

```

```

das an Badeseen die Menschen verjagt.
Die Gelse
wird auf Grund ihrer Wirkungsweise auch als
Stechmücke bezeichnet.
<dt>Gemse:
<dd>ein großes Tier,
das in den Alpen von Menschen gejagt wird.
Die Gemse
wird auf Grund ihrer Losung manchmal fälschlich als
eierlegend bezeichnet.
</dl>

```

– – – bewirkt eine Darstellung wie

Kleine Tierkunde

- Gelse: ein kleines Tier, das an Badeseen die Menschen verjagt. Die Gelse wird auf Grund ihrer Wirkungsweise auch als Stechmücke bezeichnet.
- Gemse: ein großes Tier, das in den Alpen von Menschen gejagt wird. Die Gemse wird auf Grund ihrer Losung manchmal fälschlich als eierlegend bezeichnet.

– – –

In →XHTML muss man <dt> und <dd> immer mit </dt> bzw. </dd> beenden.

## 2.10 Formatierte Texteingabe (preformatted) <pre>

Im Gegensatz zum automatischen Zeilenumbruch von normalen →Absätzen wird zwischen <pre> und </pre> stehender Text so ausgegeben, wie er eingegeben wird, also mit allen Leerstellen und Zeilenwechsell. Dabei wird eine nicht-proportionale Schrift verwendet, bei der alle Buchstaben und Leerstellen die gleiche Breite haben. Dadurch kann man hier Leerstellen zum Einrücken oder zum spaltenweisen Ausrichten verwenden, was bei normalen Absätzen und bei Proportionalschriften nicht möglich wäre.

### Beispiel:

– – – Die Eingabe von

```

<pre>
11111000000  bin&auml;r
          3700  oktal
           7C0  hexadezimal
          1984  dezimal
</pre>

```

– – – bewirkt eine Darstellung wie

```

11111000000  binaer
          3700  oktal
           7C0  hexadezimal
          1984  dezimal

```

- - -

Zwischen `<pre>` und `</pre>` können auch gewisse HTML-Befehle wie z.B. `→Links` und `→Entities` verwendet werden, und die Sonderzeichen `<` und `>` und `&` *müssen* als `→Entities` `&lt;`; und `&gt;`; bzw. `&amp;` geschrieben werden.

Für Einrückungen und Abstände sollten Sie keine Tabulator-Zeichen verwenden, da diese von verschiedenen Web-Browsern verschieden interpretiert werden, sondern stets die richtige Anzahl von Leerstellen.

Vor `<pre>` und nach `</pre>` wird jeweils ein neuer Absatz oder eine neue Zeile begonnen.

## 2.11 Tabelle (table) `<table>`

Seit `→HTML 3.2` gibt es HTML-Befehle für Tabellen:

Die gesamte Tabelle beginnt mit `<table>` und endet mit `</table>`.

Innerhalb der Tabelle muss jede Tabellenzeile, auch die erste, mit `<tr>` (table row) beginnen.

Innerhalb der Tabellenzeile muss jedes Feld (Spaltenelement), auch das erste, mit `<td>` (table cell for data) oder `<th>` (table cell for header) beginnen. Mit `<td>` gibt man die normalen Datenfelder an; sie werden standardmäßig linksbündig in ihrer Spalte dargestellt. Mit `<th>` kann man die Spalten- und Zeilen-Bezeichnungen angeben, sie werden standardmäßig zentriert in ihrer Spalte dargestellt.

Befehle der Form `</td>`, `</th>` und `</tr>` zum Beenden von Feldern und Tabellenzeilen sind *nicht* nötig, jedes neue Feld und jede neue Zeile beendet automatisch das vorherige Feld.

Wenn man `<table border>` angibt, wird die Tabelle mit einem Rahmen und Trennlinien versehen. Bei `<table border=0>` werden keine Rahmen und Trennlinien verwendet. In den Befehlen `<table>`, `<tr>`, `<td>` und `<th>` kann man weitere Parameter angeben, unter anderem z.B. für die `→Ausrichtung`:

`align=left, right` oder `center`  
oder `char=","` für die Ausrichtung am Dezimalkomma,  
und `valign=top, bottom` oder `middle`

### Beispiel:

- - - *Die Eingabe von*

```
<table border>
<tr><td align=right>1111100000
      <td align=left>bin&auml;r
<tr><td align=right>370
      <td align=left>oktal
<tr><td align=right>7C0
      <td align=left>hexadezimal
<tr><td align=right>1984
      <td align=left>dezimal
</table>
```

--- bewirkt eine Darstellung wie

11111000000	binär
370	oktal
7C0	hexadezimal
1984	dezimal

---

In →XHTML muss man `<tr>` `<td>` `<th>` jeweils mit `</tr>` `</td>` `</th>` beenden.

Bei alten →Web-Browsern, die die Tabellen-Befehle noch nicht unterstützen, werden diese Befehle ignoriert und der Tabelleninhalt wird einfach als fortlaufender Text dargestellt, also z.B. wie

11111000000 binär 370 oktal 7C0 hexadezimal 1984 dezimal

---

In →HTML 4 sind zahlreiche weitergehende Möglichkeiten für die Gestaltung und Strukturierung von Tabellen vorgesehen. Unter anderem kann man die Tabelle mit `<thead>` `<tbody>` und `<tfoot>` in einen Kopfteil, Hauptteil und Fußteil aufteilen. Wenn ein (Bildschirm-) Seitenwechsel innerhalb der Tabelle erfolgt, werden dann die Kopf- und Fußzeilen so wiederholt, dass sie immer sichtbar sind. Bei älteren Web-Browsern wird diese Unterscheidung ignoriert und der gesamte Tabelleninhalt (einschließlich Kopf- und Fußteil) wird einfach ohne Wiederholungen dargestellt.

### Beispiel:

--- Die Eingabe von

```
<table frame=hsides rules=groups cellspacing=12 >
<thead>
  <tr> <td align=right>Wiener Staatsoper
      <td align=left>25. 10. 1970
  <tr> ...
</thead>
<tbody>
  <tr> <td align=right>Philipp II.
      <td align=left>Nicolai Ghiaurov
  <tr> ...
</tbody>
<tfoot>
  <tr> <td align=right>weitere Vorstellungen:
      <td align=left>28.10., 26.11.1970
</tfoot>
</table>
```

--- bewirkt eine Darstellung wie

Wiener Staatsoper	25. 10. 1970
Giuseppe Verdi	<b>Don Carlos</b>
<i>Rolle:</i>	<i>Besetzung:</i>
Philipp II.	Nicolai Ghiaurov
Don Carlos	Franco Corelli
Posa	Eberhard Wächter
Großinquisitor	Martti Talvela
Mönch	Tugomir Franc
Elisabeth	Gundula Janowitz
Eboli	Shirley Verrett
Tebaldo	Edita Gruberova
Gräfin Aremberg	Christa Reichert
Graf Lerma	Ewald Aichberger
Stimme vom Himmel	Judith Blegen
weitere Vorstellungen: 28.10., 26.11.1970	

– – – *Seitenwechsel*

Wiener Staatsoper	25. 10. 1970
Giuseppe Verdi	<b>Don Carlos</b>
<i>Rolle:</i>	<i>Besetzung:</i>
Dirigent	Horst Stein
Chorleiter	Norbert Balatsch
Bühnenmusik	Ralf Hossfeld
Bühnenbild und Kostüme	Jürgen Rose
Regie	Otto Schenk
weitere Vorstellungen: 28.10., 26.11.1970	

– – – *und je nach der verwendeten Browser-Version werden die Kopf- und Fußzeilen beim Vor- und Zurückblättern wiederholt oder nur einfach angezeigt.*

– – –

Man kann Tabellen unter Umständen auch für →Layout-Effekte wie →spezielle Anordnungen oder →Einrückungen einsetzen und für diesen Zweck auch weitere Parameter wie z.B. →**width** verwenden.

Dabei müssen aber die im Kapitel →Layout und Spezialeffekte und insbesondere auch für →**width** angeführten Hinweise genau beachtet werden, damit die Informationen trotz dieser "Tricks" für *alle* Leser sinnvoll lesbar bleiben.

## 2.12 Mathematik und Chemie <sub></sub> <sup></sup>

→HTML enthält keine Befehle für die Darstellung von komplexen mathematischen Formeln mit Brüchen, Wurzeln, Integralen, griechischen Buchstaben und dergleichen oder für die Darstellung von chemischen Formeln. Die dafür vorgesehen XML-Anwendungen →MathML und →CML werden derzeit von den meisten →Web-Browsern noch nicht unterstützt. Solche Formeln kann man daher bis auf weiteres nur als →Bilder einfügen.

*Einfache* mathematische und chemische Formeln können aber mit reinem HTML dargestellt werden. Dazu gibt es die normalen Sonderzeichen wie + und = sowie die folgenden Befehle für das Hoch- und Tiefstellen:

Zwischen <sup> und </sup> stehender Text wird hochgestellt (superscript).

Zwischen <sub> und </sub> stehender Text wird tiefgestellt (subscript).

Diese Befehle werden nicht von allen →Web-Browsern unterstützt, das Ergebnis ist aber in vielen Fällen auch bei den Web-Browsern brauchbar, die diese Befehle ignorieren.

### Beispiele:

– – – *Die Eingabe von*

```
<p align=center>  
a<sup>2</sup> + b<sup>2</sup> = c<sup>2</sup>
```

– – – *bewirkt eine Darstellung wie*

$$a^2 + b^2 = c^2$$

– – – *Die Eingabe von*

```
<p align=center>  
<b>X</b> . <b>Y</b> = x<sub>1</sub> y<sub>1</sub> +  
x<sub>2</sub> y<sub>2</sub> + x<sub>3</sub> y<sub>3</sub>
```

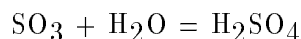
– – – *bewirkt eine Darstellung wie*

$$\mathbf{X} \cdot \mathbf{Y} = x_1 y_1 + x_2 y_2 + x_3 y_3$$

– – – *Die Eingabe von*

```
<p align=center>  
S<sub>3</sub> + H<sub>2</sub>O = H<sub>2</sub>S<sub>4</sub>
```

– – – *bewirkt eine Darstellung wie*



– – –

---

### 3. Hypertext-Links

---

- →Vorwort
- →Verweise zu anderen Informationen <a href>
- →URL (Uniform Resource Locator)
  - →absolute URLs im WWW
  - →relative URLs im WWW
  - →URLs für andere Internet-Services
- →Listen von Verweisen
- →Markierungen innerhalb eines HTML-Files <a name>
- →Inhaltsverzeichnisse
- Spezialeffekte:
  - →Formulare <form>
  - →Image-Maps <form> usemap
  - →Electronic Mail (mailto)
  - →Navigationshilfen <link>
  - →Target-Angaben bei Frames

### 3.1 Verweise zu anderen Informationen <a href>

Vielleicht die wichtigste Eigenschaft des →World Wide Web und der →Hypertext Markup Language ist die Möglichkeit, dass der Leser mit Hilfe von →Hypertext-Verbindungen ("Links") von einer Information zu einer anderen Information springen kann.

Zu diesem Zweck gibt man im HTML-Text einen "Anker" (anchor) der folgenden Form an:

```
<a href="URL">Link-Text</a>
```

Der zwischen <a> und </a> stehende Link-Text ist der am Bildschirm sichtbare Text, der vom Benutzer "ausgewählt" werden kann (Klick mit der Maus oder Drücken einer bestimmten Taste wie Return oder Rechtspfeil, je nach dem verwendeten →Web-Browser). Viele Web-Browser kennzeichnen diesen Link-Text durch Unterstreichen, manche durch eine spezielle Farbe, oft durch Unterstreichen *und* Farbe, wobei oft auch noch verschiedene Farben oder Unterstreichungsarten für bereits aufgesuchte und noch nicht aufgesuchte Links verwendet werden.

Innerhalb des Befehls <a> wird mit href= (Hypertext Reference) angegeben, zu welcher Information der Sprung bei Auswahl dieses Hypertext-Links erfolgen soll. Diese Adresse wird in der Form eines →URL (Uniform Resource Locator) angegeben.

Nachdem der Benutzer die Information, zu der er auf diese Weise verzweigt hat, gelesen hat, kann er mit Hilfe des Zurück-Befehls ("back") seines →Web-Browsers wieder an die Stelle zurückkehren, von der der Sprung ausgegangen ist.

Bei der Formulierung des Textes und der Platzierung der Links sollten Sie darauf achten, dass der dem Leser durch die Unterstreichung und Farbe in die Augen springende Link-Text die wichtigen Stichworte enthält und nicht ein unwichtiges Wort wie "hier" oder gar ein nicht für alle Web-Browser gültiges "click here".

Ein **Beispiel** für eine schlechte Formulierung:

– – – *Die Eingabe von*

```
F&uuml;r ausf&uuml;hrlichere Informationen &uuml;ber  
absolute und relative URLs klicken Sie  
<a href="hein3.html#url">hier</a>.
```

– – – *bewirkt eine Darstellung wie*

Für ausführlichere Informationen über absolute und relative URLs klicken Sie →hier.

– – –

Beispiele für bessere Formulierungen:

Hier können Sie ausführlichere Informationen über →absolute und relative URLs auswählen.

oder:

Als Adresse sind →absolute oder relative URLs anzugeben.

– – –

Verweise auf die →Startseite Ihrer Firma oder Universität sollten Sie nicht als →Home-Page sondern mit den Namen Ihrer Firma bzw. Universität bezeichnen. Für den Leser klingt "Geh nach Hause!" (go to home page, go home) wie das unfreundliche Wegschicken zu seinem eigenen Zuhause und nicht wie eine freundliche Einladung zum Besuch der Informationen des Autors.

Verweise sollten niemals das Wort "zurück" enthalten, denn Sie als Autor können ja nicht wissen, von wo der Leser auf Ihre Information gekommen ist – von Ihrer Startseite, von einem Verweis in irgendeiner anderen Web-Page, aus seinen eigenen →Boommars oder aus einer →Suchhilfe.

## 3.2 URL (Uniform Resource Locator)

Der URL ist eine Art Adresse, die alle Angaben enthält, die für das Auffinden einer bestimmten Information im World Wide Web notwendig sind. URLs haben grundsätzlich das Format

`methode:objekt`

Ein typischer URL hat etwa den folgenden Aufbau:

`protokoll://hostname/directoryname/filename#marke`

### Beispiel:

`http://www.boku.ac.at/htmlreif/hein3.html?url`

bedeutet einen Sprung an die Stelle "url" im File "hein3.html" im Directory "htmlreif" auf dem WWW-Server "www.boku.ac.at".

- Das **Protokoll** gibt die Art des Zugriffs und damit die Art der Information an. Für Informationen auf →WWW-Servern ist →"http" anzugeben, für andere Informationssysteme wie Gopher oder FTP entsprechende →andere Angaben.
- Der **Hostname** gibt die →Internet-Adresse des →Servers an, auf dem die Information gespeichert ist (voller Hostname mit "Domain"). In Spezialfällen können beim Hostnamen noch weitere Angaben in der Form `username:paßwort@hostname:portnummer` notwendig sein: ein Username (mit oder ohne Paßwort) für den Zugriff, oder die Portnummer, unter der das Server-Programm auf diesem Rechner läuft.
- Der **Directoryname** und der **Filename** geben an, welches File vom →Server auf den →Client-Rechner übertragen werden soll.
- Mit einer **Marke** kann angegeben werden, welche Stelle innerhalb des Files vom →Web-Browser am Bildschirm angezeigt werden soll.

Falls Teile des URL (z.B. der Filename) Leerstellen oder Sonderzeichen enthalten, müssen diese durch einen Hexadezimal-Code der Form %XX ersetzt werden, z.B.

**%20** für eine Leerstelle,

**%2F** für einen Schrägstrich,

**%3F** für ein Fragezeichen.

Es müssen nicht immer alle Teile des URL angegeben werden. Dementsprechend wird zwischen →absoluten und →relativen URLs unterschieden:

- Für Hypertext-Links zwischen Informationen, die gemeinsam erstellt, abgespeichert und gewartet werden, sollten Sie stets →relative URLs verwenden. Damit werden Probleme bei einer eventuellen Übertragung dieser Files in ein anderes Directory oder auf einen anderen WWW-Server oder auf einen Client-Rechner vermieden.
- Für Hypertext-Links auf Informationen, die unabhängig von Ihren HTML-Files gewartet werden oder in einem anderen Directory oder auf einem anderen Server liegen, sollten Sie stets →absolute URLs verwenden.

Achtung! Im Gegensatz zu den HTML-Befehlswörtern, bei denen die Groß- oder Kleinschreibung egal ist, dürfen innerhalb der URLs Groß- und Kleinbuchstaben nicht verwechselt werden.

### 3.2.1 Absolute URLs im WWW

Unter absoluten URLs werden Adressen verstanden, die unabhängig vom Kontext verstanden werden, d.h. egal in welchem HTML-File sie stehen, der Zugriff erfolgt immer auf dieselbe Information.

Beispiele:

`http://hostname/`

bedeutet die WWW-Start-Page auf dem Rechner "hostname".

`http://hostname:8000/`

bedeutet die WWW-Start-Page eines WWW-Servers, der auf dem Rechner "hostname" unter der Portnummer 8000 (statt unter der Standard-Portnummer 80) läuft.

`http://hostname/directory/`

bedeutet eine Übersicht über ein auf dem Rechner "hostname" liegendes Directory oder die in diesem Directory liegende Start-Page (wenn es eine solche gibt).

Der Schrägstrich nach dem Directory-Namen ist notwendig, um ihn von File-Namen zu unterscheiden. Manche Web-Browser können den Unterschied auch bei fehlendem Schrägstrich feststellen (allerdings nur mit doppeltem Zugriff und damit erhöhter Netz-Lastung). In vielen Fällen führt das Fehlen des Schrägstrichs aber zu verschiedenen Fehlern bei weiteren Links. Der Schrägstrich nach dem Directory-Namen muss deshalb *immer* angegeben werden.

`http://hostname/directory/filename`

bedeutet ein auf dem Rechner "hostname" im angegebenen Directory liegendes File.

`http://hostname/directory/filename#marke`

bedeutet den Sprung auf eine bestimmte Stelle innerhalb dieses Files.

`http://hostname/directory/filename?name=wert&name=wert`

bedeutet die Ausführung eines →CGI-Programms mit Parametern.

`file:///directory/filename`

bedeutet ein auf dem lokalen Rechner im angegebenen Directory liegendes File.

Auch im Fall von PCs sind dabei Laufwerk und Directory nicht in der DOS-Schreibweise sondern in der URL-Form

`file:///c:/directory/filename`

anzugeben.

### 3.2.2 Relative URLs im WWW

Relative URLs geben die Adresse relativ zur Lage des HTML-Files an, in dem das Hypertext-Link steht. Dies ist für zusammengehörende Files günstig, die eventuell gemeinsam auf einen anderen Bereich übertragen werden.

Beispiele:

**#marke**

bedeutet eine andere Stelle innerhalb desselben Files.

**filename**

bedeutet ein anderes File innerhalb desselben Directory auf demselben WWW-Server.

**filename#marke**

bedeutet eine bestimmte Stelle in diesem anderen File.

**directory/filename**

bedeutet ein File in einem Subdirectory des aktuellen Directory (relativer Pfad).

**../filename**

bedeutet ein File im Parent-Directory des aktuellen Directory (relativer Pfad).

**/directory/filename**

bedeutet ein File in einem bestimmten Directory (absoluter Pfad) auf demselben WWW-Server.

### 3.2.3 URLs für andere Internet-Services

URLs können nicht nur auf →WWW-Server sondern auch auf →FTP-Server oder andere →Internet-Services verweisen.

Dazu sind im URL entsprechend andere →Protokolle und Parameter anzugeben.

Beispiele:

**ftp://hostname/directory/**

für ein Directory auf einem →anonymen FTP-Server.

**ftp://hostname/directory/filename**

für ein File auf einem →anonymen FTP-Server.

**telnet://userid@hostname**

für eine →Telnet-Verbindung zu einem Internet-Rechner mit Login unter einer bestimmten Userid.

**news:news.group.name**

für eine bestimmte →Usenet-Newsgruppe auf dem lokalen News-Server (NNTP-Server) des →WWW-Client.

**mailto:userid@hostname**

für das Senden von →Electronic Mail an eine bestimmte Mail-Adresse über das lokale Mail-System des →WWW-Client (siehe →Interaktion).

### 3.3 Listen von Verweisen

Für Menüs, in denen der Leser aus mehreren →Hypertext-Links auswählen kann, verwenden Sie am besten übersichtliche →Listen der Form

```
<ul>
<li><a href="URL1">Beschreibung 1</a>
<li><a href="URL2">Beschreibung 2</a>
<li><a href="URL3">Beschreibung 3</a>
</ul>
```

Ein Beispiel für eine solche Auswahl-Liste sehen Sie in den →Referenzen.

### 3.4 Markierungen innerhalb eines HTML-Files <a name>

Für die Markierung von möglichen Sprungzielen innerhalb Ihres HTML-Files können Sie "Anker" (anchor) der folgenden Form angeben:

```
<a name="marke">Text</a>
```

Der zwischen <a> und </a> stehende Text ist das Sprungziel. Er darf nicht leer sein und darf auch HTML-Befehle enthalten (aber keine anderen <a>-Befehle).

Der mit name= definierte Name der Sprungmarke kann dann in →Hypertext-Links in der Form

```
<a href="#marke">Link-Text</a>
```

oder

```
<a href="filename#marke">Link-Text</a>
```

angegeben werden und bewirkt einen Sprung zu der markierten Stelle innerhalb des HTML-Files.

Bitte, beachten Sie, dass das Nummernzeichen # nur bei href= und nicht bei name= anzugeben ist und dass im Gegensatz zu den HTML-Befehlswörtern, bei denen die Groß- oder Kleinschreibung egal ist, in den Sprungmarken Groß- und Kleinbuchstaben unterschieden werden und daher nicht verwechselt werden dürfen.

**Beispiel** (frei nach D. E. Knuth: The TeXbook):

– – – *Die Eingabe von*

```
<dl>
<dt><a name="endlos">endlos:</a>
<dd>siehe
<a href="#schleife">Schleife</a>.
<dt><a name="schleife">Schleife:</a>
<dd>siehe
<a href="#endlos">endlos</a>.
</dl>
```

– – – *bewirkt eine Darstellung wie*

endlos:   siehe →Schleife.

Schleife:   siehe →endlos.

- - -

In →HTML 4 und →XHTML ist vorgesehen, dass man Sprungmarken mit `id="marke"` statt mit `name=` vereinbaren kann, und die Angabe von `id=` ist nicht nur in eigenen `<a>`-Befehlen sondern auch direkt in Befehlen wie `→<hX>`, `→<p>`, `→<ul>` u.dgl. möglich. Beispiele:

```
<h2 id="marke1">&Uuml;berschrift</h2>
<p id="marke2">Text...
```

Dies wird aber derzeit von vielen Browsern noch nicht unterstützt und sollte daher noch nicht eingesetzt oder mit der alten Version kombiniert werden:

```
<h2 id="marke1"><a name="marke1"> ... </a></h2>
```

### 3.5 Inhaltsverzeichnisse

Bei HTML-Files, die länger als zwei, drei Bildschirm-Seiten sind, ist es empfehlenswert, an den Anfang ein Inhaltsverzeichnis mit Sprüngen zu den einzelnen Abschnitten innerhalb des Files zu stellen, etwa in der folgenden Form:

```
<h1>Haupt-Ueberschrift</h1>
<ul>
<li><a href="#kap1">Kapitel-Ueberschrift 1</a>
<li><a href="#kap2">Kapitel-Ueberschrift 2</a>
<li><a href="#kap3">Kapitel-Ueberschrift 3</a>
</ul>
<p>
einleitender Text ...
</p><hr><p>

<h2><a name="kap1">Kapitel-Ueberschrift 1</a></h2>
Text des ersten Kapitels ...
</p><hr><p>

<h2><a name="kap2">Kapitel-Ueberschrift 2</a></h2>
Text des zweiten Kapitels ...
</p><hr><p>

<h2><a name="kap3">Kapitel-Ueberschrift 3</a></h2>
Text des dritten Kapitels ...
</p><hr><p>
```

Beispiele dafür sehen Sie in allen Kapiteln dieser HTML-Einführung.

Nach →HTML 4 geht das noch einfacher mit Markierungen der Form

```
<h2 id="kap1">Kapitel-Ueberschrift 1</h2>
Text des ersten Kapitels ...
</p><hr><p>
```

Es gibt auch →Umwandlungsprogramme (wie z.B. rftohtml) und Hilfsprogramme, die solche "klickbare" Inhaltsverzeichnisse automatisch aus allen im HTML-File vorkommenden Überschriften erstellen, mit einer der Hierarchie der Befehle <h1> <h2> <h3> entsprechenden Schachtelung der Liste.

---

## 4. Bilder und Töne

---

- →Vorwort
- →Bilder – ja oder nein?
- →Die Wirkung auf die Menschen
- →Inline-Bilder `<img>` `<object>`
- →Inline-Objekte `<object>`
- →externe Bilder, Töne, Filme
- →Kleine und große Bilder (thumbnails)
- Spezialeffekte:
  - →Anordnung (`align`)
  - →Image-Maps `<map>` `usemap`

### 4.1 Bilder – ja oder nein?

"Ein Bild sagt mehr als tausend Worte."

Ein Bild braucht aber auch mehr Speicherplatz und vor allem mehr Übertragungszeit als tausend Worte. Dies schafft insbesondere dann Probleme, wenn der Benutzer auf das →World Wide Web über langsame oder teure Telefon-Leitungen oder mit PCs mit geringen Speichergrößen zugreift.

Die meisten →Web-Browser haben deshalb eine Option, die Darstellung von Bildern auszuschalten, und die wird erfahrungsgemäß auch von vielen Benutzern verwendet.

Es ist also empfehlenswert, Bilder nur dann einzusetzen, wenn sie tatsächlich mehr Informationen als Worte enthalten oder wenn der optische Eindruck wesentlich ist, und mit unnötigen Verzierungen sparsam umzugehen.

Natürlich gibt es viele Anwendungen, bei denen auf Bilder nicht verzichtet werden kann und soll, von Landkarten, Konstruktionszeichnungen und Diagrammen bis zu Firmen-Logos und Kunstwerken. Auch können Symbole und Piktogramme für den Benutzer sehr nützlich sein, wenn sie richtig und konsistent eingesetzt werden, und die zugehörigen Files sind meistens so klein, dass sie nur wenig Probleme mit Speicherplatz und Übertragungszeit bewirken. Allerdings müssen dabei stets →Alternativen für den Fall vorgesehen werden, dass die Symbole vom →Web-Browser nicht dargestellt werden.

Bei →Inline-Bildern, die mit dem jeweiligen HTML-File sofort mitübertragen werden müssen, ist Sparsamkeit besonders wichtig. Größere Bilder, die eine höhere Netzbelastung und Übertragungszeit benötigen, sollten deshalb besser als →externe Files zur Verfügung gestellt werden, die vom Benutzer nur auf ausdrücklichen Wunsch übertragen werden, eventuell mit der Hilfe von sogenannten →"Thumbnails".

Jedenfalls sollten Sie als Autor von →Web-Pages darauf achten, dass ein buntes und ansprechendes Aussehen nicht den Zugriff auf die wesentlichen Informationen behindert oder von ihnen →ablenkt. Dies ist so ähnlich wie die Entscheidung zwischen einer Ansichtskarte oder einem Brief.

## 4.2 Die Wirkung auf die Menschen

Wenn man neben normalem Text auch andere Elemente wie Bilder, Töne, Videosequenzen und dergleichen einsetzen will, gibt es grundsätzlich zwei Möglichkeiten:

- →Inline-Bilder oder →Inline-Objekte, die in die Web-Page eingebunden sind und gleichzeitig mit dem Text dargestellt werden,
- →externe Bilder, Animationen, Tonsequenzen, Videosequenzen etc., die für sich allein, separat vom Text dargestellt werden.

Auf die →technischen Unterschiede in Hinblick auf Erreichbarkeit, Übertragungszeit etc. wurde bereits →oben hingewiesen.

Hier folgen ein paar Anmerkungen zur unmittelbaren Wirkung auf die Menschen in Hinblick auf den vom Autor der Web-Page gewünschten Effekt.

Im Fall der externen Objekte wird jedes Objekt dem Betrachter bzw. Zuhörer für sich allein dargeboten und kann daher mit der vollen Aufmerksamkeit des Menschen rechnen.

Im Fall von Inline-Bildern und Objekten strömen jedoch mehrere Reize gleichzeitig auf den Menschen ein und konkurrieren um seine Aufmerksamkeit. Man muss daher ein paar psychologische Grundkenntnisse beachten, um den gewünschten Effekt zu erzielen und ihn nicht selbst zu zerstören:

**Bilder** ziehen die Aufmerksamkeit meist rascher und stärker an als Texte. Wenn z.B. ein Text vier Elemente beschreibt und nur drei von diesen Elementen durch eine Abbildung illustriert sind, werden viele Leser das vierte Element, bei dem das Bild fehlt, übersehen.

Wenn man will, dass die Leute den Text lesen, dann muss man ihn entweder komplett bebildern oder auf Inline-Bilder komplett verzichten.

**Bewegte** Objekte ziehen immer sofort die gesamte Aufmerksamkeit des Menschen auf sich und lenken damit stark von allen nicht bewegten Elementen ab. Wenn sich irgendwo auf dem Bildschirm etwas bewegt oder dreht oder etwas blinkt, ist es für den Menschen fast unmöglich, einen Text konzentriert zu lesen oder ein Bild genau zu betrachten.

Wenn man will, dass die Leute den Text lesen, darf man seine Wirkung nicht durch bewegte Effekte stören. Animationen und Videosequenzen werden nur dann sinnvoll eingesetzt, wenn sie die Hauptinformation enthalten und sich die gesamte Aufmerksamkeit des Lesers auf sie konzentrieren soll.

Besondere Vorsicht ist bei allen **akustischen** Effekten geboten, hier vor allem in Hinblick auf die Umgebung. Akustische Signale, Sprachausgabe, Hintergrundmusik usw. können sich extrem störend auswirken, wenn mehrere Personen in einem

Raum arbeiten oder wenn neben dem Computer auch andere Schallquellen vorhanden sind, z.B. bei Gesprächen mit persönlich anwesenden Kunden oder bei telefonischen Auskünften.

Deshalb sollte man alle Töne und Geräusche möglichst immer nur als →externe Objekte realisieren, die nur auf ausdrücklichen Wunsch ("Anklicken") des Benutzers abgespielt werden, und niemals als →Inline-Objekte, die automatisch beim Ansehen einer Web-Page laut werden.

Außerdem sollte man beachten, dass manche Menschen gewisse **Behinderungen** aufweisen können, wie z.B. Kurzsichtigkeit, Farbenblindheit oder Schwerhörigkeit, und dass es Fälle gibt, wo jemand die Web-Page auf einem einfachen Schwarz-weiß-Drucker ausdrucken und den Printout dann anderen Menschen zeigen möchte.

Man sollte daher immer Alternativen vorsehen, damit auch diese Menschen die wesentlichen Informationen erfassen können, also z.B. immer Text-Alternativen zu Bildern oder Tönen angeben, konventionelle Markierungen zusätzlich zu Farbeffekten verwenden, die vom Leser eingestellten Schriftgrößen und Bildschirmfarben nicht verändern, usw. Entsprechende Hinweise finden Sie in den jeweiligen Kapiteln der HTML-Einführung.

### 4.3 Inline-Bilder <img> <object>

Unter Inline-Bildern versteht man Bilder und Symbole, die *zwischen* dem Text innerhalb eines HTML-Files erscheinen – im Gegensatz zu →externen Bildern, die nur auf ausdrücklichen Wunsch des Benutzers und als separate Web-Page übertragen werden.

Ein Inline-Bild wird in die Web-Page mit einem Befehl der Form  
``  
eingefügt.

*In →XHTML muss man stattdessen*

``  
*schreiben.*

Der URL gibt das File an, in dem das Bild gespeichert ist. Im einfachsten Fall (→relativer URL) besteht dieser URL nur aus dem Filenamen des Bildes, mit der richtigen "Extension":

`xxxx.gif` für Bilder im GIF-Format,  
`xxxx.jpg` für Bilder im JPEG-Format.

Dies sind die beiden Bildformate, die von den meisten →Web-Browsern unterstützt werden. Andere Format wie "X Bitmap" oder "MS Windows Bitmap" werden nicht universell unterstützt und sollten deshalb nicht verwendet werden.

Die Erzeugung der Bild-Files erfolgt mit spezieller Graphik-Software wie z.B. Paintshop oder dergleichen. Hinweise zum Erstellen von Graphiken finden Sie bei →Stefan Münz. Wichtig ist, dass die Auflösung und die Farbpalette auf die von den meisten Web-Browsern unterstützten Möglichkeiten abgestimmt ist und das File nicht durch eine unnötig hohe Auflösung oder Farbgenauigkeit viel zu groß wird und damit die Übertragung an den Client viel zu lange dauern würde.

Mit dem Parameter `alt=` muss der Text angegeben werden, der an Stelle des Bildes am Bildschirm erscheinen soll, falls der Benutzer die Darstellung von Bildern ausgeschaltet hat oder einen Web-Browser verwendet, der keine Bilder darstellt, sowie für →Suchmaschinen. Wenn der Parameter `alt=` nicht angegeben wird, geben die meisten Web-Browser an Stelle des Bildes einen sinnlosen Textstring wie "[IMAGE]" oder ein leeres Bild-Symbol (z.B. ein buntes Rechteck) aus.

**Beispiel:**

--- *Die Eingabe von*

```

```

--- *bewirkt eine Darstellung wie*



*(Je nach dem verwendeten Web-Browser sollten Sie hier entweder das bunte WWW-Logo oder die drei Buchstaben WWW sehen.)*

---

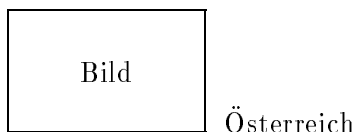
In vielen Fällen ist es günstiger, den erklärenden Text nicht nur *anstelle* des Bildes sondern *immer* neben dem Bild auszugeben und den alt-Text dafür auf den Leerstring zu setzen.

**Beispiel:**

--- *Die Eingabe von*

```
 &Ouml;sterreich
```

--- *bewirkt eine Darstellung wie*



---

Der mit `alt=` angegebene Text kann keine HTML-Befehle enthalten. In →HTML 4 ist deshalb als Alternative zum Befehl

```

```

auch ein wesentlich mächtigerer Befehl der Form

```
→<object data="URL"> Text </object>
```

vorgesehen, bei dem im Text auch weitere HTML-Befehle verwendet werden können (siehe die →Beschreibung des `<object>`-Befehls).

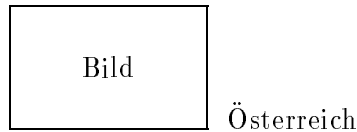
Mit speziellen →Align-Parametern kann die Position des Bildes und des nachfolgenden Textes verändert werden.

**Beispiel:**

--- *Die Eingabe von*

```
<p align=center>
 &Ouml;sterreich
</p>
```

— — — bewirkt eine Darstellung wie



— — —

Weitere Beispiele dazu finden Sie im Kapitel →"Layout und Spezialeffekte".

Bitte beachten Sie die Hinweise auf die →Nachteile von allzu vielen allzu großen Inline-Bildern und auf die Möglichkeit von →externen Bildern und →"Thumbnails".

#### 4.4 Inline-Objekte <object>

In →HTML 4 ist als Ersatz für die Befehle →<img>, <embed>, <bgsound>, <applet> u.a. ein wesentlich mächtigerer <object>-Befehl vorgesehen. Allerdings wird der <object>-Befehl von vielen Web-Browsern noch nicht oder noch nicht vollständig unterstützt. Es kann daher sinnvoll sein, eher auf den →<img>-Befehl oder auf →externe Dateien auszuweichen.

Die wichtigsten Erweiterungen gegenüber dem →<img>-Befehl sind:

- Mit <object> können nicht nur Bilder sondern auch andere Objekte in eine Web-Page eingebaut werden, Töne, Animationen, Video-Sequenzen, Java-Programme und andere, je nachdem, was die jeweiligen Web-Browser unterstützen. Allerdings müssen die →technischen und →psychologischen Hinweise beachtet werden, wie weit das sinnvoll ist und ob man diese Objekte nicht lieber als →externe Dateien separat zur Verfügung stellt.
- Der mit  
→  
angegebene Text kann keine HTML-Befehle enthalten. Der mit  
<object ... > Text </object>  
angegebene Alternativ-Text kann hingegen auch HTML-Befehle und sogar weitere Inline-Elemente enthalten.

Der <object>-Befehl hat grundsätzlich den folgenden Aufbau:

```
<object data="URL" type="MIME-Type">
... HTML-Text für die Alternative ...
</object>
```

Mit URL und MIME-Type wird angegeben, welche Datei eingefügt werden soll und um was für eine Art von Datei es sich handelt. Hier ein paar Beispiele für Filenamen und zugehörige MIME-Types:

```

xxxx.gif      image/gif
xxxx.jpg      image/jpeg
xxxx.mid      audio/midi
xxxx.wav      audio/x-wav
xxxx.mov      video/quicktime
\condbreak{3.5cm}
xxxx.mov      video/quicktime
xxxx.mgp      video/mpeg
xxxx.ps       application/postscript
xxxx.pdf      application/pdf
xxxx.txt      text/plain

```

### Beispiel 1:

```

<h3>Zugriffsstatistik</h3>
<p>
<object data="zugriff.gif" type="image/gif">
  <table border>
    <tr><th>Jahr
      <th>Files<br>am Server
      <th>Zugriffe<br>pro Monat
    <tr><td align=right>1993
      <td align=right>0
      <td align=right>0
    <tr><td align=right>1994
      <td align=right>160
      <td align=right>7.000
    <tr><td align=right>1995
      <td align=right>410
      <td align=right>25.000
    <tr><td align=right>1996
      <td align=right>6.500
      <td align=right>158.000
    <tr><td align=right>1997
      <td align=right>14.500
      <td align=right>325.000
    </table>
  </object>

```

Wenn der Web-Browser <object> versteht und die Darstellung von Bildern eingeschaltet ist, erscheint das Diagramm mit der graphischen Darstellung der Zugriffsstatistik. Wenn nicht, werden die Zugriffszahlen in Form einer Tabelle

Jahr	Files am Server	Zugriffe pro Monat
1993	0	0
1994	160	7.000
1995	410	25.000
1996	6.500	158.000
1997	14.500	325.000

- - -

**Beispiel 2:**

```
<h3>Entwicklung eines Schmetterlings</h3>
<p>
<object data="butterfly.mpg" type="video/mpeg">
  <p>
  Der
  <a href="/brehm/tierleben.html#schmett">Schmetterling</a>
  durchläuft 4 Entwicklungsstufen:
  <ol>
  <li> Ei
  <li> Raupe
  <li> Puppe (Kokon)
  <li> Schmetterling
  </ol>
</object>
```

Wenn der Web-Browser `<object>` versteht und MPEG-Videos darstellen kann, erscheint ein Video, das die Entwicklung des Schmetterlings zeigt. Wenn nicht, werden die 4 wichtigsten Phasen der Entwicklung mit je einem Bild plus erklärendem Text gezeigt bzw., falls die Darstellung von Bildern ausgeschaltet ist, nur die erklärenden Texte. Das sieht dann etwa so aus:

**Entwicklung eines Schmetterlings**

Der → Schmetterling durchläuft 4 Entwicklungsstufen:

1. Ei
2. Raupe
3. Puppe (Kokon)
4. Schmetterling

- - -

## 4.5 Externe Bilder, Töne, Filme

Im Gegensatz zu  $\rightarrow$ Inline-Bildern und  $\rightarrow$ Inline-Objekten, die automatisch innerhalb von Text-Files dargestellt werden, sollen größere Bilder sowie Töne, Filme und dergleichen extern, d.h. mit Hilfe von  $\rightarrow$ Hypertext-Links der Form

```
<a href="URL">Link-Text</a>
```

realisiert werden.

In diesem Fall sieht der Benutzer zunächst nur den Link-Text (oder einen sogenannten  $\rightarrow$ "Thumbnail") am Bildschirm und bekommt das Bild bzw. die Tonfolge oder den Video-Film erst dann auf seinen  $\rightarrow$ Client übertragen, wenn er dieses Link  $\rightarrow$ "auswählt".

Darin ist "URL" der URL des Files, das das Bild oder die Tonfolge oder den Film enthält. Im einfachsten Fall ( $\rightarrow$ relativer URL) besteht dieser URL nur aus dem Filenamem mit der richtigen "Extension". Die Extension des Filenamens gibt dem Web-Browser an, was das File enthält.

Beispiele:

```
xxxx.html für HTML-Files,  
xxxx.ps für PostScript-Files,  
xxxx.gif für Bilder im GIF-Format,  
xxxx.jpg für Bilder im JPEG-Format,  
xxxx.mid für Töne im MIDI-Format,  
xxxx.wav für Töne im WAV-Format,  
xxxx.mov für Video-Filme im QuickTime-Format,  
xxxx.mpg für Video-Filme im MPEG-Format.
```

Je nach der Extension wird vom  $\rightarrow$ Web-Browser die entsprechende "Helper-Applikation" aufgerufen, die das File verarbeitet und den Inhalt ausgibt. Natürlich funktioniert das nur, wenn auf dem Client-Rechner die entsprechende Software installiert ist.

Es kann nützlich sein, bei Links auf große Files anzugeben, wie groß das betreffende File ungefähr ist, damit der Benutzer entscheiden kann, ob er es übertragen will, und abschätzen kann, wie lange die Übertragung dauern wird.

**Beispiel:**

--- *Die Eingabe von*

```
<a href="big.jpg">Gro&szlig;es Bild ohne Bikini (JPEG, 45k)</a>
```

--- *bewirkt eine Darstellung wie*

$\rightarrow$ Großes Bild ohne Bikini (JPEG, 45k)

---

## 4.6 Kleine und große Bilder (thumbnails)

Im Fall von großen Bildern kann es sinnvoll sein, eine *kleine* Version des Bildes als →Inline-Bild ins HTML-File einzufügen, das gleichzeitig ein →Link auf das eigentliche, große Bild ist.

Klein bedeutet hier nicht nur eine kleine Darstellung am Bildschirm oder Printout sondern vor allem auch eine geringe Dateigröße und damit kürzere Übertragungszeit (z.B. durch eine weniger genaue Auflösung und weniger Farben als das Originalbild).

Dies hat den Vorteil, dass der Benutzer sofort sehen kann, um was für ein Bild es sich handelt, und dadurch besser entscheiden kann, ob er das große Bild tatsächlich auf seinen →Client übertragen will.


Diese kleinen Bilder sehen meist wie Briefmarken aus und werden im Fachjargon als "Daumennägel" (thumbnails) bezeichnet.

### Beispiel:

– – – *Die Eingabe von*

```
<a href="big.jpg">  
  . . . "Geburt der Venus" von Sandro Botticelli (45k)</a>
```

– – – *bewirkt eine Darstellung wie*

→  . . . "Geburt der Venus" von Sandro Botticelli (45k)

– – –

---

## 5. Layout und Spezialeffekte

---

- →Vorwort
- →Schönes Layout mit HTML – wie geht das?
  - →Logisches Markup und Layout-Hinweise
  - →Wenn die Glocken locken...
  - →Norm oder nicht Norm, das ist hier die Frage
- →Klassen (class) und Style-Sheets <style>
- →Schrift
  - →Schriftarten
  - →Schriftgrößen
- →Farben
  - →Wie werden Farben sichtbar?
  - →Farben mit <em> <strong> class <style>
  - →Farben mit <body> <em> <strong> <font>
- →Anordnung (align)
  - →linksbündig, rechtsbündig, zentriert
  - →Chaos oder Harmonie
  - →unten, oben, neben Bildern
- →Abstände
  - →horizontale Abstände
  - →Einrückungen
  - →vertikale Abstände
- →Trennlinien <hr>
- →Numerierungen <ol>
- →Frames <frameset> <frame> <noframes>
- →Navigationshilfen <link>
- →Schlagwörter für Suchhilfen <meta>

- →Interaktion mit dem Benutzer – mehr Leben ins World Wide Web
  - →CGI-Programme am Server
  - →Java-Programme am Client <applet> <object>
  - →JavaScript und →Active-X am Client <script>
  - →Server-Side Includes und →Dynamic HTML
  - →Zugriffe zählen
  - →Formulare <form>
  - →Image-Maps <map> usemap
  - →Electronic Mail (mailto)
  - →Sichere Übertragung mit SSL und Secure HTTP

## 5.1 Schönes Layout mit HTML – wie geht das?

- →Inhalt und Form
- →Logisches Markup und Layout-Hinweise
- →Wenn die Glocken locken...
- →Richtige HTML
- →Norm oder nicht Norm, das ist hier die Frage

### 5.1.1 Logisches Markup und Layout-Hinweise

Wie im Abschnitt →"Inhalt und Form" erläutert, wird im →HTML-File am →WWW-Server nur die *logische Bedeutung* der Text-Teile festgelegt, und sie werden von den →Web-Browsern der Benutzer auf deren →Client-Bildschirmen jeweils so dargestellt, wie es für diesen Benutzer und diesen Client am besten ist.

Man kann aber im HTML-File *Hinweise* angeben, wie und in welchem Layout die Elemente am Client nach Maßgabe der technischen Möglichkeiten und der persönlichen Einstellungen des Benutzers dargestellt werden sollen. Damit kann das Layout zwar nicht zwingend vorgeschrieben, aber weitgehend *beeinflusst* werden. Die verschiedenen Web-Browser befolgen diese Hinweise jeweils *so gut wie möglich*. Besonders interessant ist in diesem Zusammenhang die seit →HTML 3.2 vorgesehene Möglichkeit von →"Style-Sheets".

In Spezialfällen, bei denen das *genaue* Layout wichtig ist, kann es sinnvoll sein, die Information als →PostScript- oder →PDF- oder →RTF-File oder als →Bild oder →"Image-Map" verfügbar zu machen.

### 5.1.2 Wenn die Glocken locken...

Es ist niemals günstig, spezielle Layout-Effekte dadurch zu erreichen, dass man HTML-Befehle für etwas anderes als deren  $\rightarrow$ logische Bedeutung einsetzt.

Es mag zum Beispiel lustig aussehen, in einem Text alle Anfangsbuchstaben mit fetten Buchstaben darzustellen. Aber in einer Konstruktion wie

```
<strong>F</strong>rohe <strong>F</strong>eier  
mit <strong>G</strong>locken und <strong>F</strong>l&ouml;ten
```

würden manche  $\rightarrow$ Suchhilfen vielleicht nicht die Stichwörter *frohe*, *Feier*, *Glocken und Flöten* finden sondern die Stichwörter *rohe*, *Eier*, *locken und löten*, und auch im Falle einer Sprachausgabe oder bei einer automatischen Übersetzung in eine andere Sprache könnten dieselben Mißverständnisse auftreten.

Solche Spezialeffekte kann man besser mit Hilfe von  $\rightarrow$ Klassen und  $\rightarrow$ Style-Sheets erreichen.

### 5.1.3 Norm oder nicht Norm, das ist hier die Frage

Sie als Autor haben die Wahl:

- Entweder Sie berücksichtigen die  $\rightarrow$ De-facto-Norm von HTML, dann sind Ihre Informationen für *alle* Interessenten lesbar,
- oder Sie verwenden unbedacht  $\rightarrow$ Browser-spezifische Erweiterungen, dann sehen Ihre Informationen für *manche* Leser ganz besonders schön aus und sind für viele andere völlig unleserlich.

Die Kunst des HTML-Autors besteht darin, die genormten alten HTML-Befehle und die Browser-spezifischen neuen HTML-Befehle so zu kombinieren, dass die Webseiten mit den gängigen Web-Browsern möglichst schön aussehen und trotzdem mit *allen* Web-Browsern zumindestens sinnvoll lesbar sind.

Bitte beachten Sie in diesem Zusammenhang auch die Warnung vor  $\rightarrow$ Firmenabhängigkeiten.

Diese Grundsätze gelten nicht nur für die HTML-Files sondern für *alle* Files, die für den Zugriff durch viele Benutzer gedacht sind. Wenn Sie zum Beispiel einen Text im Binär-Format von "MS-Word 6.0 für Windows" abspeichern, kann er von Benutzern anderer PC-Programme oder von Macintosh-Benutzern nicht verwendet werden. Günstiger wäre es in diesem Fall, den Text im "Rich Text Format" (RTF) abzuspeichern, das von allen Textverarbeitungsprogrammen verstanden wird, oder als  $\rightarrow$ PostScript-File oder noch besser als  $\rightarrow$ normgerechtes HTML-File.

## 5.2 Klassen (class) und Style-Sheets <style>

→HTML 3.2 sieht eine Möglichkeit vor, vom Autor definierte Markups ("Klassen") für spezielle Elemente zu verwenden und in eigenen "Style-Sheets" anzugeben, in welchem **Layout** die Elemente von den →Web-Browsern nach Maßgabe der technischen Möglichkeiten dargestellt werden sollen.

Diese Style-Sheets sind ein sehr mächtiges Instrument. Sie bieten dem Autor vielfältige Möglichkeiten, sehr schöne und wirkungsvolle, aber auch sehr häßliche oder verwirrende Layouts zu erzeugen. Deshalb ist ein gutes →Konzept und die Hilfe von professionellen Layout-Designern oder von guter Fachliteratur zu diesem Thema empfehlenswert.

### Beispiele:

Ein Autor kann besonders wichtige Hinweise zwischen `<p class="wichtig">` und `</p>` einschließen und im Style-Sheet angeben, dass alle mit `class="wichtig"` markierten Absätze eingerückt und in einer großen, fetten Schrift geschrieben werden sollen.

Ein Autor kann das sogenannte "Kleingedruckte" in einem Vertragstext mit `<p class="unwichtig">` markieren und im Style-Sheet angeben, dass alle mit `class="unwichtig"` markierten Texte in einer kleineren Schrift auf hellblauem Hintergrund geschrieben werden sollen.

Eine Firma kann mit Hilfe eines Style-Sheet für alle ihre Web-Pages ein besonderes, einheitliches, für die Firma typisches Layout festlegen ("Corporate Identity"): die Schriftart, die Farben von Text und Hintergrund, die Abstände und Einrückungen von Absätzen und Listen, die Größe und Farbe der Überschriften, und so weiter.

Ein Autor kann für Gruppen von Web-Pages, die für verschiedene Zwecke dienen (z.B. Marketing oder technische Details) oder die sich an verschiedene Zielgruppen wenden (z.B. an Kinder oder Erwachsene) jeweils verschiedene, dafür geeignete Layouts einsetzen.

---

Wenn der Web-Browser keine Style-Sheets unterstützt, werden die Class-Hinweise →ignoriert. Wenn ein Client bestimmte im Style-Sheet geforderte Elemente nicht darstellen kann (z.B. keine Farben), werden diese Style-Angaben ignoriert oder durch andere Effekte ersetzt. Klassen und Style-Sheets geben dem Autor also die Möglichkeit, das zum →logischen Markup gehörende →Layout so zu spezifizieren, dass das Aussehen am →Client so genau wie möglich seinen Wünschen und Vorstellungen entspricht, die Information aber trotzdem auf *allen* verschiedenen Clients dargestellt werden kann.

Die "Style-Sheets" werden am besten als separate Files gespeichert, die von mehreren HTML-Files verwendet werden können. Dieses Style-File wird jeweils im →Head des HTML-Files mit einem Link-Befehl der Form

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

angegeben. Einzelne Style-Angaben können auch innerhalb des HTML-Files angegeben werden, und zwar zwischen `<style>` und `</style>` ebenfalls innerhalb des →Head.

Der Inhalt der Style-Sheets wird in einer Sprache wie CSS1 (Cascading Style Sheets) oder DSSSL (Document Style and Semantics Specification Language) festgelegt. Für die kompletten Spezifikationen und für Hinweise und Beispiele zu ihrer Verwendung wird auf die →Referenzen verwiesen.

Hier nur ein kurzes **Beispiel**, eine Skizze für eine →bunte Seite, auf der der normale Text schwarz auf weiß erscheinen soll, wichtiger Text rot auf weiß in fetter Schrift, die Bezeichnung "BOKU" grün in der jeweiligen Schriftart, und Hinweise (blockquotes) eingerückt werden:

```
<html>
<head>
<title>Die bunte Studentenseite der BOKU</title>

<style type="text/css">
BODY { color: black; background: white; font-family: Times, serif }
A:link { color: #990000; background: white;
        text-decoration: underline }
A:visited { color: #660000; background: white;
           text-decoration: underline }
A:active { color: #FF0000; background: yellow }
.boku    { color: #009900; background: white }
.wichtig { color: red; background: white;
          font-style: normal; font-weight: bold }
EM { font-style: normal; font-weight: bold }
BLOCKQUOTE { margin-left: 10%; margin-right: 10% }
</style>

</head>
<body>
<h1>PCs f&uuml;r <em class="boku">BOKU</em>-Studenten</h1>
<p>
Der <em>Zentrale Informatikdienst</em> der
<em class="boku">Universit&auml;t f&uuml;r Bodenkultur</em>
stellt den
<em class="boku">BOKU</em>-Studenten
PCs in den Benutzerr&auml;umen zur Verf&uuml;gung.
<blockquote class="wichtig">
Achtung!
Diese PCs d&uuml;rfen nur f&uuml;r die Lehre und Forschung der
<em class="boku">BOKU</em>
verwendet werden und nicht f&uuml;r andere Zwecke!
</blockquote>
Genauere Informationen ...
</body>
</html>
```

- - -

Das Ergebnis wird dann so ähnlich wie die folgenden Zeilen aussehen:

## PCs für BOKU-Studenten

Der **Zentrale Informatikdienst** der **Universität für Bodenkultur** stellt den BOKU-Studenten PCs in den Benutzerräumen zur Verfügung.

**Achtung! Diese PCs dürfen nur für die Lehre und Forschung der BOKU verwendet werden und nicht für andere Zwecke!**

Genauere Informationen ...

- - -

*Wie Ihr Web-Browser dieses Beispiel tatsächlich darstellt, können Sie →hier ausprobieren.*

Style-Sheets werden nur von den neueren Web-Browsern unterstützt und auch von denen zum Teil nur unvollständig.

Deshalb sollten Sie die Style-Angaben *nicht* wie im obigen Beispiel zwischen `<style>` und `</style>` innerhalb des HTML-Files angeben (weil das manche Browser verwirren könnte) sondern in einem separaten Style-File, das mit `<link>` angegeben wird, und Sie sollten die Style- und Klassen-Angaben mit den von verschiedenen Web-Browsern unterstützten anderen HTML-Tags wie z.B. →`<em>`, →`<strong>`, →`<blockquote>`, →`<body>`, etc. so kombinieren, dass Sie den gewünschten Effekt auf möglichst vielen Web-Browsern erreichen.

## 5.3 Schrift

- →Schriftarten
- →Schriftgrößen
- →Schrift- und Hintergrundfarben

### 5.3.1 Schriftarten

In Spezialfällen kann man die Schriftart mit folgenden Befehlen festlegen:

```
<b> . . . . . </b> für fette Schrift (boldface),  
<i> . . . . . </i> für kursive Schrift (italic),  
<u> . . . . . </u> für Unterstreichungen (underline),  
<tt> . . . </tt> für nicht proportionale Schrift (teletype),  
<code> . . . </code> für Computer-Schrift.
```

Diese Befehle sollten Sie nur dann verwenden, wenn eine bestimmte Schriftart per Konvention vorgeschrieben ist (z.B. Fettdruck für Vektoren und Kursivdruck für skalare Variable in der Mathematik, Computer-Schrift für Programmbeispiele, Kursivdruck für lateinische Pflanzen- und Tiernamen und dergleichen).

Wenn es jedoch darum geht, Wörter oder Textteile mit einer bestimmten Bedeutung vom normalen Text abzuheben, sollten Sie *nicht* diese Befehle verwenden sondern die Markierung mit →`<em>` oder →`<strong>`.

Unterstreichungen sollten Sie überhaupt *vermeiden*, weil sie Verwechslungen mit →Hypertext-Links bewirken würden.

Manche Schriftarten können von manchen →Clients nicht dargestellt werden, sie werden dann durch eine andere geeignete Schriftart ersetzt.

### 5.3.2 Schriftgrößen

Die Schriftgröße wird vom Benutzer auf seinem →Client so eingestellt, wie es für seine Bildschirm-Auflösung und seine Augen am besten ist.

In →HTML 3.2 und 4 sind mehrere Möglichkeiten vorgesehen, die Schriftgröße von bestimmten Wörtern oder Textteilen zu →beeinflussen:

`<big> . . . . . </big>` für größere Schrift,  
`<small> . . . </small>` für kleinere Schrift,  
`<font size="+n"> . . . </font>` für eine um n Stufen größere Schrift,  
`<font size="-n"> . . . </font>` für eine um n Stufen kleinere Schrift,  
`<font size=n> . . . . . </font>` für Schrift in einer bestimmten Größe,  
sowie mit →Style-Sheets. Ob und wie diese Größenangaben wirken, hängt vom →Web-Browser und vom →Client-Bildschirm ab.

Viele, aber nicht alle Web-Browser stellen →Überschriften der Ebenen 1 bis 3 durch fette und große Schrift dar, manche stellen Überschriften der Ebenen 5 und 6 durch kleinere Schrift dar. Die entsprechenden Befehle `<hx>` sollten aber wirklich nur für Überschriften der jeweiligen Ebene verwendet werden und nicht für andere Zwecke, denn das hätte unerwünschte Nebeneffekte auf eine eventuelle Einrückung der nachfolgenden Texte sowie auf →Suchhilfen und automatisch erstellte Inhaltsverzeichnisse.

## 5.4 Farben

### 5.4.1 Wie werden Farben sichtbar?

Im allgemeinen ist es günstiger, wenn Sie dem Leser überlassen, welche Schrift- und Hintergrundfarben er auf seinem →Client-Rechner verwendet - in Abhängigkeit von den technischen Möglichkeiten seines Bildschirms und Druckers, den Lichtverhältnissen in seinem Zimmer, der Sehschärfe seiner Augen, seinen Gewohnheiten, dem Zusammenspiel mit anderen Bildschirmfenstern - kurz, so wie es für ihn am angenehmsten ist.

Wenn Sie als Autor trotzdem auf Ihren Web-Pages bestimmte Farben für bestimmte Zwecke einsetzen wollen, dann müssen Sie die folgenden Punkte beachten:

- Wenn Sie die Farbe für den Text festlegen, müssen Sie auch eine dazu passende Farbe für den Hintergrund festlegen, damit nicht z.B. bei einem Benutzer, der weiße Schrift auf blauem Hintergrund eingestellt hat, plötzlich blaue Schrift auf blauem Hintergrund entsteht.
- Wenn Sie die Farbe für einen bestimmten Textteil festlegen, müssen Sie auch dazu passende Farben für alle anderen Textteile festlegen.
- Sie können sich nicht darauf verlassen, dass alle Leser die Farben tatsächlich sehen werden, deshalb müssen Sie zusätzlich immer auch andere Mittel einsetzen, um den erwünschten Effekt zu erreichen (z.B. →`<em>` oder →`<strong>`). Der Leser könnte den Text ja auf einem Schwarzweißdrucker ausdrucken oder ihn mit einem →Web-Browser ansehen, der keine Farben darstellt oder nur für 16 Grundfarben konfiguriert ist, oder er könnte farbenblind sein.

- Achten Sie bei der Wahl der Farben darauf, welche Schriftfarben auf welchen Hintergrundfarben gut lesbar sind, welche Farben ästhetisch zueinander passen, und welche Assoziationen sie bei den Lesern wecken (Verkehrszeichen, Temperaturen, Gefühle, politische Parteien, Firmen, Sportklubs, ...).

In →HTML 2 gibt es keine Befehle für die Spezifikation von Farben, sondern nur allgemein für Hervorhebungen mit →`<em>` und →`<strong>`. In →HTML 3.2 und 4 ist die Möglichkeit vorgesehen, Farben und andere graphische Effekte entweder im jedem Einzelfall mit Parametern in →`<body>` und `<font>` oder wesentlich eleganter und allgemeiner mit →Klassen und Style-Sheets zu vereinbaren. Diese Möglichkeiten werden freilich nicht von allen →Web-Browser unterstützt. Im folgenden wird angegeben, wie Sie diese verschiedenen Möglichkeit so *kombinieren* sollen, dass Sie den gewünschten Effekt auf →möglichst vielen Web-Browsern erreichen.

Die Farben werden in diesen Befehlen meist in der Form `#rrggbb` spezifiziert, wobei `rr`, `gg` und `bb` hexadezimale Zahlenangaben zwischen 00 (0) und FF (255) für den Rot-, Grün- und Blauanteil sind. Damit können theoretisch ca. 16 Millionen verschiedene Farben spezifiziert werden, praktisch sollten Sie sich jedoch auf die 216 Farben beschränken, die aus Kombinationen von 00, 33, 66, 99, CC und FF zusammengesetzt sind, oder überhaupt nur auf die aus 00 und FF zusammengesetzten Grundfarben. **Beispiele** (wobei die Sternchen je nach dem verwendeten Browser in den jeweiligen Farben erscheinen oder nicht):

```
#FFFFFF = weiß *****
#000000 = schwarz *****
#FF0000 = rot *****
#00FF00 = grün *****
#0000FF = blau *****
#FFFF00 = gelb *****
#FF00FF = magenta (lila) *****
#00FFFF = cyan (blaugrün) *****
#999999 = grau *****
#000066 = dunkelblau *****
#9999FF = hellblau *****
#660000 = dunkelrot *****
#FF9999 = rosa *****
```

In manchen Fällen können Farben auch durch bestimmte englische Wörter spezifiziert werden, z.B. white, black, red, blue, green.

Für die Realisierung gibt es zwei Möglichkeiten:

- die →neue, elegante Methode mit Style-Sheets, die allerdings nur von neuen Web-Browsern voll unterstützt wird,
- eine →fehleranfällige, etwas ältere Methode mit `<body>` und `<font>`, die von einigen Web-Browsern der mittleren Generation unterstützt wird.

### 5.4.2 Farben mit `<em>` `<strong>` `class` `<style>`

Die beste und eleganteste Methode ist es, die Farb-Angaben mittels  $\rightarrow$ Klassen-Angaben und in  $\rightarrow$ Style-Sheets zu realisieren. Dabei muss man beachten, dass bei unvollständigen Angaben durch das "Cascading" von allgemeinen und speziellen Benutzer- und Autoren-Angaben unerwünschte Effekte wie "blau auf blau" entstehen können, und deshalb immer in jedem einzelnen Fall sowohl die Vorder- als auch die Hintergrundfarben angeben. Außerdem muss man die Hervorhebung durch Farben (mit `class=`) auch durch "normale" Hervorhebungen mit `<em>` `<strong>` etc. kombinieren, damit der gewünschte Effekt auch bei älteren Web-Browsern oder Monochrom-Geräten (z.B. Notebooks oder Palmtops) und beim Ausdrucken auf Schwarz-weiß-Druckern erreicht wird.

#### Beispiel:

Sie wollen erreichen, dass der Name Ihrer Firma immer in blauer Farbe und/oder fetter oder stark hervorgehobener Schrift erscheint. Als Kontrast dazu wählen Sie weiß für den Hintergrund, schwarz für den normalen Text und verschiedene Rottöne für Hypertext-Links.

In Ihrem HTML-File "enzian.html" geben Sie zu diesem Zweck die folgende Kombination von Befehlen an:

```
<html>
<head>
<title>Enzian</title>
<link rev=made href="webmaster@www.enzian.com">
<link rel=stylesheet href="enzian.css" type="text/css">
</head>
<body>
<h1 class=enzian>Enzian</h1>
<p>
Wer
<strong class=enzian>Enzian</strong>
trinkt, wird schneller
<strong class=enzian>blau</strong>.
Dies wurde von
<a href="knieriem.html">namhaften Experten</a>
getestet ...
</body>
</html>
```

und im Style-Sheet-File "enzian.css" die folgenden Spezifikationen:

```
BODY { color: black; background: white }
A:link { color: #990000; background: white }
A:visited { color: #660000; background: white }
A:active { color: #FF0000; background: white }
.enzian { color: #0000FF; background: white; font-weight: bold }
```

Anmerkung: Theoretisch könnten Sie diese Style-Angaben direkt im HTML-File angeben, zwischen `<style type="text/css">` und `</style>` innerhalb von `<head>`,

anstelle des Style-Sheet-Link, aber manche nicht normgerechte Browser würden die Style-Angaben dann als Text am Bildschirm anzeigen statt sie zu interpretieren oder zu ignorieren.

Das Ergebnis wird dann so ähnlich wie die folgenden Zeilen aussehen:

### **Enzian**

Wer **Enzian** trinkt, wird schneller **blau**. Dies wurde von namhaften Experten getestet ...

- - -

*Wie Ihr Web-Browser dieses Beispiel tatsächlich darstellt, können Sie →hier ausprobieren.*

#### 5.4.3 Farben mit <body> <em> <strong> <font>

Viele (aber nicht alle) Web-Browser unterstützen die Angabe von Farben in <body> und <font>.

In <body> sind sämtliche Farben für normalen Text, Hintergrund und Link-Texte anzugeben, und man sollte unbedingt immer *alle* fünf Farben gemeinsam spezifizieren, um unerwünschte Effekte beim Zusammenspiel zwischen den vom Autor definierten Farben und den vom Benutzer in seinem Client eingestellten Präferenzen zu vermeiden.

In <font> kann man aber nur die Text-Farbe angeben, und damit können unerwünschte Effekte wie z.B. blauer Text auf blauem Hintergrund durch ein ungünstiges Zusammentreffen von Autoren-definierten Farben und Benutzer-Präferenzen nicht ausgeschlossen werden. Es wird deshalb dringend empfohlen, den <font>-Befehl *nicht* zu verwenden, sondern entweder nur die generellen Farbwünsche im <body>-Befehl oder die Spezifikation von Farben für spezielle Elemente mit →Klassen-Angaben und Style-Sheets.

Jedenfalls muss man <font>-Befehle immer mit "normalen" Hervorhebungen mittels <em> <strong> oder dergleichen kombinieren, damit der gewünschte Effekt auch bei älteren Web-Browsern oder Monochrom-Geräten und beim Ausdrucken auf Schwarzweiß-Druckern erreicht wird.

#### **Beispiel:**

Sie wollen erreichen, dass der Name Ihrer Firma immer in blauer Farbe und/oder fetter oder stark hervorgehobener Schrift erscheint. Als Kontrast dazu wählen Sie weiß für den Hintergrund, schwarz für den normalen Text und verschiedene Rottöne für Hypertext-Links.

Wenn Sie dafür die alte Methode mit <body> und <font> verwenden wollen, dann geben Sie die folgende Kombination von HTML-Befehlen an:

```
<html>
<head>
<title>Enzian</title>
<link rev=made href="webmaster@www.enzian.com">
</head>
```

```

<body text="#000000" bgcolor="#FFFFFF"
      link="#990000" vlink="#660000" alink="#FF0000" >
<h1><font color="#0000FF">Enzian</font></h1>
<p>
Wer
<strong><font color="#0000FF">Enzian</font></strong>
trinkt, wird schneller
<strong><font color="#0000FF">blau</font></strong>.
Dies wurde von
<a href="knieriem.html">namhaften Experten</a>
getestet ...
</body>
</html>

```

Besser wäre es aber, wie gesagt, *nicht* `<font>` sondern  $\rightarrow$  Style-Sheets zu verwenden.

Hier noch ein **Beispiel** dafür, wie Sie es *nicht* machen sollten:

– – – *Die Eingabe von*

```

<font color="#999999">Grau</font>
ist in Goethes "Faust" die Theorie, doch
<font color="#00FF00">gr&uuml;n</font>
des Lebens
<font color="#FF9900">goldner</font>
Baum, und was man
<font color="#000000">schwarz</font>
auf
<font color="#FFFFFF">wei&szlig;</font>
besitzt, kann man getrost nach Hause tragen.

```

– – – *bewirkt die Ausgabe von*

Grau ist in Goethes "Faust" die Theorie, doch grün des Lebens goldner Baum, und was man schwarz auf weiß besitzt, kann man getrost nach Hause tragen.

*und je nachdem, welchen Web-Browser Sie verwenden, sehen Sie hier entweder überhaupt keine Hervorhebung der Farbennamen, oder ein Teil dieser Wörter hebt sich nicht gut genug von der Hintergrundfarbe ab.*

– – –

## 5.5 Anordnung (align)

- $\rightarrow$  linksbündig, rechtsbündig, zentriert
- $\rightarrow$  Chaos oder Harmonie
- $\rightarrow$  unten, oben, neben Bildern

### 5.5.1 Linksbündig, rechtsbündig, zentriert

Die Anordnung und Ausrichtung von Texten und Bildern in Absätzen, Überschriften und Tabellen kann mit Align-Hinweisen in den HTML-Befehlen `→<p>`, `→<h1>`, `→<td>` usw. beeinflusst werden.

Die Align-Hinweise werden von manchen Web-Browsern befolgt und von anderen →ignoriert.

#### Beispiele:

– – – *Die Eingabe von*

```
<p align=left>
Linksbündig ist die normale Ausrichtung von Absätzen.
<p align=right>
Hoffentlich geht das
<br>
mit rechten Dingen zu.
<p align=center>
Im Reich der<br>Mitte
<p>
```

– – – *bewirkt eine Darstellung wie*

Linksbündig ist die normale Ausrichtung von Absätzen.

Hoffentlich geht das  
mit rechten Dingen zu.

Im Reich der  
Mitte

– – –

*In →XHTML muss man die Parameterwerte in Quotes einschließen, also z.B. align="center" statt align=center schreiben.*

Ein Bereich von mehreren Absätzen kann man mit dem Befehl `<div>` zusammengefaßt werden. So kann man z.B. mit einer Befehlsfolge wie

```
<div align=center>
<p>
erster zentrierter Absatz
<p>
zweiter zentrierter Absatz
</div>
```

eine Folge von mehreren zentrierten Absätze erreichen.

## Chaos oder Harmonie

Zentrierte und linksbündige Layouts sollen **niemals vermisch**t werden:

In linksbündigen Layouts sucht das Auge des Lesers entlang des linken Randes nach allen wichtigen Elementen (Überschriften, Aufzählungen, Numerierungen u.dgl.).

In zentrierten Layouts sucht der Leser in der Mitte des Bildschirms nach den wichtigen Elementen.

Wenn zentrierte und linksbündige Überschriften oder Listen abwechseln, funktioniert das rasche Auffinden der wichtigen Elemente nicht mehr, der Überblick geht verloren, und das Layout wird als unübersichtlich empfunden.

### Beispiel:

Chaos

ist schlecht lesbar.

Harmonie

ist gut lesbar.

— — —

Deshalb: entweder *alle* Überschriften zentriert, oder (besser) *alle* Überschriften linksbündig.

Umgekehrt ist es aber üblich und empfehlenswert, bestimmte "fremde Elemente" wie Titelseiten, Bilder, Tabellen oder mathematische Formeln durch Zentrierung deutlich vom laufenden Text abzuheben und damit die Übersichtlichkeit und Lesbarkeit des laufenden Textes zu vergrößern.

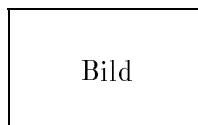
### Beispiel:

— — — *Die Eingabe von*

```
<p align=center>  

```

— — — *bewirkt eine Darstellung wie*



— — —

## Nicht <center> sondern <p align=center> oder <div align=center>

Manche, aber nicht alle Web-Browser erlauben als Alternative zum Befehlspar  
<p align=center> ... <p> bzw.  
<div align=center> ... </div>  
auch das Befehlspar  
<center> ... </center>

Dieses Befehlspar sollen Sie aber *nicht* verwenden, weil dabei die logische Struktur des Textes völlig unklar wird: Die Web-Browser, die diesen Befehl verstehen, beginnen nämlich bei <center> und </center> einen neuen Absatz, die anderen Web-Browser aber *nicht*, weil sie den unbekanntenen Befehl einfach ignorieren!

**Beispiel** (frei nach Otto Schenk in der "Fledermaus"):

— — — *Die Eingabe von*

Ich sagte:

```
<center>nichts</center>w&uuml;rldiger Herr Direktor!
```

— — — *bewirkt auf den meisten Browsern eine Darstellung wie*

Ich sagte:

nichts

würdiger Herr Direktor!

— — — *aber auf anderen Browsern eventuell eine Darstellung wie*

Ich sagte: nichtswürdiger Herr Direktor!

— — —

### 5.5.2 Unten, oben, neben Bildern

Bei →Inline-Bildern kann mit Align-Hinweisen im Befehl →<img> bzw. →<object> bestimmt werden, wie der Text *neben* dem Bild positioniert werden soll:

align=top für oben,  
align=bottom für unten,  
align=middle für auf der halben Höhe.

Die Align-Hinweise werden von manchen Web-Browsern befolgt und von anderen →ignoriert.

**Beispiele:**

— — — *Die Eingabe von*


Luft . . .

```

```

. . . Köpfe

— — — *bewirkt eine Darstellung wie*

Luft . . .  . . . Köpfe

— — — *Die Eingabe von*

Wasser . . .  
  
. . . Arme

-- bewirkt eine Darstellung wie

Wasser . . .  . . . Arme

--- Die Eingabe von

Erde . . .  
  
. . . Beine

-- bewirkt eine Darstellung wie

Erde . . .  . . . Beine

---

In  $\rightarrow$ XHTML muss man die Parameterwerte in Quotes einschließen, also z.B. *align="top"* statt *align=top* schreiben.

Seit  $\rightarrow$ HTML 3.2 ist auch die Möglichkeit vorgesehen, ein Bild nicht als Teil einer Textzeile darzustellen sondern so, dass der mehrzeilige Text neben dem Bild bzw. um das Bild herum läuft. Zu diesem Zweck wird im  $\rightarrow$ <img> bzw.  $\rightarrow$ <object> Befehl der Parameter

*align=left* bzw.

*align=right*

angegeben, und nach dem Text, der um das Bild herumlaufen soll, der Befehl

$\rightarrow$ <br clear=all>

damit der nachfolgende Text wieder mit der normalen Textbreite unter dem Bild erscheint.

### Beispiel:

-- Die Eingabe von

```
<p>  
  
World  
<br>  
Wide  
<br>  
Web  
<br clear=all>
```

-- bewirkt eine Darstellung wie

 World  
Wide  
Web

- - -

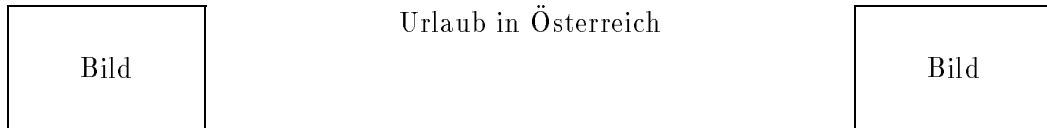
Auf diese Weise kann man auch zwei Bilder mit einem dazwischen →zentrierten Text kombinieren.

**Beispiel:**

- - - *Die Eingabe von*

```
<p>  
  
  
<p align=center>Urlaub in &Ouml;sterreich</p>  
<br clear=all>
```

- - - *bewirkt eine Darstellung wie*



- - -

Zu bedenken ist, dass nicht alle Web-Browser diese Möglichkeit unterstützen und der Text daher unter Umständen etwas anders positioniert oder erst nach den beiden Bildern erscheint.

## 5.6 Abstände

- →horizontale Abstände
- →Einrückungen
- →vertikale Abstände

### 5.6.1 Horizontale Abstände

Wenn man bestimmte Abstände zwischen Wörtern oder eine bestimmte Ausrichtung von Wörtern untereinander erreichen will, empfiehlt sich die Verwendung von →`<table>` oder von →`<pre>`.

Bei manchen Browsern kann man größere Wortabstände auch mit der Entity →`&nbsp;` oder mit unsichtbaren ("transparenten") →Inline-Bildern erreichen, dies führt aber bei anderen Browsern zu völlig unbrauchbaren Ergebnissen und sollte deshalb *nicht* verwendet werden.

In manchen Fällen kann man anstelle von größeren Wortabständen auch einfache andere Mittel einsetzen wie . . . Punkte oder - - - Gedankenstriche.

## 5.6.2 Einrückungen

Viele Web-Browser stellen  $\rightarrow$ Definition-Listen und  $\rightarrow$ Blockquotes durch eingerückte Absätze dar, andere aber nicht.

Trotzdem sollte man *immer*, wenn ein untergeordneter Text zu einem übergeordneten Text "dazugehört", eine  $\rightarrow$ Definition-Liste verwenden, egal ob der Browser dafür Einrückungen oder eine andere, für den Client besser geeignete Darstellung verwendet.

Ebenso sollte man für Absätze, die sich vom normalen Text abheben sollen, *immer*  $\rightarrow$ Blockquotes verwenden, egal ob der Browser dafür Einrückungen oder eine andere, für den Client besser geeignete Darstellung verwendet.

In anderen Fällen kann man Einrückungen mit  $\rightarrow$ <pre> oder mit  $\rightarrow$ Tabellen erreichen.

Hier folgt ein **Beispiel**, wie man den Bildschirm bzw. Printout in zwei Spalten im Verhältnis des "Goldenen Schnitts" (38:62:100) aufteilen kann:

– – – *Die Eingabe von*

```
<table border=0 width="100%">
<tr>
<td valign=top align=left width="38%">
<p><b>Goldener Schnitt:</b>
<td valign=top align=left width="62%">
<p>Der Goldene Schnitt ist die Aufteilung,
bei der sich der kleinere Teil zum größeren
so wie der größere zur Gesamtbreite
verhält, also
<p align=center>x : (1-x) = 1 : x
<p>Dieses Verhältnis gilt als besonders ästhetisch
und wird sowohl in der Kunst als auch in der Drucktechnik
oft eingesetzt.
<tr>
<td valign=top align=left width="38%">
<p>...
<td valign=top align=left width="62%">
<p>...
</table>
```

– – – *bewirkt eine Darstellung wie*

### **Goldener Schnitt:**

Der Goldene Schnitt ist die Aufteilung, bei der sich der kleinere Teil zum größeren so wie der größere zur Gesamtbreite verhält, also

$$x : (1-x) = 1 : x$$

Dieses Verhältnis gilt als besonders ästhetisch und wird sowohl in der Kunst als auch in der Drucktechnik oft eingesetzt.

...

- - -

Wichtig ist dabei, dass die `width`-Angaben immer nur *relativ* zur Bildschirm- bzw. Fenstergröße (also in Prozenten) und niemals absolut (in Pixeln oder Zentimetern) angegeben werden.

Allerdings kann man sich nicht darauf verlassen, dass das angegebene Verhältnis tatsächlich in allen Fällen eingehalten wird. Wenn das Bildschirmfenster zu schmal ist, um die Texte in den angegebenen Spalten darzustellen, kann der Web-Browser die Spaltenbreiten entsprechend anpassen.

Wichtig ist im obigen Beispiel außerdem die Angabe der `<p>`-Befehle innerhalb jeder Tabellenzelle. Damit wird sichergestellt, dass der Text auch von älteren Web-Browsern richtig dargestellt wird.

### 5.6.3 Vertikale Abstände

Manche Browser erzeugen bei mehrfachen `→<p>`- oder `→<br>`-Befehlen zusätzliche Leerzeilen, bei anderen Browsern werden aber leere Zeilen und leere Absätze ignoriert und alle Absätze und Zeilen mit einheitlichen Abständen dargestellt.

Da auf Bildschirmen meist wesentlich weniger Zeilen sichtbar sind als auf einer Papierseite, würden größere vertikale Abstände die Übersichtlichkeit nicht (wie auf dem Papier) erhöhen sondern verringern. Falls mehrere Leerzeilen zufällig am unteren Rand des Bildschirmfensters erscheinen, sieht das so aus, als wäre das gesamte File (die gesamte Web-Page) bereits zu Ende, und der Leser wird eventuell gar nicht mehr weiterlesen.

Für die Trennung von Text-Abschnitten sollten deshalb *nicht* zusätzliche Leerzeilen sondern `→Trennlinien` oder andere Elemente wie z.B.

```
<p align=center> * * * <p>
```

verwendet werden.

## 5.7 Trennlinien (horizontal rule) `<hr>`

Mit `<hr>` kann man eine waagrechte Trennlinie in den Text einfügen. Vor und nach der Linie erfolgt ein Zeilenwechsel oder neuer Absatz.

*In →XHTML muss man `<hr />` statt `<hr>` schreiben.*

## 5.8 Numerierungen in Listen <ol>

Standardmäßig werden die Listenelemente in →numerierten Listen mit den natürlichen Zahlen 1, 2, 3 ... numeriert. Bei den meisten Web-Browsern kann man die Numerierung durch die folgenden Parameter in den Befehlen <ol> und <li> beeinflussen:

```
<ol type=A> Numerierung mit Großbuchstaben A, B, C ...
<ol type=a> Numerierung mit Kleinbuchstaben a, b, c ...
<ol type=1> Numerierung mit Zahlen 1, 2, 3 ...
<ol type=I> Numerierung mit römischen Zahlen I, II, III, IV, V ...
<ol type=i> Numerierung mit kleinen römischen Zahlen i, ii, iii, iv, v ...

<ol start=n> Anfangswert der Numerierung

<li type=x value=n> spezieller Wert bei einem Listenelement
```

Diese Parameter werden von manchen →Web-Browsern unterstützt und von den anderen ignoriert.

### Beispiel:

— — — *Die Eingabe von*

```
<p>
Das Jahr beginnt mit den Monaten
<ol type=1>
<li>J&auml;nner
<li>Februar
</ol>
und endet mit den Monaten
<ol type=1 start=11>
<li>November
<li>Dezember
</ol>
```

— — — *bewirkt eine Darstellung wie*

Das Jahr beginnt mit den Monaten

1. Jänner
2. Februar

und endet mit den Monaten

11. November
12. Dezember

— — —

Da diese Parameter nicht von allen Web-Browsern unterstützt werden, kann man sich nicht darauf verlassen, dass die Bezeichnungen so erscheinen, wie man sich gewünscht hat. Deshalb sind Formulierungen wie "siehe Punkt C" zu vermeiden (der Punkt könnte ja mit 3 numeriert erscheinen); für solche Verweise sollte man ohnehin

besser →Hypertext-Verweise und →Sprungmarken verwenden. Wenn die Bezeichnung wichtig ist und sichergestellt werden soll, dass sie auf *allen* Web-Brwosern richtig erscheint, muss man deshalb →Definitions-Listen mit expliziter Angabe der Bezeichnungen verwenden.

**Beispiel:**

– – – *Statt*

```
<p>
Toto-Tips:
<ol type=1>
<li>erste Mannschaft gewinnt
<li>zweite Mannschaft gewinnt
<li type=A value=X>Spiel endet unentschieden
</ol>
```

– – – *verwendet man besser*

```
<p>
Toto-Tips:
<dl compact>
<dt>1
<dd>erste Mannschaft gewinnt
<dt>2
<dd>zweite Mannschaft gewinnt
<dt>X
<dd>Spiel endet unentschieden
</dl>
```

– – – *und erhält dann eine Darstellung wie*

```
Toto-Tips:
  1  erste Mannschaft gewinnt
  2  zweite Mannschaft gewinnt
  X  Spiel endet unentschieden
```

– – –

## 5.9 Frames <frameset> <frame> <noframes>

In →HTML 4 ist vorgesehen, dass man anstelle von einfachen Web-Seiten mit dem normalen Aufbau von →Head und Body, die wie gewohnt im Bildschirmfenster dargestellt werden, auch kompliziertere, mehrteilige Bildschirmstrukturen definieren kann, die als "Frames" (Rahmen) bezeichnet werden.

Frames werden von den meisten neueren Web-Browsern unterstützt, aber von vielen älteren Browser-Versionen noch nicht oder nur mit einem sehr unbequemen Benutzer-Interface. Auch die meisten Analyse-Tools und →Suchmaschinen ignorieren die Frames und verwenden nur die in <noframes> angegebene Information.

Bei der Definition von Frames hat das HTML-File nicht den →normalen Aufbau mit Head und Body sondern einen davon abweichenden Aufbau in der folgenden Form:

```
<html>
<head>
    <title>Titel des Files</title>
    ...
</head>
<frameset ... >
    <frame src="xxx.html" name="xxx" >
    <frame src="yyy.html" name="yyy" >
    ...
<noframes>
    <body>
    ...
    Information im normalen HTML-Format
    oder zumindest Links auf xxx.html und yyy.html
    ...
    </body>
</noframes>
</frameset>
</html>
```

Im <frameset>-Befehl wird mit Parametern wie z.B. `rows="20%,80%"` oder `cols="33%,33%,33%"` angegeben, wieviel Platz die darin enthaltenen Frames übereinander bzw. nebeneinander bekommen sollen.

Innerhalb von <frameset> wird mittels <frame>-Befehlen und eventuellen weiteren <frameset>-Befehlen angegeben, welche Teil-Informationen wo innerhalb des aktuellen Bildschirmfensters oder in einem anderen Fenster dargestellt werden sollen.

Bei Hypertext-Sprüngen von einem Frame zu einer neuen Information in einem anderen Frame muss im →<a href>-Befehl mit `target=` der Name des Ziel-Frame angegeben werden; bei `target="_top"` wird die neue Information im vollen Bildschirmfenster dargestellt.

Die Frameset- und Frame-Angaben werden von manchen Web-Browsern befolgt und von den anderen sowie von den meisten →Suchmaschinen ignoriert.

Innerhalb von <noframes> *muss* deshalb die gleiche Information nochmals in normaler HTML-Schreibweise von <body> bis </body> angegeben werden - entweder

die Summe der in den Frames enthaltenen Teilinformationen, oder zumindest eine Start-Seite oder ein Inhaltsverzeichnis mit →Hypertext-Verweisen auf die restlichen Teilinformationen. Hier nur eine lakonische Feststellung wie "Ihr Browser versteht keine Frame-Befehle" einzufügen, wäre nicht zielführend.

Eine typische Anwendung von Frames ist z.B. eine Zweiteilung des Bildschirms in ein kleines Fenster mit einer Knopf-Leiste oder einem Menü für die Auswahl der Informationen und ein großes Fenster für das Lesen der ausgewählten Informationen. Für solche Effekte wäre freilich die Verwendung von →Navigationshilfen mit <link> günstiger, die wird aber leider auch nicht von allen Web-Browsern voll unterstützt.

Generell ist von der Verwendung von Frames eher →abzuraten.

#### Warum Sie Frames vermeiden sollten

Das →World-Wide Web verdankt seinen Siegeszug vor allem seiner auch für ungeübte Benutzer leichten und einfachen Bedienbarkeit. Die meisten Benutzer kommen mit nur 4 Funktionen aus, die bei manchen Programmen – einer alten Konvention folgend - einfach den 4 Pfeiltasten zugeordnet sind:

1. Vorblättern in der Information  
(Tiefpfeil oder Page-Down oder Mausklick im Scrollbar),
2. Zurückblättern in der Information  
(Hochpfeil oder Page-Up oder Mausklick im Scrollbar),
3. Sprung zu einer neuen Information  
(Rechtspfeil oder Return-Taste oder Mausklick auf ein Link),
4. Rückkehr zur alten Information  
(Linkspfeil oder Escape-Taste oder Mausklick auf Back).

Bei der Verwendung von Frames gilt dieses einfache und für die meisten Benutzer gewohnte Bedienungsprinzip nicht mehr, sondern die Navigation wird mehrdimensional und viel komplizierter. Statt des einfachen, eindimensionalen "Vor und Zurück" zwischen alter und neuer Information braucht der Benutzer bei Frames eine verwirrende Fülle von Möglichkeiten, in den verschiedenen Teilbereichen verschieden weit vor oder zurück zu gehen. Manche Web-Browser bieten dafür verschiedene Back-Buttons oder zusätzliche Pull-Down-Menüs an, die von den Benutzern freilich erst gefunden werden müssen.

Noch ärger wird die Verwirrung, wenn innerhalb eines Frame auf eine Information gesprungen wird, die wiederum aus mehreren Frames besteht, oder wenn die Information auf mehrere Fenster verteilt ist, die teilweise gar nicht mehr sichtbar (weil durch andere Fenster verdeckt) sind.

Außerdem verbraucht jedes neue Fenster zusätzliche Ressourcen am →Client-Rechner, der dadurch unbrauchbar langsam werden oder sogar "abstürzen" kann. Andererseits ist innerhalb *eines* Bildschirmfensters meist nicht genügend Platz, um alle Teilinformationen nebeneinander lesen zu können. Das gleiche gilt für das Ausdrucken der Informationen auf Papier.

Es ist leider auch nur schwer möglich, eine über mehrere Frames verteilte Information später wiederzufinden. Ein Frameset, das aus  $n$  Frames besteht, müßte durch eine

Kombination von  $(n+1) \rightarrow$ URLs beschrieben werden: ein URL für das Frameset und  $n$  URLs für die momentanen Inhalte der  $n$  Frames.  $\rightarrow$ Bookmarks, Browser-History und  $\rightarrow$ Suchhilfen bauen aber immer nur auf einzelnen URLs auf. Man erhält deshalb entweder nur den Urzustand des Frameset (bevor der Inhalt durch "Weiterklicken" verändert wurde) oder nur den Inhalt, der in einem einzigen Frame enthalten war (ohne das Frameset und ohne den Inhalt der anderen Frames).

Auf Grund dieser Schwierigkeiten ist für das Konzept und die Gestaltung von Frames durch den Autor ein erfahrener Fachmann für benutzerfreundliche Mensch-Maschinen-Interfaces notwendig, und für die Verwendung der Frames durch die Benutzer eine entsprechende genauere Anwenderschulung.

## 5.10 Navigationshilfen <link>

Der Link-Befehl dient allgemein dazu, die Zusammengehörigkeit oder Verbindung (englisch link) einer Web-Page mit anderen Web-Pages oder Personen oder Informationen anzugeben. Diese Informationen werden vom Web-Browser dann in einer geeigneten Form verwendet, z.B. innerhalb von Befehlen wie "sende eine E-Mail an den Autor", oder in eigenen Buttons oder Menüs *außerhalb* des normalen Textfeldes ( $\rightarrow$ Body) angezeigt - so ähnlich wie bei  $\rightarrow$ Frames. Die Link-Befehle müssen deshalb stets im  $\rightarrow$ Head des HTML-Files stehen.

Innerhalb des Linkbefehls ist entweder mit **rel=** anzugeben, zu welcher anderen Information man von der aktuellen Web-Seite springen kann, oder es ist mit **rev=** anzugeben, von welcher anderen Information oder von welcher Person die aktuelle Web-Seite stammt. Mit **href=** ist der  $\rightarrow$ URL dieser anderen Information anzugeben.

Die wichtigsten Link-Befehle sind:

```
<link rev=made href="mailto:user@host.domain">
```

für die  $\rightarrow$ Mail-Adresse des Autors

```
<link rel=stylesheet href="URL" type="...">
```

für ein  $\rightarrow$ Stylesheet-File

```
<link rel=... href="URL">
```

für verschiedene Navigationshilfen

Mit **rel=** ist hier ein Schlüsselwort anzugeben, das die Art der Information angibt, auf die verwiesen wird. Die wichtigsten Möglichkeiten sind: **up** oder **top**, **begin** oder **first**, **previous**, **next**, **end** oder **last**, **toc** (table of contents), **index**, **glossary**, **author**, **copyright**, **help**.

```
<link rel=print href="URL" type="...">
```

für ein druckfertiges File dieser Information, mit Angabe des Filetyps (z.B. Post-Script oder RTF), damit das entsprechende Druck- oder Viewer-Programm vom Web-Browser gestartet werden kann (Helper-Application, Plugin-Programm).

Andere Navigationshilfen wie die Rückkehr zur Start-Seite des Benutzers ("home") oder zu der vom Benutzer zuletzt gelesenen Web-Seite ("back") hängen nicht vom Autor einer Web-Seite sondern nur vom Leser ab und werden daher nur programmintern im Web-Browser gespeichert und von diesem entsprechend angezeigt. Es gibt keinen HTML-Befehl, der den Back- oder Home-Button des Browsers simuliert, und selbst wenn man eine programmtechnische Lösung dafür findet, würde man damit

den History-Mechanismus des Web-Browsers und die für den Benutzer gewohnte Bedeutung der Back- und Forward- und Home-Buttons in seinem Web-Browser dreh-einanderbringen und den Benutzer damit verwirren statt ihm zu helfen.

### Beispiel:

Das 3. Kapitel (File `hein3.html`) der HTML-Einführung enthält die folgenden Link-Befehle:

```
<link rev=made      href="mailto:partl@mail.boku.ac.at">
<link rel=toc       href="hein.html#inhalt">
<link rel=index     href="hein.html#index">
<link rel=glossary  href="heinwas.html">
<link rel=copyright href="hein.html#copy">
<link rel=help      href="hein.html#vorwort">
<link rel=up        href="hein.html">
<link rel=previous  href="hein2.html">
<link rel=next      href="hein4.html">
<link rel=print     href="html Einf.ps" type="application/postscript">
```

- - -

Manche Browser zeigen diese Links als zusätzliche Buttons oder Menü-Punkte neben den normalen Browser-Befehlen an, andere Browser zeigen sie in eigenen kleinen Fenstern oder Button-Leisten oder Pull-Down-Menüs über oder neben dem Text-Fenster an.

Gegenüber →Frames oder sonstigen →Hypertext-Links, die vom Autor einer jeden Web-Seite separat und daher meist verschieden definiert werden, haben die mit diesen genormten Link-Befehlen definierten Verbindungen und Navigationshilfen den großen Vorteil, dass sie vom jeweiligen Web-Browser für *alle* vom Benutzer gelesenen Web-Seiten immer im gleichen, für den Benutzer gewohnten Format dargestellt werden und die Navigation im World-Wide Web für den Benutzer dadurch wesentlich übersichtlicher und einfacher wird.

Leider werden die Link-Navigationshilfen von vielen Web-Browsen derzeit noch nicht voll unterstützt. Trotzdem solle man diese Angaben schon jetzt vorsorglich in allen HTML-Files einfügen, denn sie können nicht schaden und sind auch für menschliche Leser der HTML-Files ("view source") nützlich.

## 5.11 Schlagwörter für Suchhilfen eintragen <meta>

Die meisten →Suchmaschinen können automatisch aus den im HTML-File vorkommenden Wörtern und aus der mit den HTML-Befehlen angegebenen Bedeutung dieser Wörter (z.B. Überschriften) "erraten", welche Wörter die Stichworte sind, die den Inhalt der Web-Page richtig wiedergeben.

Manchmal kann es aber sinnvoll sein, zusätzlich aussagekräftige, aber eventuell so nicht im Text vorkommende **Schlagwörter** oder eine über den →<title> hinausgehende **Beschreibung** der Web-Page anzugeben.

Dafür (und für noch ein paar andere spezielle Zwecke, die mit dem Protokoll →HTTP zusammenhängen) kann man innerhalb des →<head> einen oder mehrere <meta>-Befehle angeben:

```
<meta name=keywords content="xxx, yyy, zzz">
```

für die Angabe von Schlagworten

```
<meta name=description content="...">
```

für die Angabe einer *kurzen* Beschreibung

Ob innerhalb des "content" →Entities oder nur reine ASCII-Zeichen richtig funktionieren, und überhaupt, welche Angaben in welchem Format von einer Suchmaschine verarbeitet werden, wird meist in der Online-Hilfe der jeweiligen Suchmaschine angegeben.

Die meisten Suchmaschinen sind intelligent genug programmiert, dass sie eine unrichtige oder übertriebene Beschlagwortung, mit der manche Autoren eine höhere Priorität für ihre Web-Page bei der Suche erschleichen wollen, erkennen und entsprechend "bestrafen". Auch hier gilt also: "Lügen haben kurze Beine".

## 5.12 Interaktion mit dem Benutzer

### Mehr Leben ins World Wide Web!

- →Aktionen am Server (CGI, SSI, ASP, PHP)
- →Zugriffe zählen
- →Formulare <form>
- →Image-Maps <map> usemap
- →Aktionen am Client (Java, JavaScript, Active-X, DHTML, XSL)
- →Java-Applets <applet> <param> <object>
- →JavaScript <script> <noscript>
- →Paßwort-Schutz und Sicherheit (SSL, https)
- →Dynamische Web-Pages und Datenbanken
- →Electronic Mail (mailto)

Hier finden Sie nur kurze Hinweise auf diese Möglichkeiten für "Fortgeschrittene". Genauere Informationen müssen Sie der Dokumentation des jeweiligen →WWW-Servers und den →Referenzen entnehmen.

### 5.12.1 Aktionen am Server (CGI, SSI, ASP, PHP)

→CGI (Common Gateway Interface) ist die im Protokoll HTTP vorgesehene Schnittstelle zu Programmen, die am →Web-Server installiert sind. Ein CGI-Programm oder eine CGI-Prozedur ist ein Computer-Programm, das auf dem →WWW-Server läuft und die Ausgabe wie ein HTML-File an den →Client sendet.

Im Gegensatz zu HTML-Files, die einen fixen Inhalt haben, kann man damit Informationen verteilen, die sich laufend ändern oder die von Benutzer-Eingaben abhängen.

Der Aufruf der CGI-Prozedur erfolgt in einem →Hypertext-Link oder einem →Formular mit einem →URL der Form

```
http://hostname/cgi-directory/filename
```

oder

```
http://hostname/cgi-directory/filename?parameterliste
```

#### Beispiel:

— — — *Eine einfache Unix-Shell-Prozedur wie*

```
#!/bin/sh
echo "Content-Type: text/html"
echo ""
echo "<html><head><title>Datum</title></head>"
echo "<body><p>Today is "
/bin/date
echo "</body></html>"
exit
```

— — — *sendet z.B. den folgenden HTML-Code an den Client*

```
<html><head><title>Datum</title></head>
<body><p>Today is
Thu Mar 8 19:30:00 MET 2001
</body></html>
```

— — — *und das bewirkt eine Darstellung wie*

```
Today is Thu Mar 8 19:30:00 MET 2001
```

— — —

Eine etwas ausführlichere Version einer solchen Datumsanzeige finden Sie an der →BOKU Wien.

CGI-Programme und Prozeduren können in beliebigen Sprachen geschrieben werden (Shell-Scripts, Perl-Scripts, C-Programme, Java-Programme u.a.). Die Detail-Informationen dazu finden Sie in der Dokumentation des jeweiligen →WWW-Servers und in den →Referenzen, siehe auch die Hinweise von →Stefan Münz zu CGI-Programmen und Perl.

Wenn das CGI-Programm in →Java geschrieben wird und wenn der Web-Server es unterstützt, dann kann man dieses Java-Programm als sogenanntes →Java-Servlet

einrichten, das dann besonders effizient innerhalb des Web-Servers läuft. Mehr darüber finden Sie in der →Java-Einführung und in den dort angeführten Referenzen.

Andere Möglichkeiten, nicht ein fixes HTML-File sondern eine zur Aufrufzeit dynamisch generierte bzw. modifizierte HTML-Seite auszugeben, sind

- →Server-Side Includes (SSI, File-Extension .shtml)
- →Active Server Pages (ASP, File-Extension .asp)
- →Java Server Pages (ASP, File-Extension .jsp)
- →PHP (File-Extension .php oder .php3)
- dynamische Web-Pages (siehe das Kapitel über →Datenbanken)

Ob und wie das funktioniert, hängt von der verwendeten Web-Server-Software ab und ist in deren Dokumentation beschrieben.

### Beispiel

Hier ein Beispiel für ein →PHP-File, das eine →Datenbank-Abfrage in einen HTML-Text einbettet:

– – – *Das PHP-File am Server*

```
<html>
<head>
<title>freie Plaetze</title>
</head>
<body>
<p>
Im <b>HTML-Kurs</b> sind noch

<?php
    $con = ocilogon ('KursDB', 'user', 'password');
    $sql = "SELECT frei FROM kurse WHERE titel='HTML-Kurs' " ;
    $stmt = ociparse ($con, $sql);
    ociexecute ($stmt);
    while ( ocifetch ($stmt) ) {
        $frei = ocireresult ($stmt, 'frei');
        print($frei);
    }
    ocilogoff ($con);
?>

Pl&auml;tze frei.
</body>
</html>
```

– – – *sendet z.B. den folgenden HTML-Code an den Client*

```

<html>
<head>
<title>freie Plaetze</title>
</head>
<body>
<p>
Im <b>HTML-Kurs</b> sind noch
3
Pl&auml;tze frei.
</body>
</html>

```

– – – und das bewirkt eine Darstellung wie

Im **HTML-Kurs** sind noch 3 Plätze frei.

– – –

Für vollständige Informationen über PHP wird auf die →Referenzen verwiesen.

### 5.12.2 Zugriffe zählen

Sie als Autor möchten wahrscheinlich wissen, wie oft Ihre Informationen von Interessenten in aller Welt gelesen werden.

Relativ leicht können Sie erfahren, wie oft auf Ihr HTML-File am →WWW-Server zugegriffen wurde. Die meisten WWW-Server führen ein "Log-File", in dem alle Zugriffe protokolliert werden, und Sie können mit einem einfachen Unix-Shell-Script zählen, wie oft Ihr Filename darin vorkommt. Beispiel:

```

#!/bin/sh
url="$1"
logfile=/usr/local/apache/logs/access_log
n='grep -i "$url" $logfile | wc -l'
echo "$n Zugriffe auf $url"
exit

```

Außerdem finden Sie im Log-File auch die Information, von welchen →Client-Rechnern auf Ihr File zugegriffen wurde (aber im allgemeinen nicht, von welchen Personen).

Damit zählen Sie aber nur die Anzahl der File-Übertragungen und nicht, wie oft die Information tatsächlich gelesen wurde. Schuld daran sind die sehr nützlichen sogenannten **Cache-Speicher**, die zur Entlastung der Netzverbindungen notwendig sind:

Die meisten →Web-Browser haben einen lokalen Cache-Speicher. Wenn z.B. ein Benutzer in Berlin zehn mal die in Wien gespeicherte HTML-Einführung liest, wird sie nur beim ersten Mal von Wien nach Berlin übertragen und, solange der Platz ausreicht, in seinem Cache-Speicher abgelegt. Bei allen weiteren Zugriffen wird das File nicht nochmals von Wien nach Berlin übertragen, sondern viel einfacher und schneller aus dem lokalen Cache geholt (außer der File-Inhalt wurde inzwischen erneuert).

Viele Institutionen verwenden auch globale →Cache-Server ("Proxy"), die dann für alle Zugriffe aus diesem Bereich wirken. Wenn z.B. tausend Benutzer in Berlin die in Wien gespeicherte HTML-Einführung lesen, wird sie nur beim ersten Mal von Wien nach Berlin übertragen und eine Kopie am Berliner Cache-Server abgelegt (solange der Speicherplatz ausreicht). Bei allen weiteren Zugriffen wird das File nicht nochmals von Wien nach Berlin übertragen, sondern viel schneller aus dem Berliner Cache-Server geholt (außer der File-Inhalt wurde inzwischen erneuert).

Die Zahl, wie oft Ihr File tatsächlich gelesen wurde, ist im allgemeinen also wesentlich höher als die am WWW-Server gezählte Zahl der File-Übertragungen.

Manche Autoren wollen die Zahl der Zugriffe auch innerhalb des HTML-Files speichern. Dies ist zwar technisch möglich (z.B. mit Hilfe einer →CGI-Prozedur), aber *nicht* empfehlenswert, weil das eine zusätzliche Netz- und Server-Belastung bedeutet und weil die Anzahl der File-Übertragungen meist ohnehin nur für den Autor der Information und den Verwalter des WWW-Servers, aber nicht für die Leser interessant ist.

### 5.12.3 Formulare <form>

Der Inhalt von HTML-Files wird jeweils vom →WWW-Server zum Benutzer am →Client gesendet. Mit Hilfe von Formularen, die in HTML-Files stehen, können in der Gegenrichtung die Benutzer bestimmte Informationen am Client eingeben und an den WWW-Server senden. Diese Daten werden meistens von einer auf diesem Server laufenden →CGI-Prozedur verarbeitet.

#### Beispiel:

— — — *Die Eingabe von*

```
<form method="get"
action="http://www.boku.ac.at/cgi-bin/hein-get">
<b>Anmeldung</b> zur Geburtstagsfeier am 8. M&auml;r;z
<p>
<input type="radio" name="kommt" value="ja" checked>
Ich komme.
<br>
<input type="radio" name="kommt" value="nein">
Ich komme nicht.
<p>
Name:
<input name="wer" size="40" maxlength="512">
<p>
Telefonnummer:
<input name="tel" size="20" maxlength="512">
<p>
<input type="submit" value="Anmeldung absenden">
</form>
```

— — — *bewirkt eine Darstellung wie*

**Anmeldung** zur Geburtstagsfeier am 8. März

Ich komme.  
 Ich komme nicht.  
Name:   
Telefonnummer:

- - -

*Anmerkung: Die in diesem Beispiel verwendete CGI-Prozedur dient nur für Tests durch die Leser dieser Einführung. Wenn Sie dieses Formular ausfüllen und absenden, bekommen Sie eine kurze Test-Antwort, aber (leider) keine wirkliche Einladung zum "Heurigen".*

Eine "echte" Anmeldungs-Prozedur würde die Zu- und Absagen mit Name und Telefonnummer in einer Datenbank speichern und dann die Bestätigung an den Client senden.

Das Senden einer solchen Antwort ist *immer* notwendig, denn der Benutzer muss auf seinem Bildschirm erkennen können, dass das "Anklicken" des Submit-Knopfes funktioniert hat ("Feedback"). Im einfachsten Fall genügt eine kurze Meldung, dass die Eingabe verarbeitet wird, und eventuell ein Hinweis, dass der Benutzer mit der Back-Taste oder dem Back-Befehl seines Browsers zur vorherigen Information zurückkehren und weiterarbeiten kann.

Bei der Gestaltung von Formularen müssen die Normen und Konventionen für Benutzerschnittstellen und die Gewohnheiten und Erwartungen der Benutzer berücksichtigt werden.

Für genauere Informationen über die vom Server unterstützten Übertragungsmethoden ("get" oder "post") und die Übergabe der Eingabedaten an die CGI-Prozedur wird auf die Dokumentation des jeweiligen →WWW-Servers und auf die →Referenzen verwiesen. Die CGI-Prozeduren müssen so geschrieben werden, dass Ihre Ausführung kein Sicherheitsrisiko für den Server-Computer darstellen kann, egal was für eventuell seltsame Eingaben von den Benutzern kommen.

Wenn Sie keine Möglichkeit haben, ein CGI-Programm auf dem von Ihnen verwendeten →WWW-Server zu installieren, bietet sich als Alternative die Verwendung von →Electronic Mail an.

Manche (wenige) Web-Browser bieten die Möglichkeit, Formulare per E-Mail zu verarbeiten. Dazu würden Sie im Formular mit

```
<form action="mailto:user@host.domain" enctype="text/plain">
```

Ihre eigene Mail-Adresse angeben, und jedesmal, wenn ein Leser dann den Submit-Knopf "drückt", bekommen Sie den Formularinhalt als lesbaren Text per E-Mail in Ihre Mailbox gesendet und können ihn dann händisch oder mit einem auf Ihrem Computer laufenden Programm verarbeiten.

Allerdings werden solche Mailto-Formulare von sehr vielen Web-Browsern *nicht* unterstützt. Diese Methode kann daher nur innerhalb von kleinen, geschlossenen Benutzergruppen eingesetzt werden, bei denen man sicher ist, dass sie alle eine

dafür geeignete Browser-Version verwenden, aber *nicht* für allgemein verwendbare Anwendungen.

Deshalb sollte man, wenn man keine CGI-Programme verwenden kann, lieber gar keine Formulare verwenden sondern ein →normales Mailto-Link, bei dem die Benutzer den Text "formlos" schreiben und per E-Mail absenden können. Diese Möglichkeit steht dann *allen* Internet-Benutzern offen.

**Beispiel:**

```
<p>
Bitte senden Sie Ihre Anmeldung per
<a href="mailto:user@host.domain">E-Mail
an user@host.domain</a>
oder per Telefax an ...
```

- - -

#### 5.12.4 Image-Maps (ismap <map> usemap)

Image-Maps (Bild-Karten) erlauben die →Verzweigung zu bestimmten Informationen durch die Auswahl ("Anklicken") von bestimmten Regionen innerhalb eines →Inline-Bildes.

Dafür gibt es zwei verschiedene Verfahren:

- In →HTML 2 sind nur →Server-seitige Image-Maps (ismap) vorgesehen, die mit einer →CGI-Prozedur am Server verarbeitet werden. Dies ist die ältere und etwas kompliziertere Methode.
- Seit →HTML 3.2 sind auch →Client-seitige Image-Maps (<map> und usemap) vorgesehen, die direkt im HTML-File spezifiziert und vom Web-Browser verarbeitet werden. Dies ist die neuere Methode, sie ist einfacher zu realisieren und wird von allen neueren Web-Browsern unterstützt.

In →HTML 4 wird empfohlen, in Zukunft nur mehr →Client-seitige und nicht mehr Server-seitige Image-Maps in HTML-Files einzubauen.

#### Server-seitige Image-Maps (ismap)

Bei Server-seitigen Image-Maps muss die Verarbeitung durch eine auf dem →WWW-Server laufende →CGI-Prozedur erfolgen. Diese Möglichkeit war bereits in →HTML 2 vorgesehen und wird von allen grafikfähigen Web-Browsern unterstützt.

Im HTML-File ist eine Kombination aus einem →<a href>-Befehl für die CGI-Prozedur und einem →<img>-Befehl für das Bild mit der Angabe "ismap" anzugeben.

**Beispiel:**

- - - *Die Eingabe von*

```
<a href="http://www.boku.ac.at/cgi-bin/hein-map">
</a>
```

--- bewirkt eine Darstellung wie



und je nachdem, wo in diesem Bild Sie "klicken", bekommen Sie eine entsprechende Antwort von der CGI-Prozedur.

---

Für genauere Informationen über das Zusammenspiel zwischen Bild und CGI-Prozedur und über Hilfsprogramme zu deren Erstellung wird auf die →Referenzen verwiesen.

### Client-seitige Image-Maps <map> usemap

Bei Client-seitigen Image-Maps erfolgt die Verarbeitung durch den →Web-Browser auf Grund von entsprechenden Angaben im HTML-File, und es ist keine →CGI-Prozedur notwendig. Diese Möglichkeit ist seit →HTML 3.2 vorgesehen und wird von allen neueren Web-Browsern unterstützt.

Die Spezifikation, welche Teile des Bildes den Sprung zu welchem →Hypertext-Link bewirken sollen, erfolgt mit Hilfe von <map> entweder im selben HTML-File wie die Verwendung des Bildes oder in einem separaten File, das gemeinsam mit dem Bild abgespeichert ist.

Im →<img> oder →<object>-Befehl des Bildes wird dann mit "usemap" auf diese Map-Spezifikation verwiesen.

Da nicht alle Benutzer die Darstellung von Bildern eingeschaltet haben und da manche Web-Browser und viele →Suchmaschinen solche Image-Maps nicht richtig verarbeiten, muss man bei einer Image-Map zusätzlich ein "normales" Link mit →<a href> angeben, das zu einer →Liste von Hypertext-Links in einfacher Textform führt. Damit wird sichergestellt, dass *alle* Benutzer und Suchhilfen die betreffenden Informationen erreichen können.

#### Beispiel:

--- Die Eingabe von

```
<map name="map3">
<area shape=rect coords="0,0,89,15" href="#oben" alt="oben">
<area shape=rect coords="0,16,89,31" href="#mitte" alt="mitte">
<area shape=rect coords="0,32,89,47" href="#unten" alt="unten">
</map>
<p>Klicken Sie auf eines der drei Felder der Fahne
<a href="#ohnemap">
</a>
```

--- bewirkt die Ausgabe von



*Je nachdem, wo Sie in diesem Bild "klicken", gelangen Sie zu einer der folgenden vier Stellen:*

1. Sie haben das obere rote Feld ausgewählt.
2. Sie haben das mittlere weiße Feld ausgewählt.
3. Sie haben das untere rote Feld ausgewählt.
4. Ihr Browser unterstützt Client-seitige Image-Maps nicht. Sie sind deshalb hier bei der Alternative "ohne Map" gelandet und können die drei Möglichkeiten nun mit normalen Hypertext-Links auswählen:

- →Informationen zum oberen Bereich
- →Informationen zum mittleren Bereich
- →Informationen zum unteren Bereich

*Die Eingabe für die im vierten Punkt beschriebene Alternative hat einen Aufbau der folgenden Form:*

```
<a name="ohnemap">Ihr Browser ... </a>
Sie ... k&ouml;nne ... nun ... ausw&auml;hlen:
<ul>
<li><a href="#oben">Informationen zum oberen Bereich</a>
<li><a href="#mitte">Informationen zum mittleren Bereich</a>
<li><a href="#unten">Informationen zum unteren Bereich</a>
</ul>
```

---

Manche Web-Browser stellen `<map>` nicht als Bild sondern als eine Art Pull-down-Menü mit den in den ALT-Parametern angegebenen Texten dar. Die Angabe dieser Texte mit `alt=` in `<area>` ist deshalb zwingend vorgeschrieben.

### 5.12.5 Aktionen am Client (Java, JavaScript, Active-X, DHTML, XSL)

Im Gegensatz zu →CGI-Programmen und →Datenbanksystemen, die am →Server laufen, gibt es auch Möglichkeiten, Berechnungen oder sonstige Aktionen direkt am →Client ablaufen zu lassen, was in vielen Fällen effizienter ist. Allerdings sind dabei Sicherungsmechanismen notwendig, damit nicht irgendwelche fremde Personen oder Hacker Web-Pages so schreiben können, dass auf dem Client-Rechner (PC) des Lesers irgendwelche Aktionen ausgeführt werden, die der Leser gar nicht will.

Viele Benutzer haben wegen dieser Sicherheitsrisiken die Möglichkeiten von JavaScript und dergleichen in ihren Web-Browsern ausgeschaltet. Deswegen und weil diese Möglichkeiten nicht in allen Web-Browsern richtig funktionieren auch von Suchmaschinen nicht unterstützt werden, muss der Autor der Web-Page immer auch Alternativen in gewöhnlichem HTML vorsehen (mit `<noscript>` etc.).

Die wichtigsten Möglichkeiten sind:

- →Java-Applets
- →JavaScript

- andere Script-Sprachen (z.B. VBScript, nur im MS Internet-Explorer)
- Active-X: eine direkte Verbindung zwischen Web-Browser und Betriebssystem, nur im MS Internet Explorer
- Dynamic HTML (DHTML): ein Zusammenspiel zwischen →JavaScript und →Style Sheets
- →XSL extended Stylesheets

### 5.12.6 Java-Applets <applet> <param> <object>

Was ist Java?

- Java ist eine Insel in Indonesien.
- Java ist ein amerikanischer Ausdruck für starken Kaffee.
- Java ist eine moderne Programmiersprache.
- Java ist nicht →JavaScript.

Java ist eine **Programmiersprache**, ähnlich wie Pascal, Modula, C und C++.

Java ist **plattformunabhängig**. Wenn Sie ein Java-Programm auf einem Computer schreiben und kompilieren, dann läuft es unverändert auf allen Arten von Computern, egal ob Windows-PC, Macintosh, Unix-Rechner oder TV-Settopbox.

Java ist **objektorientiert** und eignet sich daher auch sehr gut für komplexe Anwendungen, die aus vielen Einzelteilen bestehen, sowie für graphische User-Interfaces.

Java hat eine umfangreiche **Klassenbibliothek**, in der Sie viele Klassen für die verschiedensten Zwecke schon fertig vorfinden und damit auch komplizierte Aufgaben mit wenig Aufwand realisieren können (z.B. Bildbearbeitung, dynamisch wachsende Listen, Sortieren, Multithreading, E-Mail, Socket-Verbindungen über das Internet, Datenbankoperationen u.v.a.).

Java hat zahlreiche **Sicherheitsmechanismen** eingebaut, die vor typischen Programmfehlern schützen. So können Java-Programme zum Beispiel keine "allgemeine Schutzverletzung" bewirken, weil es die Sprachelemente, die dazu führen können (Pointer-Arithmetik u.a.), in Java gar nicht gibt.

Aus allen diesen Gründen eignet sich Java nicht nur für "normale" Computer-Programme sondern auch besonders gut für Anwendungen, die über das Internet verbreitet werden. Neben normalen Java-**Applikationen** gibt es daher auch Java-**Applets**, die in Web-Pages eingebaut werden und dann bei jedem Benutzer innerhalb des Web-Browsers ablaufen.

Durch spezielle Sicherheitsvorkehrungen ist sichergestellt, dass diese Applets in den Web-Browser "eingesperrt" bleiben und keine bösen Nebenwirkungen auf den Rechner des Benutzers haben können: Applets können z.B. nicht auf die dort gespeicherten Dateien zugreifen und keine Systemfunktionen oder anderen Programme aufrufen.

Java ist eine neue und moderne Programmiersprache. Dieser Vorteil ist gleichzeitig auch ein Nachteil: Java ist so neu, dass seine Eigenschaften noch laufend erweitert

und verbessert werden. Seit 1995 gibt es Version 1.0, Anfang 1997 kam Version 1.1 heraus, Ende 1998 die Version 1.2 (auch Java 2 genannt). Die Hersteller der Web-Browser haben Mühe, mit dieser Entwicklung Schritt zu halten. Dies sollte sich aber in hoffentlich nicht zu ferner Zukunft bessern.

Ich bin davon überzeugt, dass Java in ein bis zwei Jahren die nötige Reife und Stabilität erreichen wird und nicht nur eine aufregende Gegenwart sondern auch eine gute und stabile Zukunft hat.

## Java-Applets in HTML

Innerhalb der Web-Page werden Applets mit dem HTML-Tag `<applet>` eingefügt und aufgerufen. Dies erfolgt nach dem folgenden Schema:

```
<applet code="Xxxx.class" width=nnn height=nnn>
<param name="xxx" value="yyy">
... Text ...
</applet>
```

Im **Applet**-Tag muss mit dem Parameter **code** der Name des Java-Applet angegeben werden (Class-File mit dem plattformunabhängigen Bytecode), und mit **width** und **height** die Breite und Höhe des Applet in Pixels.

In diesem Fall muss sich das Class-File im selben Directory am selben Server wie das HTML-File befinden. Wenn dies nicht der Fall ist, muss man mit einem zusätzlichen Parameter **codebase** den →URL des Directory angeben, aus dem das Class-File geladen werden soll. Beispiel:

```
<applet codebase="http://www.boku.ac.at/javaeinf/"
       code="HelloApp.class" width=300 height=100>
```

Falls das Class-File kein einzelnes File ist sondern in einem komprimierten Archiv (ZIP- oder JAR-File) enthalten ist, ist der Name dieses Archiv-Files mit dem Parameter **archive** anzugeben:

```
<applet codebase="http://hostname/dirname/"
       archive="xxxxx.jar" code="Xxxxx.class"
       width=300 height=100>
```

Mit einem oder mehreren **Param**-Tags können Parameter an das Applet übergeben werden.

Der (abgesehen von den Param-Tags) zwischen `<applet>` und `</applet>` stehende **Text** wird von den Browser-Versionen, die Java nicht unterstützen, an Stelle des Java-Applet dargestellt. Dies kann beliebig formatierter HTML-Text sein, also z.B. auch Bilder enthalten.

In der neuen HTML-Norm HTML 4.0 ist vorgesehen, dass statt dem Tag `<applet>` der Tag `<object>` verwendet werden soll. Dies wird aber von dem meisten Browsern noch nicht unterstützt.

Ausführlichere Informationen über Java finden Sie in der →Java-Einführung und in den dort angeführten Referenzen sowie in der Newsgruppe →de.comp.lang.java

#### Beispiel: Lake-Applet

Hier ein Beispiel, wie Sie ein fertiges Java-Applet in Ihre Web-Page einbauen und mit <param>-Tags an Ihre Bedürfnisse anpassen können, das – inzwischen schon allzu populäre – Lake-Applet von →David Griffiths. Dieses Applet stellt ein Bild als Spiegelung in einem See dar. Das Class-File muss zu diesem Zweck in dasselbe Directory kopiert werden, wo sich auch Ihr HTML-File und Ihr Bild befindet, und mit den Parametern image und href wird angegeben, welches Bild sich im See spiegeln soll und was passieren soll, wenn der User auf das Bild klickt.

Das HTML-File sieht so aus:

```
<html>
<head>
<title>Badende Venus</title>
</head>
<body>
<h1 align=center>Lake Applet</h1>
<p align=center>
<applet code="lake.class" width=604 height=723>
<param name=image value="http://www.boku.ac.at/htmlleinf/big.jpg">
<param name=href value="http://www.boku.ac.at/htmlleinf/hein53.html#java">
  Ohne Java k&ouml;nnen Sie das Bild nur
  ohne Wasserspiegelung sehen:
  <p align=center>
  
</applet>
</body>
</html>
```

Wenn Ihr Web-Browser Java unterstützt, dann können Sie →hier ausprobieren, wie das aussieht.

#### 5.12.7 JavaScript, <script> <noscript>

Was ist JavaScript?

JavaScript ist etwas anderes als →Java.

JavaScript ist *keine* selbständige Programmiersprache, sondern eine Erweiterung von HTML durch ein paar Script-Befehle, die dann vom Web-Browser ausgeführt werden.

JavaScript ist *nicht* plattformunabhängig, sondern läuft nur in Netscape und Internet Explorer, und auch da nicht in allen Versionen in der gleichen Weise.

JavaScript verfügt *nicht* über die guten Sicherheitsvorkehrungen von Java und kann daher durchaus böse Nebenwirkungen auf den Benutzer haben. In noch stärkerem Maße gilt dies übrigens für Active-X und Active Desktop im Internet Explorer und in Windows 98.

## JavaScript in HTML

JavaScript wird mit dem HTML-Tag `<script>` direkt in das HTML-File eingebettet:

```
<script>
... JavaScript-Programm ...
</script>
<noscript>
... Text ...
</noscript>
```

Im `<noscript>`-Block wird in normaler HTML-Schreibweise die Information angegeben, die in der Web-Page erscheinen soll, falls der Benutzer JavaScript ausgeschaltet hat oder falls der Web-Browser oder die Suchmaschine JavaScript nicht unterstützt.

Wenn das JavaScript zum Beispiel dazu dient, Hilfstexte ("Tooltips") anzuzeigen, wenn die Maus an eine bestimmte Stelle zeigt, dann sollen dieselben Erklärungen im `<noscript>`-Block als normale HTML-Texte stehen. Oder wenn das JavaScript dazu dient, auf andere Web-Pages zu verzweigen, müssen im `<noscript>`-Block die gleichen Links als normale `<a href>`-Befehle erscheinen. Dies ist sowohl für Anwender als auch für Suchmaschinen wichtig, die sonst die Informationen in den Links gar nicht erreichen können.

### Online-Informationen über JavaScript

Ausführlichere Informationen über JavaScript finden Sie vor allem bei →Stefan Münz sowie in der Newsgruppe →de.comp.lang.javascript

### Soll ich Java oder JavaScript verwenden?

Wenn Sie nur ein paar Spezialeffekte in Ihre Web-Page einbauen wollen – z.B. dass sich ein Link ändert, wenn die Maus darüber fährt, oder dass die Dateneingabe in einem Formularfeld kontrolliert wird, – dann genügt **JavaScript**.

Wenn Sie in Ihrer Web-Page eine komfortable Benutzerführung oder Berechnungen oder Simulationen ablaufen lassen wollen, dann brauchen Sie **Java**. Java eignet sich außerdem nicht nur für →Applets und →Servlets sondern auch für eigenständige Anwendungen (Applikationen) als eine modernere und bequemere Alternative zu anderen Programmiersprachen wie Cobol, Fortran, Basic und C.

### 5.12.8 Paßwort-Schutz und Sicherheit (SSL, https)

Wenn man den Zugriff auf bestimmte Web-Pages auf bestimmte Personen oder Personengruppen einschränken will, dann soll man die dafür im →Web-Server vorgesehenen Funktionen verwenden. Alle Web-Server-Produkte erlauben es, den Zugriff auf Client-Rechner einer Domain (also z.B. nur für alle Benutzer innerhalb des Intranet der Firma oder Universität) zu beschränken oder den Zugriff von der Eingabe eines Username und Paßworts abhängig zu machen, also nur für bestimmte Personen verfügbar zu machen (z.B. nur für zahlende Kunden).

Solche Einschränkungen und Paßwort-Kontrollen dürfen grundsätzlich nur am Server erfolgen. Wenn man versucht, Einschränkungen oder Paßwort-Kontrollen mit →Applets oder →JavaScript am Client durchzuführen, können erfahrene Benutzer den Inhalt der Applets bzw. Scripts ansehen und den darin enthaltenen Schutz einfach umgehen. Man darf, bildlich gesprochen, niemals Schloß und Schlüssel gleichzeitig aus der Hand geben. Ein sicherer Schutz ist *nur* auf dem Server möglich.

Die von der Web-Server-Software und dem Protokoll HTTP unterstützte Paßwort-Abfrage hat außerdem den Vorteil, dass das vom Benutzer im Web-Browser eingegebene Paßwort verschlüsselt vom Client an den Server übertragen wird und daher gegen Ausspähen geschützt ist.

Falls andere wichtige oder sensible Daten zwischen Client und Server übertragen werden (z.B. nach Eintragen in einem →Formular an das →CGI-Programm), dann muss diese Übertragung verschlüsselt erfolgen. Zu diesem Zweck müssen sowohl der →Web-Server als auch der →Browser die automatische Verschlüsselung der Daten unterstützen. Dazu werden eine Internet-Verbindung mit SSL (secure socket layer) und das Protokoll https (secure HTTP) verwendet.

### 5.12.9 Dynamische Web-Pages und Datenbanken

Die meisten Firmen, Universitäten und Institutionen verwenden Workflow-Systeme oder Datenbanken für die Verwaltung von umfangreichen Informationen und Daten. Deshalb ist es in den letzten Jahren immer wichtiger geworden, über das →World-Wide Web nicht bloß einzelne Dokumente verfügbar zu machen, die in Form von eigenständigen HTML-Files →erstellt und mühsam →laufend aktualisiert werden müssen, sondern einen möglichst direkten Zugriff auf die jederzeit aktuellen Informationen und Daten in den Workflow-Systemen und Datenbanken zu ermöglichen.

Dies wird als "dynamische Web-Pages" bezeichnet und über Schnittstellen zwischen dem Web-Server und dem Workflow-System bzw. der Datenbank realisiert, oder über integrierte Server-Systeme, bei denen das Workflow-System bzw. das Datenbanksystem direkt von den Web-Browsern über das Protokoll →HTTP bzw. die →CGI-Schnittstelle angesprochen werden kann und das System die Ergebnisse im Format →HTML an den Web-Browser ausgibt. Beispiele dafür sind z.B. der im Workflow-System Lotus Notes integrierte Domino-Server oder der im Datenbanksystem Oracle integrierte Web-Server.

### 5.12.10 Electronic Mail (mailto)

Die Angabe eines →mailto-URL in einem →<a href>-Befehl gibt dem Leser die Möglichkeit, von seinem →Client aus eine kurze Nachricht per →E-Mail an eine bestimmte Mail-Adresse zu senden, z.B. an den →Autor der Web-Page.

Der Benutzer muss das Subject, den Text der Nachricht und seine Absender-Adresse am →Client eintippen, und er muss die Erlaubnis zum Absenden von Mail haben.

In →HTML 4.0 ist vorgesehen, dass man das "Subject" (den Betreff) in der Form <a href="mailto:mailadresse" title="subject">...</a> angeben kann. Dies wird allerdings derzeit noch von vielen Web-Browsern ignoriert, d.h. die E-Mail wird von denen zwar richtig zugestellt, aber eventuell nicht mit dem angegebenen Subject.

Es gibt ein paar Web-Browser, die stattdessen die nicht genormte Form <a href="mailto:mailadresse?subject">...</a> unterstützen. Diese Form sollte man aber *niemals* verwenden, denn sie führt bei allen anderen Web-Browsern dazu, dass die E-Mail *überhaupt nicht zugestellt* werden kann, weil sie eine durch das "Anhängsel" ungültige Mail-Adresse aufweist.

Ob das Absenden von E-Mail innerhalb des Web-Browsers funktioniert, hängt von der Konfiguration am →Client-Rechner ab. Außerdem sind "richtige" Mail-Programme für das Versenden von Electronic Mail im allgemeinen besser geeignet, insbesondere für wichtige oder längere Mail-Texte und für das Lesen von Antworten.

Deshalb und auch für den Fall, dass der Benutzer die Web-Page auf einem Drucker ausdruckt, sollte man die Mail-Adresse nicht nur "versteckt" im URL sondern auch sichtbar im Text angeben.

#### Beispiel:

-- Die Eingabe von

```
Wenn Ihnen meine HTML-Einführung gefüllt,  
sagen Sie es bitte Ihren Kollegen. <br>  
Wenn Sie darin Fehler finden, sagen Sie es bitte  
<a href="mailto:partl@mail.boku.ac.at">mir</a>.  
<br>  
Meine Mail-Adresse ist  
<a href="mailto:partl@mail.boku.ac.at">partl@mail.boku.ac.at</a>.
```

-- bewirkt eine Darstellung wie

Wenn Ihnen meine HTML-Einführung gefällt, sagen Sie es bitte Ihren Kollegen.  
Wenn Sie darin Fehler finden, sagen Sie es bitte →mir.  
Meine Mail-Adresse ist →partl@mail.boku.ac.at.

--

---

## 6. Geschichte und Geschichten

---

**Es war einmal...**

*Ich schrieb diesen Überblick im Sommer 1995. Wenn Sie ihn lesen, kann er auch schon wieder Geschichte geworden sein, so rasant ist die Entwicklung der Technik gerade in diesem Bereich!*

- →Vom Elektronengehirn zum World Wide Web
- →Von der Textverarbeitung zur Hypertext Markup Language
- →Entwicklungen für die Zukunft

### 6.1 Vom Elektronengehirn zum World Wide Web

In den 40er-Jahren unseres Jahrhunderts wurden die ersten großen EDV-Anlagen gebaut, und in den 60er- und 70er-Jahren gehörten riesige Computer mit Lochkarten, Magnetbändern und Endlos-Printouts zu den Prestige-Objekten von Universitäten, Banken und großen Firmen, ehrfürchtig "Elektronengehirne" oder ironisch "Blechtrottel" genannt. In den 80er-Jahren zogen wesentlich kleinere "persönliche Computer" mit Textverarbeitung und Spiel-Programmen auf die Schreibtische und in die Privatwohnungen ein. Damit lösten bunte Bildschirme und Mäuse und intuitiv erfaßbare graphische Benutzeroberflächen die bis dahin nur für Wissenschaftler verständlichen Programmiersprachen ab.

Waren die ersten Computer noch in die Maschinenräume ihrer stolzen Besitzer eingesperrt, so trat bald die Notwendigkeit für eine Verbindung der Computer und eine Zusammenarbeit der Computer-Benutzer auf. Ende der 60er-Jahre wurde von einer Projektgruppe des amerikanischen Verteidigungsministeriums (ARPA) ein Computer-Netz konstruiert, das viele Subnetze ohne zentrale Kontrolle zu einem weltumspannenden Netzwerk verbinden kann: Das →Internet war geboren. Es verbreitete sich in den 70er- und 80er-Jahren vor allem im akademischen Bereich und seit Beginn der 90er-Jahre immer mehr auch im kommerziellen Bereich rund um den Erdball und ist derzeit das bei weitem umfangreichste Computer-Netzwerk der Erde.

Populär wurde das Internet 1993, als der US-Vizepräsident Al Gore den →"Information Super-Highway" zum nationalen Anliegen erklärte und bald auch die Politiker und die Massenmedien in den anderen Ländern zumindest verbale Unterstützung für diese Ideen signalisierten.

In "Das Wunder von Issaquah" erklärte damals George Gilder die Vorteile der Computer-Vernetzung mit dem folgenden Vergleich:

#### **Ein Auto im Dschungel**

Stellen Sie sich vor, wir finden mitten im Dschungel ein Auto. Wir werden bald herausfinden, dass es sich dabei um ein äußerst nützliches

Mehrzweck-Gerät handelt: Es bietet uns ein Dach gegen Regen, bequeme Sitze, helles Licht, eine Heizung und Klimaanlage, ein Radio mit Kassettenspieler, und eine Hupe zum Verjagen von wilden Tieren. Voll Bewunderung für alle diese Eigenschaften fragen wir gar nicht danach, zu welchen noch viel größeren Wundern das Auto fähig wäre, wenn wir es auf eine asphaltierte Straße brächten.

Wenn wir Computer oder PCs "nur" für die Berechnung von Zahlen und die Speicherung von Texten verwenden, sind sie für uns so nützlich wie das Auto im Dschungel. Das volle Potential ihrer möglichen Leistungen bieten sie uns erst, wenn wir sie über Computer-Netze mit anderen Computern verbinden.

Die ersten am →Internet verfügbaren Dienste waren noch recht primitiv: "Telnet" für den Zugriff auf Programme auf anderen Computern, "FTP" (File Transfer Protocol) für die Übertragung von Dateien auf andere Computer, "Electronic Mail" und "Usenet News" für den Austausch von Informationen.

Ende der 80er-Jahre wurden zwei Arten von Informations-Systemen entwickelt, die einen gegenüber diesen alten Services wesentlich bequemeren Zugriff auf weltweit verstreute Informationen ermöglichen: →"Gopher" und das →"World Wide Web" (WWW).

Das WWW, von Tim Berners-Lee im europäischen Kernforschungszentrum CERN entwickelt, wurde zunächst hauptsächlich von den dortigen Physikern für den Zugriff auf ihre Dokumentationen und Daten mit relativ einfachen, zeilenorientierten →"Web-Browsern" unter Unix und einer einfachen Hypertext-Sprache namens →"Hypertext Markup Language" (HTML) genutzt.

Populär wurde das WWW schlagartig 1993/94, als das National Center for Supercomputing Applications (NCSA) in den USA einen Web-Browser mit einer graphischen Benutzer-Oberfläche herausbrachte und allen Benutzern kostenlos zur Verfügung stellte: Mosaic, programmiert vom damaligen Studenten und späteren Netscape-Firmengründer Marc Andreessen. Damit steht das World Wide Web plötzlich mit modischen bunten Bildern und einfachen Maus-Klicks für jeden PC-Benutzer offen, und →WWW-Server mit mehr oder weniger interessanten Informationen sowie →Web-Browser mit mehr oder weniger guten Eigenschaften sprießen wie die sprichwörtlichen Schwammerln überall aus dem nahrhaften Boden der Informations-Gesellschaft. Rasch brachten auch kommerzielle Firmen wie Spry und Netscape, IBM und Silicon Graphics, AOL und Prodigy, und bald auch Microsoft ihre eigenen, zum Teil inkompatiblen WWW-Software-Produkte heraus. Alle paar Monate gibt es neue →Server- und →Client-Software und neue Ideen zu Erweiterungen von →HTML.

Die Zentrale der WWW-Entwicklung hat sich inzwischen vom CERN gelöst und wird nun von einem eigenen →W3-Consortium geleitet, dem unter anderem die Forschungseinrichtungen INRIA in Frankreich und MIT in den USA sowie einschlägige Software-Firmen wie z.B. Netscape und Microsoft angehören. Dort wird an HTML-Normen gearbeitet, die erweiterte Möglichkeiten für spezielle Darstellungsformen und Layout-Varianten bieten und die von verschiedenen Software-Firmen herausgebrachten →HTML-Varianten unter einen Hut bringen sollen. Auch an anderen

Orten wird an vielversprechenden Weiterentwicklungen gearbeitet, so zum Beispiel an  $\rightarrow$ Hyper-G,  $\rightarrow$ PDF,  $\rightarrow$ VRML und  $\rightarrow$ Java.

Es gehört zum guten Image jeder Universität und jeder Firma in den 90er-Jahren, im WWW präsent zu sein, zumindest mit ein paar bunten und werbewirksamen Bildern. Tausende Schüler und Studenten bemühen sich, ihre "cool Web pages" mit "hot links" zu anderen WWW-Servern aufzubauen und ihren Freunden in aller Welt zur Verfügung stellen. Es ist zu hoffen, dass auch die wirklich nützlichen Informationen im World Wide Web noch weiter zunehmen werden.

Die Menge der am WWW erreichbaren Informationen, Services, Texte, Bilder und Töne ist bereits so umfangreich und unüberschaubar geworden, dass das größte Problem darin liegt, eine nützliche Information aufzufinden. Deshalb sind  $\rightarrow$ Suchhilfen zu einem wichtigen Hilfsmittel geworden, und schon gibt es auch Hilfsmittel zum Auffinden von Suchhilfen. Ein möglicherweise zukunftsweisendes Forschungsprojekt stellt die Idee von "intelligenten Agenten" dar, die wie ein gutartiges Virus über das Internet ausschwärmen, um bestimmte Informationen auf allen angeschlossenen Computern zu suchen, und, wenn sie fündig geworden sind, die Ergebnisse an den Auftraggeber senden.

Jedenfalls sind Übung und Erfahrung – oder die Hilfe von geübten und erfahrenen Informations-Beratern – notwendig, um nicht wie eine Fliege in diesem weltweiten Spinnennetz hängen zu bleiben, sondern wie eine Spinne die nahrhaften Informations-Bissen darin zu ernten und zu nützen.

## 6.2 Von der Textverarbeitung zur Hypertext Markup Language

Die ersten Programme für die Ausgabe von Texten im Zeitalter der Lochkarten und Kettendrucker waren so konzipiert, dass die Angaben für die Formatierung der Texte vom Benutzer mit Hilfe spezieller Zeichenkombinationen direkt in den Text eingebettet wurden und dann bei der Ausgabe die entsprechenden Effekte wie Leerzeilen, Einrückungen, Unterstreichungen u.dgl. bewirkten. Dieses Prinzip wird als "Presentation-based Markup" bezeichnet.

Auf den "persönlichen Computern" wie Apple und PC, viele Jahre später, kam eine völlig neue Generation von Textverarbeitungs-Programmen heraus, die nach dem WYSIWYG-Prinzip (what you see is what you get) funktionieren, d.h. die Benutzer können das Layout mit der Maus direkt am Bildschirm "freihändig" einstellen.

Unbeirrt von der Popularität der WYSIWYG-Programme, entwickelte Prof. Donald Knuth an der Stanford-Universität Ende der 70er-Jahre ein computergestütztes Text-Satz-System namens "TeX" ( $\text{T}_{\text{E}}\text{X}$ ), bei dem die Layout-Elemente ebenfalls als Markup-Befehle innerhalb des Eingabe-Files und zunächst in einer vom Ausgabegerät unabhängigen Darstellung festgelegt werden, und das Druckbild dann mit Driver-Programmen auf verschiedenen Ausgabegeräten jeweils optimal ausgedruckt wird.

Leslie Lamport erweiterte dieses System Anfang der 80er-Jahre mit  $\rightarrow$ "LaTeX" ( $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ ) dahingehend, dass Inhalt und Form getrennt festgelegt werden: Der Inhalt wird nur mit einem logischen Markup versehen, d.h. mit Angaben, welche logische Bedeutung die Text-Elemente haben (z.B. Überschrift, Absatz, hervorgehobenes

Wort, Zitat, mathematische Formel, Gleichungsnummer, Liste, Listenelement, Tabelle, Tabellenspalte, Tabellenzeile usw.). In welchem Layout diese Elemente dargestellt werden (Schriftarten, Schriftgrößen, Abstände, Seitenformat usw.), wird separat davon in sogenannten Style-Files festgelegt. Dadurch kann derselbe Inhalt durch die Verwendung verschiedener Style-Files in verschiedenen Formaten ausgedruckt werden (z.B. als Skriptum, als Artikel in einer Zeitschrift und als Buch), und umgekehrt können mehrere Informationen durch die Verwendung desselben Style-Files automatisch in einem einheitlichen Layout und konsistenten Format ausgedruckt werden (z.B. alle Skripten eines Instituts, alle Briefe und alle Werbebroschüren einer Firma, oder alle Bücher eines Verlages).

Dieses Prinzip des "Content-based Markup" wurde Mitte der 80er-Jahre zur →"Standard Generalized Markup Language" SGML verallgemeinert, und die im World Wide Web verwendete →"Hypertext Markup Language" (HTML) baut auf diesem SGML-Standard auf.

So kommt es, dass ich mich beim Schreiben der vorliegenden →"HTML-Einführung" eng an die →"LaTeX-Kurzbeschreibung" anlehnen konnte, die ich 1987 gemeinsam mit Irene Hyna und Elisabeth Schlegl geschrieben und damals auf einem Listserver über das Bitnet und später auf FTP-Servern über das Internet veröffentlicht hatte.

### 6.3 Entwicklungen für die Zukunft

Hinweise auf Weiterentwicklungen und neue Trends ab 1999/2000 finden Sie im →nachfolgenden Kapitel.

---

## 7. Entwicklungen für die Zukunft

---

→HTML wird sicher noch länger die meistbenutzte Markup-Sprache im →WWW sein. Aber seit 1999/2000 entwickelt es sich in die im folgenden skizzierte Richtung weiter bzw. wird Schritt für Schritt durch die im folgenden angeführten Systeme und Markup-Sprachen ergänzt und teilweise ersetzt:

- →XML – die einfachere SGML
- →XHTML – die neue HTML
- →MathML für Mathematik
- →CML für Chemie
- →WAP und WML für Handys

Freilich werden alle diese neuen Entwicklungen derzeit von den meisten →Web-Browsern noch nicht (oder noch nicht vollständig) unterstützt.

### 7.1 XML – die einfachere SGML

**XML = Extensible Markup Language**

Eine Erklärung der Grundbegriffe finden Sie im →Glossar:

- →XML . →XML-Anwendung . →XSL

XML ist eine Metasprache zur Definition von Markup-Sprachen.

So wie →HTML mit →SGML definiert ist, so kann man mit →XML eigene Markup-Sprachen definieren, und künftige Versionen von HTML werden ebenfalls mit XML definiert werden (→XHTML).

So wie HTML festgelegt und normiert ist und daher für den weltweiten Austausch und die Übertragung und Verwendung von Web-Pages zwischen vielen verschiedenen Web-Servern und Web-Browsern geeignet ist, so kann man mit XML eigene Datei-Strukturen für verschiedene Zwecke definieren und normieren, die dann ebenfalls von vielen Personen mit vielen verschiedenen Programmen und auf vielen verschiedenen Rechnern verwendet werden können. Mit der Hilfe von →Style-Sheets können XML-Dokumente außerdem ebenfalls, so wie HTML-Files, von →Web-Browsern dargestellt und ausgedruckt werden.

Wofür kann man nun solche mit XML definierte Markup-Sprachen, sogenannte "XML-Applikationen", verwenden? Welchen Zweck kann es haben, mit XML solche Sprachen festzulegen und zu normieren? Welche Vorteile haben solche XML-Anwendungen gegenüber HTML oder anderen Datei-Formaten?

- Mit XML kann man die logische Bedeutung von Daten, Informationen und Texten definieren – ähnlich wie die Tabellen- und Spalten-Bezeichnungen in Datenbanken und Tabellenkalkulationen.

- XML ermöglicht im Gegensatz zu HTML die Definition eigener oder zusätzlicher "Befehle" (Tags) – ähnlich wie bei der Definition von Macros in der Textverarbeitung
- XML-Applikationen eignen sich als Plattform- und Software-unabhängiges Austausch-Format für Daten zwischen verschiedenen Programmen und Rechnern – ähnlich wie RTF für Texte, CVS für Tabellen, EDI für kommerzielle Anwendungen – aber in einem einheitlichen, allgemein verwendbaren, hersteller-unabhängigen Format.

Außerdem ist die Syntax von XML so streng festgelegt, dass XML-Anwendungen wesentlich einfacher, bequemer und effizienter von Programmen weiter verarbeitet werden können als HTML-Files.

Künftige →Web-Browser werden XML-Files direkt am Bildschirm darstellen und am Drucker ausdrucken können, wenn mit einem →Style-Sheet definiert ist, wie die einzelnen XML-Elemente dargestellt werden sollen. Erste Ansätze dafür gibt es im MS Internet Explorer Version 5 (mit einer Vor-Version von →XSL Style Sheets) sowie in Netscape Version 5 und Mozilla (mit →CSS Style-Sheets).

Zu diesem Zweck werden am →Web-Server sowohl das XML-File mit dem Inhalt der Information als auch das XSL- oder CSS-File mit den Layout-Angaben abgespeichert – so ähnlich wie im Textsatzsystem LaTeX, wo der Inhalt im TEX-File und das Layout im STY-File definiert werden, und so wie dort hat man auch hier die Möglichkeit, den selben Inhalt wahlweise in verschiedenen Layouts darzustellen, z.B. für große und kleine Bildschirme und für Schwarz-weiß- und Farb-Drucker.

Wenn man die Informationen für alle Benutzer verfügbar machen will, also auch für diejenigen, die noch ältere Web-Browser verwenden, muss man sie am Web-Server (zusätzlich) im normalen →HTML-Format zur Verfügung stellen – am besten mit einem Umwandlungsprogramm, das die XML-Files mit der Hilfe der Style-Files automatisch in Standard-HTML umwandelt. Und wenn die Informationen auch mit Handy-Telefonen und dergleichen erreichbar sein sollen, dann gleich auch noch im →WML-Format.

Es gibt auch Umwandlungsprogramme, die "lockere" HTML-Files in "strenge" →XHTML-Files umwandeln, damit diese mit XML-Software weiter verarbeitet oder auch durch zusätzliche Tags zu speziellen XML-Anwendungen erweitert werden können.

Für ausführlichere Informationen wird auf die

- →XML Kurz-Info

und auf die dort angegebenen →Referenzen verwiesen.

## 7.2 XHTML – die neue HTML

**XHTML = Extensible Hypertext Markup Language**

Eine Erklärung der Grundbegriffe finden Sie im →Glossar:

- →XML . →HTML . →XHTML

Mit XHTML bezeichnet man ein →HTML-File, das den strengeren Syntax-Regeln von →XML entspricht. Dies hat den Vorteil, dass sie dann nicht nur von Web-Browsern dargestellt sondern auch mit →SGML-Programmen oder →XML-Programmen weiter verarbeitet werden können.

Künftige Versionen von HTML (nach 4.0) werden vom →W3-Consortium nur mehr als →XHTML festgelegt werden.

Die wichtigsten Unterschiede von XHTML gegenüber HTML sind:

Die Groß-Klein-Schreibung ist *nicht* egal, alle XHTML-Tags und Schlüsselwörter müssen mit **Kleinbuchstaben** geschrieben werden.

Beispiel:

nicht `<P ALIGN="CENTER">`

sondern `<p align="center">`

Alle Parameter-Werte müssen in **Double-Quotes** oder Apostrophe eingeschlossen werden.

Beispiel:

nicht `<p align=center>`

sondern `<p align="center">`

**End-Tags** dürfen *niemals* weggelassen werden.

Beispiel:

nicht `<p> ... <p> ...`

sondern `<p> ... </p> <p> ... </p>`

**Einzelemente**, bei denen es keinen End-Tag gibt, müssen in der Form `<xxx />` geschrieben werden.

Beispiel:

nicht `<br>` und ``

sondern `<br />` und ``

Start- und End-tags müssen immer **richtig geschachtelt** werden.

Beispiel:

nicht `<p><b> ... </p></b>`

sondern `<p><b> ... </b></p>`

**Textmarken** müssen mit `id="marke"` definiert werden.

Beispiel:

nicht `<p><a name="xxx"> ... </a></p>`

sondern `<p id="xxx"> ... </p>`

Damit das mit alten *und* neuen Web-Browsern funktioniert, sollte man aber lieber beides kombinieren:

```
<p id="xxx"><a name="xxx"> ... </a></p>
```

Erklärungen, *warum* diese strengeren Syntax-Regeln sinnvoll und notwendig sind, finden Sie in der →XML Kurz-Info im Kapitel über die →XML-Syntax.

Es gibt **Umwandlungsprogramme**, die "normale" HTML-Files automatisch in XHTML umwandeln, z.B. das Programm "tidy" beim →W3-Consortium.

Für ausführlichere Informationen wird auf die →Referenzen verwiesen.

### 7.3 MathML für Mathematik

MathML ist eine →XML-Anwendung zur Speicherung und Darstellung von mathematischen Formeln. Derzeit wird MathML von den meisten (älteren) Web-Browsern leider noch nicht unterstützt.

Für ausführlichere Informationen wird auf die →Referenzen verwiesen.

### 7.4 CML für Chemie

CML ist eine →XML-Anwendung zur Speicherung und Darstellung von chemischen Formeln. Derzeit wird CML von den meisten (älteren) Web-Browsern leider noch nicht unterstützt.

Für ausführlichere Informationen wird auf die →Referenzen verwiesen.

### 7.5 WAP und WML für Handys

**WAP = Wireless Application Protocol**

**WML = Wireless Markup Language**

Eine Erklärung der Grundbegriffe finden Sie im →Glossar:

- →WAP . →WAP-Gateway
- →WML . →XML

HTML hat zwar den Vorteil, dass sich die in den Web-Pages enthaltenen Informationen durch das Prinzip des logischen Markup automatisch an die Fenstergröße und Schriftgröße des Client-Rechners anpassen, egal ob es sich um einen Notebook oder PC oder eine Workstation mit großem oder kleinem, niedrig oder hoch auflösendem Bildschirm handelt

Allerdings funktioniert diese automatische Anpassung nur innerhalb eines Bereiches von wenigstens einigermaßen vergleichbaren Fenster- und Papiergrößen. Typische PC- und Fernseh-Bildschirme bieten Platz für etwa 200 Wörter in ca. 20 Zeilen, Papierblätter für etwa 600 Wörter in ca. 60 Zeilen.

Für Geräte mit sehr kleinen Displays, die nur etwa 10 bis 20 Wörter in 2 bis 5 Zeilen darstellen können, muss man die Informationen im allgemeinen extra aufbereiten, in einem stark gekürzten und wesentlich kompakteren Format. Da solche Geräte meistens auch nur über geringere Rechen-Ressourcen verfügen als PCs oder

Workstations, müssen außerdem ein Format und ein Protokoll verwendet werden, die möglichst effizient verarbeitet werden können. Deshalb werden für diese Zwecke nicht HTML und HTTP sondern  $\rightarrow$ WML (das auf XML aufbaut) und  $\rightarrow$ WAP verwendet.  $\rightarrow$ XML und damit auch WML hat strengere Regeln als HTML und erlaubt dadurch eine effizientere Verarbeitung.

Wenn man eine Information für *alle* Endbenutzer mit *allen* möglichen Geräten zur Verfügung stellen will, dann muss man sie auf dem  $\rightarrow$ Web-Server in zwei Versionen verfügbar halten:

- die **komplette** Version für große Bildschirme und Printouts als  $\rightarrow$ HTML-File oder als  $\rightarrow$ XML- oder  $\rightarrow$ XHTML-File
- und die **gekürzte** Version für kleine Displays als  $\rightarrow$ WML-File.

In den meisten Fällen wird man weder des HTML-File noch das WML-File händisch erstellen, sondern beide automatisch und dynamisch aus den in einer Datenbank oder einem Workflow-System gespeicherten Informationen generieren.

Es gibt spezielle  $\rightarrow$ WAP-Gateways, die versuchen, bestehende HTML-Informationen, so gut das geht, automatisch in WML umzuwandeln. Meistens genügt eine solche automatische Umwandlung aber nicht, weil die Informationen für die kleinen Displays anders (kürzer) formuliert und anders strukturiert werden müssen.

Für ausführlichere Informationen wird auf die

- $\rightarrow$ WML-Einführung

und auf die dort angegebenen  $\rightarrow$ Referenzen verwiesen.

---

## 8. Glossar

---

### 8.1 WWW – Was ist das?

*Ein kleines WWWörterbuch*

**WWW (World Wide Web) :** WWW ist ein Informationssystem, das einen bequemen Zugriff auf Informationen, die auf vielen verschiedenen Computern gespeichert sind, in der Form von →Hypertext- und →Hypermedia-Links ermöglicht. Der Zugriff erfolgt nach dem Prinzip von →Server und →Client über das →Internet mit dem Protokoll →HTTP. Text-Informationen werden auf den →WWW-Servern in der Form von →HTML-Files, →WML-Files und →XML-Files gespeichert. Außerdem können auch Bilder, Töne, Videos und beliebige sonstige Files über das WWW übertragen werden, und es können Programme gestartet und Benutzer-Eingaben verarbeitet werden.

WWW wurde am europäischen Kernforschungszentrum CERN in Genf entwickelt und wird vom →W3-Consortium weiter entwickelt. Der Name bedeutet so etwas wie ein "weltweites Spinnennetz".

(siehe auch →Geschichte und →Referenzen)

**A – Z :** *Die weiteren Stichwörter werden in alphabetischer Reihenfolge erklärt.*

**Active-X :** Active-X ist ein von der Firma Microsoft erfundenes System-Interface, das auch zur Ausführung von verschiedenen Aktionen innerhalb des Microsoft-→Web-Browsers Internet-Explorer verwendet werden kann. Im Gegensatz zu →Java ist dieses System weder plattformunabhängig noch mit den notwendigen Sicherheitsmechanismen ausgestattet.

Mehr über Active-X finden Sie bei der Firma Microsoft.

**Applet :** Applets sind →Java-Programme, die in →Web-Pages eingebettet sind und auf dem →Client-Rechner innerhalb des →Web-Browsers ausgeführt werden.

**ASP (Active Server Pages) :** ASP-Seiten sind HTML-Files mit besonders gekennzeichneten eingebetteten →JavaScript-Programmen, die wie bei →SSI am →Web-Server ausgeführt werden. Das Ergebnis wird dann im normalen HTML-Format (ohne JavaScript) an den →Client gesendet, ASP wird nur von wenigen Web-Servern unterstützt.

siehe auch →JSP und →PHP.

**Bookmark :** siehe →Lesezeichen

**Client :** Clients (Kunden) sind die Benutzer, die Informationen haben wollen. Client-Programme sind die Programme, mit denen die Benutzer von ihren eigenen Rechnern (PCs) aus auf die Informationen, die auf den →Servern gespeichert sind, zugreifen. WWW-Client-Programme werden auch als →Web-Browser bezeichnet.

- CGI (Common Gateway Interface)** : CGI-Programme sind Programme oder Shell-Skripts, die auf dem →WWW-Server laufen, eventuelle Daten-Eingaben des →Clients verarbeiten und die Ergebnisse im →HTML-Format an den →Client-Rechner senden. →Java-Programme können effizienter als →Servlets in den Web-Server eingebettet werden.
- CSS (Cascading Style Sheets)** : CSS ist ein vom →W3-Consortium definiertes, einfaches Format für →Style-Sheets für die Darstellung von →HTML- und →XML-Dokumenten.  
siehe auch →XSL.
- CSV (Comma separated variables)** : CSV ist ein System-unabhängiges Datei-Format für den Austausch von Tabellen zwischen Spreadesheet-Programmen und Datenbanken (MS-Excel, Lotus 1-2-3, MS-Access u.a.).  
siehe auch →XML
- DHTML (Dynamische HTML)** : Dynamische HTML ermöglicht die Veränderung der Darstellung von →Web-Pages (→HTML-Files) in Abhängigkeit von Aktionen des Benutzers direkt im →Web-Browser mit Hilfe von →JavaScript.
- DOM (Document Object Model)** : DOM ist ein Objektmodell, es beschreibt die in einem Dokument einer bestimmten →XML-Anwendung enthaltenen Elemente als Objekte, für die Verarbeitung mit einer objekt-orientierten Programmiersprache wie z.B. →Java. DOM liefert eine komplette Baumstruktur aller Objekte eines XML-Dokuments und eignet sich daher nicht für extrem große XML-Files.  
siehe auch →SAX.
- DSSSL (Document Style Semantics Specification Language)** : DSSSL ist eine sehr mächtige und daher auch sehr komplexe Sprache für die Spezifikation der Darstellung von →SGML-Dokumenten.  
siehe auch →CSS und →XSL.
- DTD (Document Type Definition)** : Eine DTD beschreibt die Struktur einer Klasse von →SGML- oder →XML-Dokumenten, also einer SGML- oder →XML-Applikation, mit Hilfe eines Text-Files, das alle Syntax-Regeln in einem von →SGML vorgeschriebenen Format enthält. Beispielsweise ist jede →HTML-Version durch eine DTD definiert. Eine Alternative dazu ist die Definition mit Hilfe eines →Schemas.
- EDI (Electronic Data Interchange)** : EDI ist ein Systeme zur Festlegung von normierten Formaten für den Austausch von Daten zwischen kommerziellen Datenverarbeitungsprogrammen (z.B. EDIFACT).  
siehe auch →XML
- Gopher** : Gopher ist ein Informationssystem, das vor dem Siegeszug des →WWW weltweit verwendet wurde. Gopher ermöglichte den Zugriff auf Informationen, die auf vielen verschiedenen Computern gespeichert waren, in einer hierarchischen Struktur von Directories (Menüs) und Files (Informationen) Der Zugriff erfolgte nach dem Prinzip von →Server und →Client über das →Internet mit dem Gopher-Protokoll.  
Gopher wurde an der Universität von Minnesota entwickelt. Der Name kommt von der Beutelratte, dem "Nationaltier" und Spitznamen von Minnesota.

**Home-Page :** Der Begriff Home-Page bedeutet diejenige  $\rightarrow$ Web-Page, bei der ein Benutzer den Zugriff auf das  $\rightarrow$ WWW beginnt, bei der er also "zu Hause" ist. Dies wird von jedem Benutzer in seinem  $\rightarrow$ Web-Browser konfiguriert (siehe auch  $\rightarrow$ Lesezeichen (Bookmarks)). Oft wird dafür die  $\rightarrow$ Startseite der eigenen Universität bzw. Firma verwendet. Das Wort Home-Page wird deshalb manchmal auch fälschlich für die  $\rightarrow$ Startseite eines Informationssystems oder Web-Servers verwendet, die für Besucher dient, die dort *nicht* "zu Hause" sind ( $\rightarrow$ Portal).

**HTML (Hypertext Markup Language) :** HTML ist das Format, in dem die Text- und  $\rightarrow$ Hypertext-Informationen im  $\rightarrow$ WWW gespeichert und übertragen werden.

- **HTML 2.0** ist die offizielle Norm, die diejenigen Grundfunktionen von HTML definiert, die von *allen* (auch den älteren)  $\rightarrow$ Web-Browsern sinnvoll dargestellt werden.
- **HTML 3.2** ist der De-facto-Standard, der diejenigen HTML-Elemente umfaßt, die von fast allen  $\rightarrow$ Browser-Versionen (ab 1996/97) weitgehend unterstützt werden.
- **HTML 4.0** ist ein Vorschlag des  $\rightarrow$ W3-Consortiums, der von neueren  $\rightarrow$ Browser-Versionen (ab 1997/98) zumindest teilweise unterstützt wird.
- Die Weiterentwicklung von HTML ab 1999/2000 wird in die Richtung von  $\rightarrow$ XHTML gehen.

HTML ist eine "Content-based Markup Language", die mit  $\rightarrow$ SGML definiert ist. HTML unterstützt ein *logisches Markup*, bei dem die logische Bedeutung der Textteile so festgelegt wird, dass sie vom jeweiligen  $\rightarrow$ Web-Browser in der für den Benutzer ( $\rightarrow$ Client) optimalen Form dargestellt werden können.

HTML-Files können mit einfachen Text-Editoren oder mit speziellen Hilfsprogrammen erstellt oder aus bestehenden Dokumenten oder Datenbanken umgewandelt werden.

Mehr über HTML finden Sie in der  $\rightarrow$ HTML-Einführung, bei  $\rightarrow$ Stefan Münz und beim  $\rightarrow$ W3Consortium.

(siehe auch  $\rightarrow$ Geschichte und  $\rightarrow$ Referenzen)

**HTTP (Hypertext Transfer Protocol) :** HTTP ist das Protokoll, nach dem die Informationen zwischen  $\rightarrow$ WWW-Servern und  $\rightarrow$ WWW-Clients über das  $\rightarrow$ Internet übertragen werden. Es gibt auch ein "Secure HTTP" (siehe  $\rightarrow$ SSL).

**HTTPS (Hypertext Transfer Protocol, secure) :** siehe  $\rightarrow$ SSL

**Hyper-G und Hyperwave :** Hyper-G ist ein Informationssystem, das Eigenschaften von  $\rightarrow$ Gopher und  $\rightarrow$ WWW vereinigt. Es unterstützt hierarchische Strukturen von  $\rightarrow$ Hypertext- und  $\rightarrow$ Hypermedia-Files sowie Stichwort- und Volltext-Suchen in diesen Strukturen ("Collections"). Hyper-G-Server (Hyperwave) können als spezielle  $\rightarrow$ Web-Server für komplexe Informationssysteme eingesetzt werden. Hyper-G wurde an der Technischen Universität von Graz entwickelt und wird von der Firma HyperWave vertrieben. Der Name ist aus "Hypermedia" und "Graz" zusammengesetzt.

- Hypermedia :** Mit Hypermedia bezeichnet man Multi-Media-Systeme (Texte, Bilder, Töne, Video-Sequenzen) mit Querverweisen wie bei →Hypertext.
- Hypertext :** Unter Hypertext versteht man Texte mit Querverweisen, die ähnlich wie in einem Lexikon oder in einer Literaturliste die Verbindung zu weiteren Informationen herstellen. Im →WWW werden solche Verweise mit der Hilfe von →URLs realisiert.  
Bei Hypertext-Dokumenten gibt es nicht (wie bei Druckwerken) eine einzige, lineare Lesereihenfolge, sondern die Leser können jede Einzelinformation über viele verschiedene Wege und von vielen verschiedenen Stellen aus erreichen.
- Information Highway :** "Information Highway", "Daten-Autobahn", "Infobahn" und dergleichen sind populäre Bezeichnungen für die Pläne zu einem weltumspannenden Netz von leistungsfähigen Verbindungen, die für viele verschiedene private und kommerzielle Zwecke genutzt werden sollen. Die zum Teil noch recht vagen Vorstellungen zielen auf etwas wie eine Vereinigung von →Internet, Telefon, Video-Konferenzen, Kabel-Fernsehen, Video-Verleih, Tele-Arbeit, Tele-Shopping und ähnlichen Services.  
(siehe auch →Geschichte)
- Internet :** Das Internet ist das umfangreichste Computer-Netzwerk der Welt. Es verbindet mehrere Millionen Computer (einschließlich PCs) und mehrere zehn Millionen Menschen.  
Der Name kommt von "Interconnected Networks" (verbundene Netze); das Internet ist ein Zusammenschluß von vielen lokalen, nationalen und internationalen Computer-Netzen, die alle das Protokoll TCP/IP verwenden und die jeweils lokal, nicht über eine Welt-Zentrale, verwaltet werden ("Domains"). Das Internet unterstützt viele verschiedene Services. Die wichtigsten sind:
- **"Telnet"** für den Aufruf von Programmen auf anderen Computern,
  - **"FTP"** (File Transfer Protocol) für die Übertragung von Files auf andere Computer,
  - **"Electronic Mail"** (elektronische Briefpost), **"Usenet News"** (Veröffentlichungen in Diskussionsforen) und **"IRC"** (Internet Relay Chat) für den Austausch von Informationen mit anderen Computer-Benutzern,
  - →**"WWW"** und →**"WWmB"** für den Zugriff auf Informationssysteme in aller Welt.
- (siehe auch →Geschichte und →Referenzen)
- Intranet :** Unter Intranet versteht man ein nicht öffentliches, firmen-internes Netz, das die gleiche Technik wie das weltweite →Internet verwendet.
- Java :** Java ist eine plattformunabhängige objekt-orientierte Programmiersprache. Im Gegensatz zur Ausführung von →CGI-Programmen auf dem →Server-Computer ermöglicht Java auch die Ausführung von Aktionen (Programmen, "Applets") auf dem →Client-Rechner und ist mit allen für diesen Zweck notwendigen Sicherheitsmechanismen ausgestattet.  
Java wurde von der Computer-Firma Sun entwickelt und wird von mehreren neueren →Web-Browsern (seit 1996/97) zumindest teilweise unterstützt. Der Name stammt von einer amerikanischen Bezeichnung für Kaffee.  
Mehr über Java finden Sie in der →Java-Einführung und bei der →Firma Sun.

- JavaScript :** JavaScript ist eine von der Firma Netscape erfundene einfache Skript-Sprache zur Ausführung von bestimmten Aktionen innerhalb des Web-Browsers. Im Gegensatz zu →Java ist diese Sprache weder Software-unabhängig noch mit den notwendigen Sicherheitsmechanismen ausgestattet.  
Mehr über JavaScript finden Sie bei →Stefan Münz und bei der Firma Netscape.
- JDK (Java Development Kit) :** Das JDK enthält alle Komponenten, die für das Erstellen und die Verwendung von Programmen und →Applets in der Programmiersprache →Java benötigt werden, also den Java-Compiler, das →Java Runtime Environment und diverse Hilfsprogramme.
- JRE (Java Runtime Environment) :** Das JRE enthält alle Komponenten, die für die Ausführung von →Java-Programmen benötigt werden, also die →Java Virtual Machine und die Java-Klassenbibliothek. Viele neuere →Web-Browser (ab 1996/97) enthalten ein JRE, das die Ausführung von →Applets innerhalb vom →Web-Pages ermöglicht.
- JSP (Java Server Pages) :** JSP-Seiten sind HTML-Files mit besonders gekennzeichneten eingebetteten →Java-Programmen, die wie bei →SSI am →Web-Server ausgeführt werden. Das Ergebnis wird dann im normalen HTML-Format (ohne Java) an den →Client gesendet, JSP wird nur von wenigen Web-Servern unterstützt.  
siehe auch →ASP und →PHP.
- JVM (Java Virtual Machine) :** Die JVM ermöglicht die Ausführung der plattformunabhängigen →Java-Programme auf einem bestimmten Rechner. Sie ist Teil des →JDK bzw. →JRE.
- Lesezeichen :** Lesezeichen (meist als "Bookmarks", "Hotlists" oder "Favorites" bezeichnet) sind so etwas wie eine persönliche →Suchhilfe: Der Benutzer speichert darin die →Adressen von Informationen, die er interessant gefunden hat, so auf seinem eigenen →Client-Rechner ab, dass er sie bei Bedarf schnell und einfach wiederfinden kann.
- PDF (Portable Document Format) :** Mit dem ebenfalls von Adobe entwickelten Format PDF können →PostScript-Dokumente mit →Hypertext-Links versehen und im →WWW gespeichert und übertragen werden. Dies kann eventuell als Alternative zu →HTML eingesetzt werden, wenn das genaue Aussehen der →Web-Page wichtiger ist als die flexible Anpassung an den jeweiligen →Client.  
PDF-Files können mit kostenlos verfügbaren Client-Programmen (z.B. Acrobat Reader) gelesen und mit kostenpflichtiger Software (Acrobat Writer) erstellt werden.
- PHP (Php Hypertext Preprocessor) :** PHP-Seiten sind HTML-Files mit besonders gekennzeichneten eingebetteten PHP-Programmen, die wie bei →SSI am →Web-Server ausgeführt werden. Das Ergebnis wird dann im normalen HTML-Format (ohne PHP) an den →Client gesendet,  
siehe auch →ASP und →JSP.  
Mehr über PHP finden Sie bei →<http://www.php.net/>. und bei →Kristian Koehntopp.
- Portal :** siehe →Startseite

- PostScript :** PostScript ist eine von der Firma Adobe entwickelte Seitenbeschreibungssprache. Im Gegensatz zum *logischen* Markup von  $\rightarrow$ HTML, das eine Anpassung der Darstellung an den  $\rightarrow$ Client ermöglicht, wird mit PostScript das *Aussehen* der Dokumente (Texte und Bilder) in allen Details festgelegt.
- Proxy-Server :** Proxy-Server ("nahe liegende" Server) sind Zwischenstufen auf dem Weg zwischen  $\rightarrow$ Client und  $\rightarrow$ Server: Der Client greift auf den Proxy-Server zu, dieser greift auf den eigentlichen Server zu und sendet dann die Ergebnisse an den Client.  
Proxy-Server dienen zur Verbindung von verschiedenen Netzen (z.B. Firewalls zwischen  $\rightarrow$ Intranet und  $\rightarrow$ Internet oder  $\rightarrow$ WAP-Gateways zwischen Telefonnetz und Internet), zum Ausfiltern von verbotenen Verbindungen (Firewalls), und zum Zwischenspeichern von mehrfach benötigten Dateien (Cache-Proxy).
- RTF (Rich Text Format) :** RTF ist ein System-unabhängiges Datei-Format für den Austausch von Texten zwischen Textverarbeitungsprogrammen (MS-Word, WordPerfect u.a.).  
siehe auch  $\rightarrow$ XML
- SAX (Simple API for XML) :** SAX ist eine Programm-Schnittstelle (Application Programmers Interface API) für die Verarbeitung einer Klasse von  $\rightarrow$ XML-Dokumenten, also einer  $\rightarrow$ XML-Applikation, mit Hilfe einer objekt-orientierten Programmiersprache wie z.B.  $\rightarrow$ Java. SAX liefert ein XML-Element nach dem anderen in einem Eingabestrom und eignet sich daher auch für sehr große XML-Files.  
siehe auch  $\rightarrow$ DOM.
- Schema (Mehrzahl: Schemata) :** Ein Schema beschreibt die Struktur einer Klasse von  $\rightarrow$ XML-Dokumenten, also einer  $\rightarrow$ XML-Applikation, ähnlich wie eine  $\rightarrow$ DTD, jedoch nicht in der DTD-Syntax sondern in einer eigenen XML-Syntax.
- Server :** Server (Verkäufer, Bedienender) sind die Computer, auf denen die Informationen gespeichert sind.  
siehe auch  $\rightarrow$ Client und  $\rightarrow$ Web-Server.
- Servlet :** Servlets sind  $\rightarrow$ Java-Programme, die wie  $\rightarrow$ CGI innerhalb des  $\rightarrow$ Web-Servers ausgeführt werden.
- SGML (Standard Generalized Markup Language) :** SGML ist eine Meta-Sprache, in der Markup-Sprachen wie z.B.  $\rightarrow$ HTML definiert werden können. Die Struktur und Syntax solcher Markup-Sprachen ("SGML-Anwendungen") wird mit einer  $\rightarrow$ DTD festgelegt.  
siehe auch  $\rightarrow$ XML
- SSI (Server-Side Includes) :** SSI ermöglicht die automatische Veränderung des Inhalts von  $\rightarrow$ Web-Pages ( $\rightarrow$ HTML-Files) am  $\rightarrow$ Web-Server.  
siehe auch  $\rightarrow$ ASP und  $\rightarrow$ JSP
- SSL (Secure Socket Layer) :** SSL (Secure Socket Layer) und das Protokoll https (Secure  $\rightarrow$ HTTP) ermöglichen die abhör- und fälschungssichere Übertragung der Informationen zwischen  $\rightarrow$ Web-Browser ( $\rightarrow$ Client) und  $\rightarrow$ Web-Server.

**Startseite (Portal) :** Mit dem Begriff Startseite (Einstiegsseite, Portal, Welcome-Page, Index-Page) wird diejenige →Web-Page bezeichnet, bei der die Leser, Gäste, Besucher, den Zugriff auf einen bestimmten Satz von Informationen oder auf einen ganzen →Web-Server beginnen sollen (falls sie nicht mittels →Suchhilfen oder →Lesezeichen direkt zu den Einzelinformationen springen). Manchmal wird dafür auch die nicht zutreffende Bezeichnung →Home-Page verwendet.

**Style-Sheets :** Style-Sheets bieten eine gute Möglichkeit, die Darstellung des Inhalts von →Web-Pages in einem einheitlichen und konsistenten Layout zu bewirken. Neuere →Web-Browser (ab 1997) unterstützen zumindest teilweise Style Sheets im Format →CSS1.

**Suchhilfen :** Die Gesamtheit der über das →Internet und im →World Wide Web verfügbaren Informationen und Services wurde ab 1994 so umfangreich und vielfältig und unüberschaubar, dass Suchhilfen für das Auffinden von Informationen notwendig sind.

Es gibt mehrere verschiedene Suchhilfen mit Datenbanken, die Stichwort- oder Volltext-Suchen entweder im gesamten Netz oder nur in bestimmten Regionen oder Servern oder für bestimmte Themengebiete ermöglichen. Sie werden oft als die Spinnen im weltweiten Spinnennetz ("Spider" oder "Crawler"), Erntemaschinen ("Harvester"), Informationsmakler ("Broker") oder einfach nur als Suchmaschinen ("Search Engines") bezeichnet. Immer wieder werden neue Suchhilfen entwickelt, und es gibt auch schon Hilfsmittel zum Auffinden der Suchhilfen ("Meta-Suchmaschinen").

Ab etwa 1999 wurde die Anzahl der Web-Pages und der über das WWW dynamisch erreichbaren Datenbank-Informationen so umfangreich, dass auch die Suchmaschinen nicht mehr alle diese Informationen erfassen können, sondern nach verschiedenen Gesichtspunkten (Region, Thema, Art, Vertrauenswürdigkeit o.dgl.) eine Auswahl treffen müssen oder hoffnungslos unvollständig bleiben.

(siehe →Referenzen).

**URL (Uniform Resource Locator) :** URL ist die "Adresse", die das →Client-Programm benötigt, um eine bestimmte Information vom jeweiligen →Server-Computer zu erhalten. Der URL enthält zu diesem Zweck Informationen wie die Art des Zugriffs (Protokoll), die Adresse des Server-Computers (Hostname), eventuell mit einem Username und Paßwort oder einer Port-Nummer, und das Directory und den Filenamen der Datei, in der die gewünschte Information gespeichert ist, sowie eventuell die Stelle innerhalb der Datei oder die Parameter für ein →CGI-Programm oder für einen Suchvorgang.

**VRML (Virtual Reality Modelling Language) :** VRML ist eine zu →HTML ähnliche →Hypermedia-Sprache für die Speicherung und Übertragung von dreidimensionalen Virtual-Reality-Szenen im →WWW, →Gopher oder →Hyper-G (Hyperwave).

**WAP (Wireless Application Protocol) :** WAP ist ein einfaches Gegenstück zum Protokoll →HTTP und eignet sich deshalb für die Übertragung von →WML-Files auf Mobil-Telefone und ähnliche kleine Geräte.

- WAP-Gateway :** Ein WAP-Gateway ist ein  $\rightarrow$ Proxy-Server, der Mobil-Telefone (Handys) auf der einen Seite mit  $\rightarrow$ Web-Servern auf der anderen Seite verbindet. Die Verbindung zwischen Handy und WAP-Gateway erfolgt über das Telefonnetz mit dem Protokoll  $\rightarrow$ WAP, die Verbindung zwischen WAP-Gateway und Web-Server über das  $\rightarrow$ Internet mit dem Protokoll  $\rightarrow$ HTTP. Die Bezeichnung WAP-Gateway wird auch für Umwandlungsprogramme verwendet, die als  $\rightarrow$ CGI oder  $\rightarrow$ Servlet auf einem  $\rightarrow$ Web-Server laufen und die dort vorhandenen  $\rightarrow$ HTML-Informationen automatisch (so gut das geht) in  $\rightarrow$ WML übersetzen.
- Web-Browser :** Als Web-Browser bezeichnet man  $\rightarrow$ Client-Programme für den Zugriff auf  $\rightarrow$ WWW- $\rightarrow$ Server. Es gibt viele verschiedene solche Programme, sowohl public domain als auch kommerziell. Die meisten laufen auf einer graphischen Benutzeroberfläche mit Maus oder Touch-Screen. Spezielle Browser-Programme können die Informationen auch in Zeilen-orientiert oder in Blindenschrift oder akustisch (als gesprochene Texte) oder in Form von dreidimensionalen Virtual-Reality-Szenen darstellen. Typische Web-Browser waren bzw. sind Mosaic, Lynx, Netscape, Internet-Explorer, Opera, WebTV, iPlanet, Mozilla. Die meisten Web-Browser unterstützen nicht nur den Zugriff auf  $\rightarrow$ WWW-Server sondern auch auf andere  $\rightarrow$ Internet-Services wie Telnet, FTP, Electronic Mail und Usenet News.  
(siehe auch  $\rightarrow$ Geschichte und  $\rightarrow$ Referenzen)
- Web-Page :** Als Web-Page (Netz-Seite) bezeichnet man ein im  $\rightarrow$ WWW veröffentlichtes  $\rightarrow$ HTML-File.  
siehe auch  $\rightarrow$ Home-Page
- Web-Server :** Web- $\rightarrow$ Server (WWW-Server) laufen meistens auf Unix-Rechnern und werden dort auch als  $\rightarrow$ HTTP-Dämonen (Hilfsgeister) bezeichnet. Es gibt mehrere solche Software-Produkte, sowohl public domain (z.B.  $\rightarrow$ Apache) als auch kommerziell. Typische Web-Server waren bzw. sind CERN, NCSA Apache, Microsoft IIS, und es gibt auch Datenbanksysteme und Workflow-Systeme mit Web-Schnittstellen wie z.B. Oracle Web Server, Oracle Application Server, Lotus Domino u.a. In die Hardware integrierte Web-Server werden auch zur Steuerung von Haushaltsgeräten, Chipkarten etc. verwendet.  
siehe auch  $\rightarrow$ Server und  $\rightarrow$ Web-Browser.
- WML (Wireless Markup Language) :** WML ist ein Gegenstück zu  $\rightarrow$ HTML für die Darstellung von Informationen auf Mobil-Telefonen (Handys) und auf anderen Geräten mit kleinen Displays. WML baut nicht auf HTML sondern auf  $\rightarrow$ XML auf, ist also ähnlich wie HTML, aber nicht mit HTML kompatibel. WML-Files werden wie HTML-Files auf  $\rightarrow$ Web-Servern gespeichert, der Zugriff erfolgt über ein  $\rightarrow$ WAP-Gateway mit dem Protokoll  $\rightarrow$ WAP. Mehr über WML finden Sie in der  $\rightarrow$ WML-Einführung und beim  $\rightarrow$ WAP-Forum.  
(Es gibt auch andere Formate, die mit WML abgekürzt werden, aber mit der Wireless Markup Language nichts zu tun haben: Webmaker Language, Website Meta Language, Widget Meta Language u.a.)
- WWW (World-wide Web) :** siehe den ersten Eintrag  $\rightarrow$ oben

**WWW:MMM und WWmB (World-wide mobile Business)** : Die Bezeichnungen WWW:MMM (world-wide web, mobile media mode) und WWmB (world-wide mobile business) meinen den Zugriff auf →Web-Informationen von Mobil-Telefonen (Handys, GSM), persönlichen Digital-Assistenten (PDA, Organizer), Palmtop-Computern, Auto-Navigationshilfen und ähnlichen Geräten aus. Dafür werden das Datei-Format →WML und das Protokoll →WAP verwendet.

**XHTML (Extensible Hypertext Markup Language)** : Mit XHTML bezeichnet man ein →HTML-File, das den strengeren Syntax-Regeln von →XML entspricht und deshalb besser von Computer-Programmen weiterverarbeitet werden kann.

- XHTML 1.0 entspricht dem Funktionsumfang von →HTML 4.0

**XML (Extensible Markup Language)** : XML ist eine vereinfachte Form von →SGML. So wie →HTML mit SGML definiert ist, so kann man mit XML eigene Markup-Sprachen oder auch eigene Erweiterungen von HTML bzw. →XHTML mit eigenen Tags für bestimmte Elemente mit bestimmten logischen Bedeutungen definieren.

Die mit XML definierten Markup-Sprachen werden als →XML-Anwendungen bezeichnet. Die Syntax, Struktur und Bedeutung der Tags wird für jede XML-Anwendung mit einer →DTD oder einem →Schema definiert. Die Verarbeitung kann mit →XML-Parsern mit →DOM oder →SAX erfolgen. Wie die Elemente sichtbar dargestellt werden sollen, kann mit →XSL oder →CSS definiert werden. XML-Dokumente können auch →Hypertext-Links enthalten, entweder wie in HTML oder in der Form von →XLink oder →XPointer.

XML-Anwendungen eignen sich einerseits für die Darstellung in →Web-Browsern - also als Ersatz oder Ergänzung von →HTML - und andererseits für die Verarbeitung mit EDV-Programmen (z.B. in der Textverarbeitung, Tabellenkalkulation, Datenbanken, kommerziellen Anwendungen u.a.) und als Austauschformat zwischen solchen Programmen - also als Ersatz für →RTF, →CSV und →EDI.

Mehr über XML finden Sie in der →XML Kurz-Info und beim →W3Consortium.

**XML-Anwendung (Applikation)** : Unter XML-Anwendung oder XML-Applikation versteht man die Festlegung (Normierung) von →XML-Befehlen für eine Klasse von XML-Dokumenten gleicher Struktur, also für einen bestimmten Zweck.

Das Format und die Struktur der XML-Files und die Eigenschaften und die Schachtelung der darin vorkommenden Elemente (XML-Befehle, Tags, Entities) werden für eine XML-Anwendung mit einer →DTD oder einem →Schema definiert - so wie man bei →EDI mit UNSM und MIG das Format und die Struktur der Nachrichten und die Bedeutung der darin enthaltenen Daten für einen bestimmten Nachrichtentyp definieren kann.

Beispiele für XML-Anwendungen sind

- DocBook (für gedruckte Texte und Bücher),
- →WML (für Online-Informationen auf kleinen Displays wie z.B. Handys),
- →XHTML (für Online-Informationen auf großen Displays wie z.B. PCs und Fernsehschirmen),
- →MathML (für mathematische Formeln),

- →CML (für chemische Formeln),
- →SVG (für Vektor-Graphiken),
- u.v.a.

**XML-Parser :** Ein XML-Parser ist ein Programm, das ein →XML-File liest und den Inhalt in der Form von →DOM oder →SAX liefert. Ein validierender Parser überprüft zusätzlich die Richtigkeit der Daten an Hand der →DTD oder des →Schemas.

**XSL (Extensible Style Language) :** Mit XSL wird ein →Style-Sheet definiert, das angibt, wie der in einem →XML-Dokument definierte Inhalt vom →Web-Browser oder von anderen Programmen dargestellt werden soll.

XSL ist mächtiger als →CSS und →DHTML:

- Mit **XSLT** (Transformation) kann man aus einem XML-File ein anderes XML-File machen, also z.B. bestimmte Elemente weglassen, die Elemente in anderen Reihenfolgen anordnen und zusätzliche Elemente hinzufügen,
- und mit **XSL-FO** (Formatierung) kann man das Layout der Darstellung für die Elemente festlegen.

---

## 9. Die roten Fäden der Ariadne

---

Hinweise zur Online-Verwendung der Hypertext-Version dieses Handbuches

### 9.1 Wegweiser

Das hier ist nicht ein lineares Handbuch, sondern ein →Hypertext. Sie können ihn auf viele verschiedene Arten lesen, je nachdem, was Sie erreichen wollen.

**Was wollen Sie erreichen?**

- →Ich möchte den **kompletten Text** am Bildschirm lesen.
- →Ich möchte den kompletten Text als Handbuch auf Papier **ausdrucken**.
- →Ich möchte im **Inhaltsverzeichnis** herumblättern.
- →Das ist mir alles zu lang, ich möchte eine **Kurz-Info**.
- →Ich möchte rasch einen bestimmten HTML-Befehl **nachschlagen**.
- →Ich möchte kurz erfahren, **wofür** WWW und HTML überhaupt gut sind.
- →Ich kenne mich im WWW schon recht gut aus und möchte alles über die **HTML-Befehle** erfahren.
- →Ich möchte nur rasch eine **kurze Information** im WWW veröffentlichen.
- →Ich möchte eine **umfangreiche Information** gut strukturiert im WWW veröffentlichen.
- →Ich möchte eine tolle **Liste** von heißen Verbindungen zu kühlen Quellen zusammenstellen.
- →Ich möchte, daß meine Informationen möglichst bunt und modern **aussehen**.
- →Ich möchte, daß meine Informationen von möglichst vielen Leuten **gelesen werden**.
- →Ich möchte mehr über die **Hintergründe** von WWW und HTML erfahren.
- →Ich möchte die HTML-Einführung auf meinem Computer **off-line lesen**.
- →Ich möchte die HTML-Einführung in einer Schulung **verwenden**.
- →Ich möchte eigentlich etwas ganz **anderes**.

### 9.1.1 Ich möchte den kompletten Text lesen.

1. →Vorwort
2. →Grundlagen
3. →Textelemente
4. →Hypertext-Links
5. →Bilder und Töne
6. →Layout
7. →Spezialeffekte
8. →Interaktion mit dem Benutzer
9. →Geschichte und Geschichten
10. →Entwicklungen für die Zukunft
11. →Wörterbuch (Glossar)
12. →Referenzen
13. →Liste der HTML-Befehle (Index)

### 9.1.2 Ich möchte das komplette Handbuch auf Papier ausdrucken.

Für das Ausdrucken der HTML-Einführung auf Papier gibt es zwei Möglichkeiten:

- Entweder Sie lesen die HTML-Files (siehe →komplettes Handbuch) und drucken jeden Teil mit dem Druck-Befehl (print) Ihres →Web-Browsers auf Ihrem lokalen Drucker aus.  
Falls Sie ein Umwandlungsprogramm wie z.B. "MS Word Internet Assistant" verwenden, können Sie dabei auch Seitenwechsel, Seitennummern u.dgl. einfügen.
- oder Sie übertragen das →PostScript-File der kompletten HTML-Einführung auf Ihren Rechner und drucken es auf einem PostScript-Drucker aus.  
Dieses PostScript-File hat eine Größe von ungefähr 1 MByte und ergibt einen Printout von ungefähr 100 Seiten.  
Eine mit Gnuzip komprimierte Version ist auch über FTP erreichbar:  
→ftp://mail.boku.ac.at/www/html Einf.ps.gz (ca. 300 kBytes)

Bitte, beachten Sie in beiden Fällen die →Copyright-Bestimmungen.

### 9.1.3 Ich möchte einen bestimmten HTML-Befehl nachschlagen.

- Verwenden Sie den Textsuche-Befehl Ihres →Web-Browsers in der →alphabetischen Liste der HTML-Befehle oder im →ausführlichen Inhaltsverzeichnis.

### 9.1.4 Ich möchte kurz erfahren, wofür WWW und HTML überhaupt gut sind.

1. →Vorwort
2. →Was ist WWW?
3. →Was ist HTML?
4. →Was kann ich im WWW veröffentlichen?
5. →Wie kann ich Web-Pages erstellen?
6. →Wie kann ich etwas im WWW veröffentlichen?

**9.1.5 Ich kenne WWW und möchte alles über HTML erfahren.**

1. →Vorwort
2. →Was ist HTML?
3. →Inhalt und Form
4. →Richtige HTML
5. →Format der Markup-Befehle (HTML-Tags)
6. →<html> <head> <body>
7. →Filename und Directory
8. →Textelemente
9. →Hypertext-Links
10. →Bilder und Töne
11. →Layout und Spezialeffekte
12. →Geschichte und Geschichten

**9.1.6 Ich möchte rasch eine kurze Information im WWW veröffentlichen.**

1. →Vorwort
2. →Verwendung von MS-Word
3. →Aufbau eines HTML-Files
4. →Absätze und Zeilenumbruch
5. →Buchstaben und Sonderzeichen
6. →Wie kann ich meine HTML-Files erstellen?
7. →Wie kann ich meine HTML-Files im WWW verfügbar machen?

**9.1.7 Ich möchte eine umfangreiche Information gut strukturiert im WWW veröffentlichen.**

1. →Vorwort
2. →Frisch geplant ist halb gewonnen!
3. →Aufteilung der Information auf einzelne HTML-Files
4. →Inhalt und Form
5. →Richtige HTML
6. →Format der Markup-Befehle (HTML-Tags)
7. →Aufbau eines HTML-Files
8. →Textelemente
9. →Hypertext-Links
10. →Inhaltsverzeichnisse
11. →Wie kann ich meine HTML-Files erstellen?
12. →Wie kann ich meine HTML-Files im WWW veröffentlichen?

**9.1.8 Ich möchte eine Liste von "heißen" Verbindungen zusammenstellen.**

1. →Vorwort
2. →Suchhilfen im WWW
3. →URLs
4. →Verweise zu anderen Informationen
5. →Listen von Verweisen
6. →Format der Listen

### 9.1.9 Ich möchte, daß meine Informationen bunt und modern aussehen

1. →Vorwort
2. →Frisch geplant ist halb gewonnen!
3. →die Wirkung auf die Menschen
4. →Inhalt und Form
5. →Richtige HTML
6. →Bilder und Töne
7. →Farben
8. →Trennlinien
9. →hervorgehobene Wörter
10. →hervorgehobene Absätze
11. →Überschriften
12. →Listen und Aufzählungen
13. →Layout und Spezialeffekte

### 9.1.10 Ich möchte, daß meine Informationen von vielen Leuten gelesen werden.

1. →Vorwort
2. →Frisch geplant ist halb gewonnen!
3. →Wie kann ich meine HTML-Files im WWW veröffentlichen?
4. →Was soll ich im WWW veröffentlichen?
5. →Wie werden meine Informationen gesucht?
6. →Wie kann ich Reklame für meine Informationen machen?
7. →Beschlagwortung für Suchhilfen mit <meta>
8. →Schönes Layout
9. →die Wirkung auf die Menschen
10. →Günstige Aufteilung
11. →Richtige HTML
12. →Inhalt und Form
13. →Zugriffe zählen

### 9.1.11 Ich möchte mehr über die Hintergründe von WWW und HTML erfahren.

1. →Vorwort
2. →Geschichte und Geschichten
3. →WWW – Was ist das?
4. →Was darf ich im WWW veröffentlichen?
5. →Was soll ich im WWW veröffentlichen?
6. →Inhalt und Form
7. →Richtige HTML
8. →Referenzen

### 9.1.12 Ich möchte etwas anderes.

- →Wer sucht, der findet...
- →...ein Auto im Dschungel?

## 9.2 Referenzen

- deutschsprachige HTML-Beschreibungen:
  - →Deutsche HTML FAQ von Rainer Klier auf  
→<http://www.franken.de/users/tychen/faq/htmlfaq.html>
  - →HTML-Dateien selbst erstellen, von Stefan Münz auf  
→<http://www.netzwelt.com/selfhtml/>
  - →HTML-Einführung von Hubert Partl (BOKU Wien) auf  
→<http://www.boku.ac.at/html Einf/>
  - →HTML Kurzanleitung von Mark Seuffert (Heidelberg)
  - →HTML Einführung von Dierk Lucyga (Uni Konstanz)
  - →HTML Kurz-Info von Hubert Partl (BOKU Wien)
  - →deutsche Version des Bare Bones Guide von Kevin Werbach (USA)
- Online-Informationen an der BOKU Wien auf  
→<http://www.boku.ac.at/zid/hand/>
  - →Internet-Handbuch von Hubert Partl
  - →weitere Informationen über das Internet
  - →Informationen über WWW und Clients
  - →Java-Einführung von Hubert Partl
  - →XML Kurz-Info von Hubert Partl
  - →WML-Einführung von Hubert Partl
  - →sonstige Handbücher
- englischsprachige Informationen:
  - →World Wide Web FAQ von Thomas Boutell auf  
→<http://www.boutell.com/faq/>
  - →HTML-Primer von Mark Andreessen et al. am NCSA
  - →Bare Bones Guide to HTML (HTML 2 und HTML 3 in Übersicht von Kevin Werbach, USA)
  - →Style-Guide des CERN bzw. W3-Consortium
  - →Markup-Guide des CERN bzw. W3-Consortium
  - →Provider-Guide des CERN bzw. W3-Consortium
  - andere Dokumentationen am →W3-Consortium und am →NCSA
  - →Yale Style Guide (USA)
  - →The HTML Writers Guild (HWG, USA)
- →W3-Consortium auf →<http://www.w3.org/>

- →allgemeine Informationen über WWW, HTTP, HTML, XML, WWW-Server, Clients, Editoren und Hilfsprogramme
- →Spezifikationen von HTML, MathML, XHTML, XML und von CSS, XSL, SVG u.a.
- →CML auf →<http://www.xml-cml.org/>
- →PHP auf →<http://www.koehntopp.de/php/>
- →Apache Web-Server auf →<http://www.apache.org/>
- →HTML-Help der Web Design Group WDG auf →<http://www.htmlhelp.com/>
  - →Hinweise zu Web-Design mit HTML und Style-Sheets
  - →Referenz-Dokumentationen
  - →Wilbur = HTML 3.2 (1996/97)
  - →Character Sets
  - →CSS1 Style Sheets
  - →Design-Hinweise
- HTML-Prüfprogramme (Validatoren):
  - →HTML-Validator des W3-Consortiums auf →<http://validator.w3.org/>
  - →HTML-Validator bei HTMLhelp
  - →CSSCheck für Style Sheets
  - →Liste von weiteren HTML-Validatoren
- Newsgruppen:
  - →[comp.infosystems.www.authoring.html](mailto:comp.infosystems.www.authoring.html)
  - →[comp.infosystems.www.authoring.site-design](mailto:comp.infosystems.www.authoring.site-design)
  - →[comp.infosystems.www.servers.unix](mailto:comp.infosystems.www.servers.unix)
  - →[de.comm.infosystems.www.authoring.misc](mailto:de.comm.infosystems.www.authoring.misc)
  - →[de.comm.infosystems.www.servers](mailto:de.comm.infosystems.www.servers)
- →WWW-Server in aller Welt
- →Suchhilfen in aller Welt
- →Webring der deutschen Webmaster

### 9.3 Liste der HTML-Befehle

Alphabetische Liste der in der vorliegenden HTML-Einführung beschriebenen HTML-Befehle ("Tags"):

→<a> →<applet> →<area>  
→<b> →<big> →<blink> →<blockquote> →<body> →<br>  
→<center> →<code>  
→<dd> →<div> →<dl> →<dt>  
→<em>  
→<font> →<form> →<frame> →<frameset>  
→<head> →<hr> →<html> →<h1> →<h2> →<h3>  
→<i> →<img> →<input>  
→<li> →<link>  
→<map> →<meta>  
→<noframes> →<noscript>  
→<object> →<ol>  
→<p> →<param> →<pre>  
→<q>  
→<script> →<small> →<strong> →<style> →<sub> →<sup>  
→<table> →<tbody> →<td> →<tfoot> →<th> →<thead> →<title> →<tr>  
→<tt>  
→<u> →<ul>  
→<!doctype> →<!-- -->

---

**Happy Web-Weaving!**

---