HTML - Einführung

Hypertext Markup Language
die Sprache des World Wide Web

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

Version 1.4
August 1996

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/htmleinf/`" 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 7 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/htmleinf/`

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

- Entweder Sie speichern die einzelnen HMTL-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. 110 kBytes) und
  →`ftp://mail.boku.ac.at/www/htmleinf.ps.gz` (ca. 300 kBytes)
  oder die mit PKZIP komprimierten MS-DOS-Files
  →`ftp://mail.boku.ac.at/www/hein.zip` (ca. 120 kBytes, ohne PostScript)

Inhaltsverzeichnis

1 Die Grundlagen **1**

1.1 WWW – Was ist das? 2
1.2 Inhalt des HTML-Files 6
1.2.1 Was darf ich im WWW veröffentlichen? 6
1.2.2 Was soll ich im WWW veröffentlichen? 6
1.2.3 Frisch geplant ist halb gewonnen! 7
1.3 Inhalt und Form 8
1.4 Richtige HTML 9
1.4.1 Was ist richtig? 9
1.4.2 Weltweite Zusammenarbeit oder Firmenabhängigkeit 9
1.5 Format der Markup-Befehle (HTML-Tags) 10
1.6 Aufbau eines HTML-Files 12
1.7 Organisation der HTML-Files 13
1.7.1 Aufteilung der Information auf einzelne HTML-Files 13
1.7.2 Filenamen und Directories 14
1.8 Wie kann ich mein HTML-File im WWW veröffentlichen? 14
1.8.1 Erstellen des HTML-Files 14
1.8.2 Testen und Validieren des HTML-Files 15
1.8.3 Abspeichern des HTML-Files 15
1.8.4 Bekanntmachen des HTML-Files 16
1.8.5 Aktualisieren der Informationen 17
1.8.6 Löschen eines HTML-Files 17

2 Text-Elemente **18**

2.1 Aufbau des HTML-Files <head> <title> <body> 18
2.2 Absatz (paragraph) <p> und Zeilenumbruch 19
2.3 Zeilenwechsel (line break)
 20
2.4 Seitenwechsel (page break) 20
2.5 Buchstaben und Sonderzeichen (entity) 21
2.6 Hervorgehobene Wörter (emphasis)   22
2.7 Hervorgehobene Absätze und Zitate (quote) <blockquote> <bq> <note> 23
2.8 Überschriften (heading) <h1> <h2> <h3> 23
2.9 Listen und Aufzählungen 24
2.9.1 Nicht numerierte Liste (unordered list) 24
2.9.2 Numerierte Liste (ordered list) 25
2.9.3 Liste von Beschreibungen (definition list) <dl> 26
2.10 Formatierte Texteingabe (preformatted) <pre> 27
2.11 Tabelle (table) <table> 28
2.12 Mathematik und Chemie <sub> <sup> 30

3 Hypertext-Links **31**
3.1 Verweise zu anderen Informationen <a href> . . . 32
3.2 URL (Uniform Resource Locator) . . . 33
3.2.1 Absolute URLs im WWW . . . 34
3.2.2 Relative URLs im WWW . . . 35
3.2.3 URLs für andere Internet-Services . . . 35
3.3 Listen von Verweisen . . . 36
3.4 Markierungen innerhalb eines HTML-Files <a name> . . . 36
3.5 Inhaltsverzeichnisse . . . 37

4 Bilder und Töne **38**
4.1 Bilder – ja oder nein? . . . 38
4.2 Inline-Bilder <img> <object> . . . 39
4.3 Externe Bilder, Töne, Filme . . . 41
4.4 Kleine und große Bilder (thumbnails) . . . 42

5 Layout und Spezialeffekte **43**
5.1 Schönes Layout mit HTML – wie geht das? . . . 44
5.1.1 Logisches Markup und Layout-Hinweise . . . 44
5.1.2 Wenn die Glocken locken... . . . 44
5.1.3 Norm oder nicht Norm, das ist hier die Frage . . . 45
5.2 Klassen (class) und Style-Sheets <style> . . . 45
5.3 Schrift . . . 48
5.3.1 Schriftarten . . . 48
5.3.2 Schriftgrößen . . . 48
5.4 Farben   <body> <font> class <style> . . . 49
5.5 Anordnung (align) . . . 51
5.5.1 Linksbündig, rechtsbündig, zentriert . . . 52
5.5.2 Unten, oben, neben Bildern . . . 54
5.6 Abstände . . . 56
5.6.1 Horizontale Abstände . . . 56
5.6.2 Einrückungen . . . 57
5.6.3 Vertikale Abstände . . . 57
5.7 Trennlinien (horizontal rule) <hr> . . . 57
5.8 Numerierungen in Listen <ol> . . . 58
5.9 Frames <frameset> <frame> <noframes> . . . 60
5.10 Navigationshilfen <link> . . . 62
5.11 Interaktion mit dem Benutzer . . . 64
5.11.1 CGI-Prozeduren (Common Gateway Interface) . . . 64
5.11.2 Zugriffe zählen . . . 65
5.11.3 Formulare <form> . . . 66
5.11.4 Index-Suche <isindex> . . . 68
5.11.5 Image-Maps (ismap <map> usemap) . . . 69
5.11.6 Electronic Mail (mailto) . . . 72

6 Geschichte und Geschichten **73**
6.1 Vom Elektronengehirn zum World Wide Web 73
6.2 Von der Textverarbeitung zur Hypertext Markup Language 75

7 Die roten Fäden der Ariadne **77**
7.1 Wegweiser 77
7.1.1 Ich möchte den kompletten Text lesen. 77
7.1.2 Ich möchte das komplette Handbuch auf Papier ausdrucken. . 78
7.1.3 Ich möchte einen bestimmten HTML-Befehl nachschlagen. . . 78
7.1.4 Ich möchte kurz erfahren, wofür WWW und HTML überhaupt gut sind. 78
7.1.5 Ich kenne WWW und möchte alles über HTML erfahren. . . 78
7.1.6 Ich möchte rasch eine kurze Information im WWW veröffentlichen. 79
7.1.7 Ich möchte eine umfangreiche Information gut strukturiert im WWW veröffentlichen. 79
7.1.8 Ich möchte eine Liste von "heißen" Verbindungen zusammenstellen. 79
7.1.9 Ich möchte, daß meine Informationen bunt und wild aussehen 79
7.1.10 Ich möchte, daß meine Informationen von vielen Leuten gelesen werden. 80
7.1.11 Ich möchte mehr über die Hintergründe von WWW und HTML erfahren. 80
7.1.12 Ich möchte etwas anderes. 80
7.2 Referenzen 81
7.3 Liste der HTML-Befehle 82

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
- →Organisation der HTML-Files
  - →Aufteilung der Information auf einzelne HTML-Files
  - →Filenamen und Directories
- →Wie kann ich mein HTML-File im WWW veröffentlichen?
  - →Erstellen des HTML-Files
  - →Testen und Validieren
  - →Abspeichern des HTML-Files
  - →Bekanntmachen des HTML-Files
  - →Aktualisieren der Informationen
  - →Löschen eines HTML-Files
- siehe auch →Geschichte und →Referenzen

1.1 WWW – Was ist das?

Ein kleines Lexikon zu WWW und HTML

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 gespeichert. Außerdem können auch Bilder, Töne und beliebige sonstige Files über das WWW übertragen werden, und es können auch Benutzer-Eingaben von Programmen, die auf dem →WWW-Servern laufen, verarbeitet werden (Formulare, Suchvorgänge u.a.).

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)

Gopher : Gopher ist ein Informationssystem, das einen bequemen Zugriff auf Informationen, die auf vielen verschiedenen Computern gespeichert sind, in einer hierarchischen Struktur von Directories (Menüs) und Files (Informationen) ermöglicht. Der Zugriff erfolgt nach dem Prinzip von →Server und →Client über das →Internet mit dem Gopher-Protokoll. Text-Informationen werden auf den →Gopher-Servern in der Form von einfachen ASCII-Files gespeichert. Außerdem können auch Bilder, Töne und beliebige sonstige Files mit Gopher übertragen werden, und neuere Gopher-Server können auch Benutzer-Eingaben verarbeiten (Ask-Blocks).
Gopher wurde an der Universität von Minnesota entwickelt. Der Name kommt von der Beutelratte, dem "Nationaltier" und Spitznamen von Minnesota.

Hyper-G : Hyper-G ist ein Informationssystem, das Eigenschaften von →Gopher und →WWW vereinigt. Es unterstützt hierarchische Strukturen von →Hypertext- und →Hypermedia-Files sowie Stichwort- und Volltext-Suchen in diesen Strukturen ("Collections").
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.

Server : Server (Verkäufer, Bedienender) sind die Computer, auf denen die Informationen gespeichert sind. WWW-Server laufen meistens auf Unix-Rechnern und werden auch als →HTTP-Dämonen bezeichnet. Es gibt mehrere solche Software-Produkte, sowohl public domain als auch kommerziell.

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.

Web-Browser : Als Web-Browser bezeichnet man →Client-Programme für den Zugriff auf →WWW-→Server. Es gibt viele verschiedene solche Programme, sowohl public domain als auch kommerziell. Manche wie z.B. Lynx unterstützen nur die

Textausgabe und funktionieren in einem einfachen Zeilenmodus mit Tastatursteuerung, die meisten wie z.B. Mosaic und Netscape unterstützen auch Bilder und funktionieren nur auf einer graphischen Benutzeroberfläche mit Maus oder Touch-Screen. Spezielle Browser-Programme können die Informationen auch in Blindenschrift oder akustisch (als gesprochene Texte) oder in Form von dreidimensionalen Virtual-Reality-Szenen darstellen.
Die meisten Web-Browser unterstützen nicht nur den Zugriff auf →WWW-Server sondern auch auf →Gopher-Server und auf andere →Internet-Services wie Telnet und FTP sowie mit Einschränkungen auch Electronic Mail und Usenet News.
(siehe auch →Geschichte und →Referenzen)

Web-Page : Als Web-Page (Netz-Seite) bezeichnet man ein im →WWW veröffentlichtes →HTML-File.

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 →"**Gopher**" für den Zugriff auf Informationssysteme in aller Welt.
(siehe auch →Geschichte und →Referenzen)

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, Telephon, Video-Konferenzen, Kabel-Fernsehen, Video-Verleih, Tele-Arbeit, Tele-Shopping und ähnlichen Services.
(siehe auch →Geschichte)

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.

Hypermedia : Mit Hypermedia bezeichnet man Multi-Media-Systeme (Texte, Bilder und Töne) mit Querverweisen wie bei →Hypertext.

HTTP (Hypertext Transfer Protocol) : HTTP ist das Protokoll, nach dem die Informationen zwischen →WWW-Servern und →WWW-Clients über das →Internet übertragen werden.

HTML (Hypertext Markup Language) : HTML ist das Format, in dem die Text- und →Hypertext-Informationen im →WWW gespeichert und übertragen werden. Der offiziell gültige Standard ist HTML 2.0, aber neue, erweiterte Versionen HTML 3.x werden vom →W3-Consortium laufend in mehreren Einzelschritten entwickelt und sind bereits teilweise als De-facto-Standard verfügbar.
HTML ist eine "Content-based Markup Language" mit →SGML-Syntax. HTML unterstützt ein *logisches Markup*, bei dem die logische Bedeutung der Textteile so festgelegt wird, daß sie vom jeweiligen →Web-Browser in der für den Benutzer (→Client) optimalen Form dargestellt werden können.
HTML-Files können mit einfachen Text-Editoren oder mit speziellen Hilfsprogrammen erstellt werden.
(siehe auch →Geschichte und →Referenzen)

SGML (Standard Generalized Markup Language) : SGML ist ein System und eine Sprache, in der Markup-Sprachen wie z.B. →HTML definiert werden können.
(siehe →Geschichte)

PostScript : PostScript ist eine von der Firma Adobe entwickelte Seitenbeschreibungssprache. Im Gegensatz zum *logischen* Markup von →HTML, das eine Anpassung der Darstellung an den →Client ermöglicht, wird mit PostScript das *Aussehen* der Dokumente (Texte und Bilder) in allen Details festgelegt.

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.

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.

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.

CGI (Common Gateway Interface) : CGI-Programme sind Programme oder Shell-Scripts, die auf dem →WWW-Server laufen, eventuelle Daten-Eingaben des →Clients verarbeiten und die Ergebnisse im →HTML-Format an den →Client-Rechner senden.

Java : →WWW unterstützt standardmäßig nur den passiven Zugriff von →Clients auf die auf den →Servern gespeicherten Informationen und die Ausführung von →CGI-Programmen auf dem →Server-Computer.
Die plattformunabhängige und mit den notwendigen Sicherheitsmechanismen ausgestattete Programmiersprache Java ermöglicht zusätzlich die Ausführung von bestimmten Aktionen (Programmen, "Applets") auf dem →Client-Rechner
Java wurde von der Computer-Firma Sun entwickelt und ist bereits teilweise verfügbar. Der Name stammt von einer amerikanischen Bezeichnung für Kaffee.

JavaScript : JavaScript ist eine von der Firma Netscape erfundene Sprache zur Ausführung von bestimmten Aktionen innerhalb des Netscape-Browsers. Im Gegemsatz zu →Java ist diese Sprache weder plattformunabhängig noch mit den notwendigen Sicherheitsmechanismen ausgestattet.

Suchhilfen : Die Gesamtheit der über das →Internet und im →World Wide Web verfügbaren Informationen und Services ist 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 ("Spiders"), 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 (siehe →Referenzen).

Lesezeichen : Lesezeichen (meist als "Bookmarks" oder "Hotlists" 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.

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 HTML-Files nach allen Regeln der Kunst schreiben, 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-Designern 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: Manche unterstützen nur eine einfache Textausgabe im Zeilenmodus mit einer einheitlichen Schrift, viele andere unterstützen auch Bilder und Verzierungen sowie verschiedene Schriftarten und -größen. Die Steuerung erfolgt bei manchen Browsern mit einer Tastatur, bei anderen mit einer Maus oder einem Touch-Screen. Es sind auch Browser denkbar, die die Informationen in Blindenschrift oder akustisch (als gesprochene Texte) oder in Form von dreidimensionalen Virtual-Reality-Szenen darstellen.

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 2 und andererseits diejenigen Teile von →HTML 3, 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...

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 jüngsten Erfahrungen lehren:

1993 wurde für Informationssysteme hauptsächlich Gopher verwendet.

1994 wurde Gopher von Mosaic fast vollständig verdrängt.

1995 wurde Mosaic von Netscape fast vollständig verdrängt, und daneben brachten auch viele andere Firmen (einschließlich der Giganten IBM und Microsoft) ihre eigenen, teilweise inkompatiblen Web-Brwowser heraus.

1996 warfen die Konkurrenten Netscape und Microsoft alle paar Monate neue Versionen ihrer Web-Browser auf den Markt, die immer ausgefeiltere Layout-Spezialitäten vor allem in Hinblick auf hochauflösende PC-Bildschirme unterstützten (Farben, Tabellen, Frames, Style-Sheets). Gleichzeitig wurde mit WebTV eine Möglichkeit geschaffen, Web-Pages auf Fernseh-Bildschirmen zu betrachten, die eine wesentlich geringere Auflösung als PC-Bildschirme aufweisen.

Wenn Ihre HTML-Files auch noch nächstes Jahr brauchbar sein sollen und wenn Sie nicht einen großen Teil der Interessenten vom Zugriff auf Ihre Informationen ausschließen wollen, dann müssen Sie als Autor auf solche firmenspezifische "Extrawürste" verzichten und möglichst nur die HTML-Befehle einsetzen, die auf *allen* Browsern ein richtig lesbares Ergebnis bewirken (→De-facto-Norm).

Beispiele für "Extrawürste", die nicht (oder noch nicht) von allen gängigen Web-Browsern richtig verarbeitet werden, sind <animate>, <applet> <blink>, <bgsound>, → , →<embed>, →<font>, →<frame>, →<frameset>, <marquee> →<object>, <script> →<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 →HTML baut auf der Syntax von →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. Wenn die Argumente Sonderzeichen enthalten, müssen sie in Anführungszeichen (Quotes) eingeschlossen werden:

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

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 →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 →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 →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 wenigen Fällen darf der End-Tag weggelassen werden, so ist z.B. die explizite Angabe von `</p>` in den obigen Beispielen nicht unbedingt notwendig.

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

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 →Textelemente, →Hypertext-Links, →Bilder und Töne.

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

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

Da sie nicht wissen können, über welche →Hypertext-Links, →Bookmarks oder →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>` dürfen (im Gegensatz zu `<h1>`) nur normale ASCII-Buchstaben stehen, keine →Entities (deutsche Umlaute etc.) und keine HTML-Befehle.

Manche Web-Browser haben eine Funktion für das Senden von Kommentaren an den Autor des HTML-Files. Zu diesem Zweck und auch aus anderen praktischen Gründen ist es notwendig, daß die Mail-Adresse des Autors als →Mailto-URL im `<link>`-Befehl angegeben wird.

Unabhängig davon ist es empfehlenswert, am Ende des lesbaren Textes (Body) eines jeden HTML-Files den Namen und die Mail-Adresse des Autors, das Datum der Erstellung oder der letzten Aktualisierung, und eventuell auch ein Hypertext-Link auf die WWW-Start-Seite (Home-Page) der Institution anzugeben.

Als erste Zeile des Files (vor dem Befehl `<html>`) kann man die →HTML-Version mit einem →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

- →Aufteilung der Information auf einzelne HTML-Files
- →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**" →abgespeichert werden.

Unter Betriebssystemen, die wie MS-DOS nur 3 Zeichen lange Extensions erlauben, müssen die Files 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 mein HTML-File im WWW veröffentlichen?

Für die Veröffentlichung eines HTML-Files über das WWW sind drei Schritte notwendig:

1. Sie müssen das HTML-File →erstellen, korrekturlesen, →testen und validieren.
2. Sie müssen das HTML-File auf einem →WWW-Server →abspeichern.
3. Sie müssen die Verfügbarkeit des 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.

1.8.1 Erstellen des HTML-Files

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

Für das Erstellen eines HTML-Files genügt ein einfacher Text-Editor. Wenn Sie es mit einem Textverarbeitungs-Programm wie MS-Word erstellen, müssen Sie das File als reines ASCII-File speichern ("nur Text").

Es gibt auch verschiedene Hilfsprogramme, mit denen Web-Pages menügesteuert am Bildschirm erstellt sowie Win-Word-Files, RTF-Files, LaTeX-Files und andere File-Formate mehr oder weniger automatisch in HTML-Files umgewandelt werden können (siehe die →Referenzen).

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

1.8.2 Testen und Validieren des 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. Mosaic oder Netscape), oder zumindest mit verschiedenen persönlichen Einstellungen (z.B. mit und ohne Bilder) und verschiedenen Fenstergrößen.

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 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).

1.8.3 Abspeichern des 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 Software-Produkte von Institutionen wie CERN und NCSA sowie kostenpflichtige Produkte von Software-Firmen 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. Mosaic 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.

1.8.4 Bekanntmachen des 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".

1.8.5 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

1.8.6 Löschen eines 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 selbt 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.

2. Text-Elemente

- →Vorwort
- →Aufbau des HTML-Files <head> <title> <body>
- →Absätze <p> und Zeilenumbruch
- →Zeilenwechsel
- →Seitenwechsel
- →Buchstaben und Sonderzeichen
- →hervorgehobene Wörter
- →hervorgehobene Absätze <blockquote> <bq> <note>
- →Überschriften <h1> <h2> <h3>
- →Listen und Aufzählungen
  - →nicht numerierte Listen
  - →numerierte Listen
  - →Beschreibungen <dl>
- →formatierte Texteingabe <pre>
- →Tabellen <table>
- →Mathematik und Chemie <sub> <sup>
- Spezialeffekte:
  - →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) und Zeilenumbruch

Im einfachsten Fall besteht ein HTML-File zwischen →<body> und →</body> nur aus laufendem Text, mit →Entities für Umlaute und Sonderzeichen und mit eingestreuten Befehlen für den Beginn eines neuen Absatzes.

Beispiel:

– – – *Die Eingabe von*

```
<p>
Das ist ein Absatz.
Der Zeilenumbruch erfolgt automatisch.
Zeilenwechsel,
einfache Leerstellen und     mehrfache     Leerstellen
bewirken alle die gleichen Wortabst&auml;nde.
<p>
Das ist ein neuer Absatz...
```

– – – *bewirkt eine Darstellung wie*

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

Das ist ein neuer Absatz...

– – –

Die meisten →Web-Browser stellen Absätze durch eine Leerzeile dar, seltener durch Einrücken der ersten Zeile wie im Buchdruck. Manche Web-Browser erzeugen bei mehrfachen -Befehlen mehrfache Leerzeilen, die meisten ignorieren jedoch leere Absätze und stellen alle Absätze mit einheitlichen Abständen dar.

Mit →Align-Parametern kann angegeben werden, wie der Absatz dargestellt werden soll (rechtsbündig, linksbündig, zentriert). Absätze *ohne* automatischen Zeilenumbruch kann man mit →<pre> erhalten.

Ein -Befehl zum Beenden von Absätzen ist *nicht* nötig, jeder neue Absatz und jede neue →Überschrift oder →Liste und dergleichen beendet automatisch den vorherigen Absatz.

2.3 Zeilenwechsel (line break)

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

Hänschen klein
ging allein
in das World Wide Web hinein.

– – –

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 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 Papier-Seite erfüllen, den Befehl →`<hr>` bzw. die Befehlsfolge `</p><hr><p>` verwenden.

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):
`<` für das Kleiner-Zeichen <
`>` für das Größer-Zeichen >
`&` 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:
`ä` für ä (Umlaut-a)
`Ä` für Ä (Umlaut-A)
`ö` für ö (Umlaut-o)
`Ö` für Ö (Umlaut-O)
`ü` für ü (Umlaut-u)
`Ü` für Ü (Umlaut-U)
`ß` für ß (scharfes s, s-z-Ligatur)

Für andere europäische Sprachen gibt es Entities wie z.B.
`é` für é (e mit Acute-Akzent)
`à` für à (a mit Grave-Akzent)
`ô` für ô (o mit Circumflex-Akzent)
`ç` für ç (c mit Cedille)
`ñ` für ñ (n mit Tilde)
`å` für å (a mit Ring)
und weitere Entities mit analog gebildeten Namen.

Manche Web-Browser unterstützen auch weitere Sonderzeichen und Spezialfunktionen wie z.B.
`°` für das Grad-Symbol ° (degree)
`©` für das Copyright-Symbol ©
`§` für das deutsche Paragraphen-Zeichen § (section)
`¶` für das amerikanische Absatz-Zeichen ¶ (paragraph)
` ` für eine Leerstelle, bei der kein Zeilenwechsel erfolgen darf (non breaking space)
`` 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 `{` 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.

Solange es noch Web-Browser gibt, die nicht alle Entities richtig darstellen, ist es empfehlenswert, für ausgefallene Symbole einfache Text-Abkürzungen zu verwenden, wie z.B.
`(C)` für das Copyright-Symbol, oder
`Par.` für das Paragraphen-Zeichen.

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

– – –

2.6 Hervorgehobene Wörter (emphasis)

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> <bq> <note>

In →HTML 2 werden zwischen `<blockquote>` und `</blockquote>` stehende Absätze 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.

In →HTML 3 werden statt `<blockquote>` zwei verschiedene Markups unterstützt:
`<bq> ... ... </bq>` für Zitate,
`<note> ... </note>` für Anmerkungen (notes), Hinweise und dergleichen.
Im Befehl `<note>` kann durch die Angabe von →Klassen und durch die Verwendung von →Style-Sheets 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:

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

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

– – –

2.9 Listen und Aufzählungen

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

2.9.1 Nicht numerierte Liste (unordered list)

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 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 f&uuml;r alle Arten
von Aufz&auml;hlungen.
<li>Listen k&ouml;nnen auch geschachtelt werden:
<ul>
<li>Elemente der inneren Liste
werden meist weiter einger&uuml;ckt und/oder
mit anderen Symbolen versehen.
<li>Hier ist das zweite Element der inneren Liste.
</ul>
<li>Hier ist ein weiteres (letztes) Element
der &auml;u&szlig;eren 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.

– – –

2.9.2 Numerierte Liste (ordered list)

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. In →HTML 3 kann man die Numerierung durch →Parameter wie `type`, `start` und `value` in den Befehlen `<ol>` und `<ol>` beeinflussen (siehe →Spezialeffekte); diese Angaben werden aber nicht von allen Web-Browsern unterstützt.

Beispiel:

– – – Die Eingabe von

```
Eine numerierte Liste:
<ol>
<li>Numerierte Listen eignen sich f&uuml;r Aufz&auml;hlungen,
bei denen die genaue Reihenfolge der Elemente wichtig ist.
<li>Listen k&ouml;nnen auch geschachtelt werden:
<ol type=a>
<li>Elemente der inneren Liste
werden meist weiter einger&uuml;ckt und
separat numeriert.
<li>Hier ist das zweite Element der inneren Liste.
</ol>
<li>Hier ist ein weiteres (letztes) Element
der &auml;u&szlig;eren Liste.
</ol>
```

– – – bewirkt eine Darstellung wie

Eine numerierte Liste:

1. Numerierte 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 numeriert.
   (b) Hier ist das zweite Element der inneren Liste.
3. Hier ist ein weiteres (letztes) Element der äußeren Liste.

– – –

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 manchen Web-Browsern kann durch die Angabe von `<dl compact>` ein kompakterers 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
Stechm&uuml;cke bezeichnet.
<dt>Gemse:
<dd>ein gro&szlig;es Tier,
das in den Alpen von Menschen gejagt wird.
Die Gemse
wird auf Grund ihrer Losung manchmal f&auml;lschlich 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.

– – –

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 Zeilenwechseln. 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 `<` und `>` bzw. `&` 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>

In →HTML 2 werden Tabellen und Tabulatoren *nicht* unterstützt, einfache Tabellen können jedoch mit Hilfe von →`<pre>` erzeugt werden.

In →HTML 3 gibt es eigene 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` für linksbündig
`align=right` für rechtsbündig
`align=center` für zentriert
`char=","` für Dezimalkomma

Beispiel:

– – – Die Eingabe von

```
<table border>
<tr><td align=right>11111000000
    <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

– – –

`<table>` wird (noch) nicht von allen →Web-Browsern unterstützt, es kann daher sinnvoll sein, stattdessen →`<pre>` zu verwenden. Eine andere Möglichkeit ist es, zwar `<table>` zu verwenden, aber vor und nach der Tabelle jeweils →`<p>`, am Ende jedes Tabellenfeldes eine Leerstelle und im letzten Feld jeder Tabellenzeile jeweils →`<br>` einzufügen, damit das Ergebnis auch mit Web-Browsern, die die Tabellen-Befehle ignorieren, brauchbar aussieht.

Beispiel:

– – – *Die Eingabe von*

```
Don Carlos
<table>
<tr><td>Philipp II.:
    <td>Nicolai Ghiaurov
<tr><td>Elisabeth von Valois:
    <td>Mirella Freni
</table>
```

– – – *bewirkt auf Browsern, die Tabellenbefehle ignorieren, eine Darstellung wie*

Don Carlos Philipp II.: Nicolai Ghiaurov Elisabeth von Valois: Mirella Freni

– – – *Die Eingabe von*

```
Don Carlos
<p><table>
<tr><td>Philipp II.:
    <td>Nicolai Ghiaurov <br>
<tr><td>Elisabeth von Valois:
    <td>Mirella Freni <br>
</table>
```

– – – *bewirkt auf Browsern, die Tabellenbefehle ignorieren, eine Darstellung wie*

Don Carlos

Philipp II.: Nicolai Ghiaurov
Elisabeth von Valois: Mirella Freni

– – –

2.12 Mathematik und Chemie <sub> <sup>

→HTML 3 enthält Befehle für die Darstellung von mathematischen und chemischen Formeln. Die einfachsten sind die auch im normalen Text verwendbaren Befehle für 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

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

– – – Die Eingabe von

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

– – – bewirkt eine Darstellung wie

$$SO_3 + H_2O = H_2SO_4$$

– – –

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>
  - →Index-Suche <isindex>
  - →Image-Maps (ismap)
  - →Electronic Mail (mailto)
  - →Navigationshilfen <link>

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 manchmal 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 Plazierung 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 nur für manche 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.

– – –

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/htmleinf/hein3.html#url`
bedeutet einen Sprung an die Stelle "url" im File "hein3.html" im Directory "htmleinf" 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.

Manche Web-Browser tolerieren es, wenn der abschließende Schrägstrich fehlt, bei anderen Web-Browsern führt das aber zu verschiedenen Fehlern. Der Schrägstrich nach dem Hostnamen bzw. nach der Portnummer muß deshalb *immer* angegeben werden.

`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-Belastung). In vielen Fallen 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.

`file:///directory/filename`
bedeutet ein auf dem lokalen Rechner im angegebenen Directory liegendes File.
Im Fall von PCs sind dabei Laufwerk und Directory nicht in der DOS-Schreibweise sondern in der 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 →Gopher-Server, →FTP-Server oder andere →Internet-Services verweisen.

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

Beispiele:

`gopher://hostname/11/directory`
für ein Directory auf einem →Gopher-Server.

`gopher://hostname/00/directory/filename`
für ein File auf einem →Gopher-Server.

`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 →Spezialeffekte).

Bestimmte →Web-Browser verwenden intern Pseudo-URLs für bestimmte Programmfunktionen (z.B. `"about:"` für die Ausgabe von Help-Texten oder `"newsrc:"` für die Subskription von Newsgruppen). Solche Pseudo-URLs dürfen klarerweise *nur* innerhalb dieses Web-Browsers und →*nicht* in HTML-Files verwendet werden.

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 3 kann man Sprungmarken mit `id="marke"` statt mit `name=` vereinbaren, 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...
```

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.

In →HTML 3 geht das noch einfacher mit Markierungen der Form

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

4. Bilder und Töne

- →Vorwort
- →Bilder – ja oder nein?
- →Inline-Bilder <img> <object>
- →externe Bilder, Töne, Filme
- →Kleine und große Bilder (thumbnails)
- Spezialeffekte:
  - →Anordnung (align)
  - →Image-Maps (ismap <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.

Es ist daher 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 genau überlegen, in welchen Fällen ein buntes und ansprechendes Aussehen wichtig ist, um Interessenten anzulocken, und in welchen Fällen der reine Informationsgehalt wichtiger ist. Der Unterschied ist so einfach wie die Wahl zwischen Ansichtskarten und Briefen...

4.2 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
`<img src="URL" alt="Text">`
eingefügt.

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,
`xxxx.xbm` für Graphiken im X-Bitmap-Format.
Dies sind die Bildformate, die von den meisten (wenn auch nicht allen) →Web-Browsern unterstützt 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.

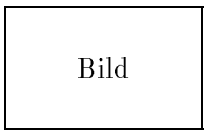
Mit dem Parameter `alt=` muß der Text angegeben werden, der an Stelle des Bildes am Bildschirm erscheinen soll, falls der Benutzer einen Web-Browser verwendet, der keine Bilder darstellt. Wenn der Parameter `alt=` nicht angegeben wird, geben die meisten Web-Browser an Stelle des Bildes den Textstring `"[IMAGE]"` oder ein allgemeines Bild-Symbol (z.B. ein buntes Rechteck) aus.

Beispiel:

– – – Die Eingabe von

```
<img src="wwwlogo.gif" alt="WWW">
```

– – – bewirkt eine Darstellung wie



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

– – –

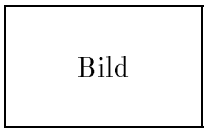
In vielen Fällen ist es günstiger, den erklärenden Text nicht nur *anstelle* des Bildes sondern *immer* neben dem Bild auszugeben und den alt-Text dafür auf den Leerstring zu setzen.

Beispiel:

– – – Die Eingabe von

```
<img src="austria.gif" alt=""> &Ouml;sterreich
```

– – – *bewirkt eine Darstellung wie*



Österreich

– – –

Der mit `alt=` angegebene Text kann keine HTML-Befehle enthalten. In →HTML 3 ist deshalb als Alternative zum Befehl
`<img src="URL" alt="Text">`
auch ein wesentlich mächtigerer Befehl der Form
`<object data="URL"> Text </object>`
vorgesehen, bei dem im Text auch weitere HTML-Befehle verwendet werden können. Dieser `<object>`-Befehl wird von den meisten Browsern noch nicht unterstützt.

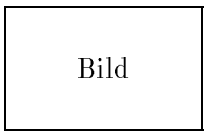
Mit speziellen →Align-Parametern kann die Position des Bildes und des nachfolgenden Textes verändert werden.

Beispiel:

– – – *Die Eingabe von*

```
<p align=center>
<img src="austria.gif" alt="" align="top"> &Ouml;sterreich
<p>
```

– – – *bewirkt eine Darstellung wie*



Österreich

– – –

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".

4.3 Externe Bilder, Töne, Filme

Im Gegensatz zu →Inline-Bildern, die automatisch innerhalb von Text-Files dargestellt werden, sollen größere Bilder sowie Töne, Filme und dergleichen extern, d.h. mit Hilfe von →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 →"Thumbnail") am Bildschirm und bekommt das Bild bzw. die Tonfolge oder den Video-Film erst dann auf seinen →Client übertragen, wenn er dieses Link →"auswählt".

Darin ist "URL" der URL des Files, das das Bild oder die Tonfolge oder den Film enthält. Im einfachsten Fall (→relativer URL) besteht dieser URL nur aus dem Filenamen 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.xbm` für Graphiken im X-Bitmap-Format,
`xxxx.au` für Töne im AU-Format,
`xxxx.wav` für Töne im WAV-Format,
`xxxx.aiff` für Töne im AIFF-Format,
`xxxx.mov` für Video-Filme im QuickTime-Format,
`xxxx.mpg` für Video-Filme im MPEG-Format.

Je nach der Extension wird vom →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*

→Großes Bild ohne Bikini (JPEG, 45k)

– – –

4.4 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.

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.

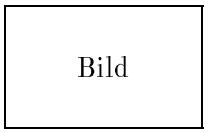
Diese kleinen Bilder sehen meist wie Briefmarken aus und werden im Fachjargon als "Daumennägel" (thumbnails) bezeichnet.

Beispiel:

– – – *Die Eingabe von*

```
<a href="big.jpg"><img src="small.jpg" alt="">
 . . . "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   <body> <font> class <style>
- →Anordnung (align)
  - →linksbündig, rechtsbündig, zentriert
  - →Chaos oder Harmonie
  - →unten, oben, neben Bildern
- →Abstände
  - →horizontale Abstände
  - →Einrückungen
  - →vertikale Abstände
- →Trennlinien <hr>
- →Numerierungen
- →Frames <frameset> <frame> <noframes>
- →Navigationshilfen <link>
- →Interaktion mit dem Benutzer – mehr Leben ins World Wide Web
  - →CGI-Prozeduren
  - →Zugriffe zählen
  - →Formulare <form>
  - →Index-Suche <isindex>
  - →Image-Maps (ismap <map> usemap)
  - →Electronic Mail (mailto)
  - siehe auch die Hinweise von →Stefan Münz zu →Java, PostScript und PDF, VRML und Plugin-Programmen sowie zu →JavaScript

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 in →HTML 3 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 →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 →Suchhilfen 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 würden dieselben Mißverständnisse auftreten.

Solche Spezialeffekte kann man besser mit Hilfe von →Klassen und →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 →De-facto-Norm von HTML, dann sind Ihre Informationen für *alle* Interessenten lesbar,
- oder Sie verwenden unbedacht →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 Web-Seiten 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 →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 →PostScript-File oder noch besser als →normgerechtes HTML-File.

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

→HTML 3 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 `<note class="wichtig">` und `</note>` einschließen und im Style-Sheet angeben, daß alle mit `class="wichtig"` markierten Notes 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 (notes) 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, A:visited { color: dark-red; background: white;
     text-decoration: underline }
A:active { color: dark-red; background: yellow }
.boku    { color: dark-green; background: white }
.wichtig { color: red; background: white;
     font-style: normal; font-weight: bold }
EM { font-style: normal; font-weight: demi-bold }
NOTE { margin-left: 3em; margin-right: 3em }
</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.
<p><note 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!
</note><p>
Genauere Informationen ...
</body>
</html>
```

– – –

Style-Sheets werden derzeit noch nicht von allen Web-Browsern unterstützt.

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>`, `<body>`, `<font>` etc. so kombinieren, daß Sie den gewünschten Effekt auf möglichst vielen Web-Browsern erreichen.

Ein dementsprechend **erweitertes Beispiel** finden Sie in den folgenden drei Files, die Sie separat ansehen und ausprobieren bzw. ausdrucken können:

- das im CSS-File `heinstyl.css` stehende →Style-Sheet
- die im HTML-File `heinstyl.html` stehende →HTML-Eingabe
- die vom HTML-File `heinstyl.html` auf Ihrem Web-Browser bewirkte →Ausgabe

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 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   <body> <font> class <style>

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 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.

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 text="#000000" bgcolor="#FFFFFF"
      link="#990000" vlink="#660000" alink="#FF0000" >
<h1 class=enzian><font color="#0000FF">Enzian</font></h1>
<p>
Wer
<strong class=enzian><font color="#0000FF">Enzian</font></strong>
trinkt, wird schneller
<strong class=enzian><font color="#0000FF">blau</font></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.

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)

- →linksbündig, rechtsbündig, zentriert
- →Chaos oder Harmonie
- →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>
Linksb&uuml;ndig ist die normale Ausrichtung von Abs&auml;tzen.
<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

– – –

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

Vorsicht!

Zentrierte und linksbündige Layouts sollen **niemals vermischt** 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>
<img src="austria.gif" alt="Oesterreich">
```

– – – *bewirkt eine Darstellung wie*

– – –

**Nicht sondern <p align=center> oder **

Manche, aber nicht alle Web-Browser erlauben als Alternative zum Befehlspaar
`<p align=center> ... <p>` bzw.
`<div align=center> ... </div>`
auch das Befehlspaar
`<center> ... </center>`
Dieses Befehlspaar 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 unbekannten Befehl einfach ignorieren!

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

– – – *Die Eingabe von*

```
Ich sagte:
<center>nichts</center>w&uuml;rdiger Herr Direktor!
```

– – – *bewirkt auf vielen Browsern eine Darstellung wie*

Ich sagte:

nichts

würdiger Herr Direktor!

– – – *aber auf den anderen Browsern 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.

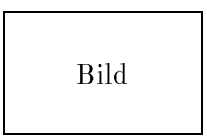
Die Align-Hinweise werden von manchen Web-Browsern befolgt und von anderen →ignoriert.

Beispiele:

– – – *Die Eingabe von*

```
Luft . . .
<img src="small.jpg" alt="[ Bild ]" align=top>
 . . . Koepfe
```

– – – *bewirkt eine Darstellung wie*

Luft . . .  . . . Köpfe

– – – *Die Eingabe von*

```
Wasser . . .
<img src="small.jpg" alt="[ Bild ]" align=middle>
 . . . Arme
```

– – – *bewirkt eine Darstellung wie*

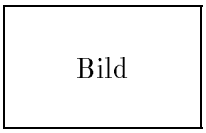
Wasser . . .

. . . Arme

– – – *Die Eingabe von*

```
Erde . . .
<img src="small.jpg" alt="[ Bild ]" align=bottom>
 . . . Beine
```

– – – *bewirkt eine Darstellung wie*



Erde . . . . . Beine

– – –

In →HTML 3 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 läft. Zu diesem Zweck wird im →`<img>` bzw. →`<object>` Befehl der Parameter
`align=left` bzw.
`align=right` angegeben
und nach dem Text, der um das Bild herumlaufen soll, der Befehl
`<→br clear=all>`
damit der nachfolgende Text mit der normalen Textbreite unter dem Bild erscheint.

Beispiel:

– – – *Die Eingabe von*

```
<p>
<img src="wwwlogo.gif" alt="" align=left>
World
<br>
Wide
<br>
Web
<br clear=all>
```

– – – *bewirkt eine Darstellung wie*

World
Wide
Web

– – –

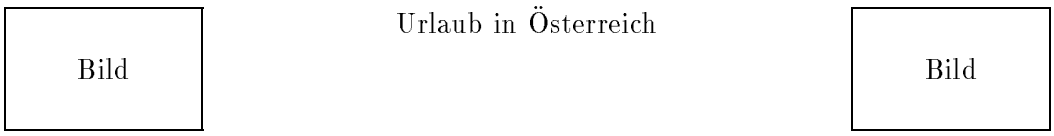
Auf diese Weise kann man kann auch zwei Bilder mit einem dazwischen →zentrierten Text kombinieren.

Beispiel:

– – – *Die Eingabe von*

```
<p>
<img src="austria.gif" alt="" align=left>
<img src="austria.gif" alt="" align=right>
<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 →`<pre>` oder von →`<table>`. In →HTML 3 gibt es auch einen Tabulator-Befehl `<tab>`, der wird jedoch nicht von allen Web-Browsern unterstützt.

Bei manchen Browsern kann man größere Wortabstände auch mit der Entity →` ` 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 →Definition-Listen und →Blockquotes und Notes durch eingerückte Absätze dar, andere aber nicht.

Trotzdem sollte man *immer*, wenn ein untergeordneter Text zu einem übergeordneten Text "dazugehört", eine →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* →Blockquotes oder Notes verwenden, egal ob der Browser dafür Einrückungen oder eine andere, für den Client besser geeignete Darstellung verwendet.

In anderen Fällen kann man Einrückungen mit →`<pre>` erreichen. In →HTML 3 gibt es auch einen Tabulator-Befehl `<tab>`, der wird jedoch nicht von allen Web-Browsern unterstützt.

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. 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.

5.8 Numerierungen in Listen

Standardmäßig werden die Listenelemente in →numerierten Listen mit den natürlichen Zahlen 1, 2, 3 ... numeriert. In →HTML 3 kann man die Numerierung durch die folgenden Parameter in den Befehlen `<ol>` und `<ol>` 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>

Manche, aber nicht alle →Web-Browser unterstützen nicht nur einfache Web-Seiten mit dem normalen Aufbau von →Head und Body, die wie gewohnt im Bildschirmfenster dargestellt werden, sondern auch die Spezifikation von komplizierteren, mehrteiligen Bildschirmstrukturen, die als "Frames" (Rahmen) bezeichnet werden.

In diesem Fall 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 ...
          </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 ignoriert.

Innerhalb von `<noframes>` *muß* 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 – für die Benutzer von Web-Browsern, die keine Frames darstellen. 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 Bildschirmes 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 empfiehlt sich allerdings eher die Verwendung von →Navigationshilfen mit <link>.

Generell ist von der Verwendung von Frames eher →abzuraten.

Warum Sie Frames vermeiden sollten

Das →World-Wide Web vedankt 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 mehrdimesional 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.

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 Anwenderschulung, was meist nur bei einem beschränkten Benutzerkreis oder bei bestimmten firmeninternen Anwendungen in einem "Intranet" erreicht werden kann.

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 (→Body) angezeigt - so ähnlich wie bei →Frames. Die Link-Befehle müssen deshalb stets im →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 mit `rev=` anzugeben, von welcher anderen Information oder von welcher Person die aktuelle Web-Seite stammt. Mit `href=` ist der →URL dieser anderen Information anzugeben.

Die wichtigsten Link-Befehle sind:

`<link rev=made href="mailto:user@host.domain">`
für die →Mail-Adresse des Autors

`<link rel=stylesheet href="URL" type="...">`
für ein →Stylesheet-File

`<link rel=... href="URL">`
für verschiedene Navigataionshilfen

Mit `rel=` ist hier ein Schlüsselwort anzugeben, das die Art der Information angibt, auf die verwiesen wird. Die wichtigsten Möglichkeiten sind:
`toc` für das Inhaltsverzeichnis (table of contents)
`index` für einen Index (Stichwortverzeichnis)
`glossary` für Begriffserkärungen
`help` für Hilfsinformationen
`author` für Informationen über den Autor (Impressum oder Home-Page oder →Mailto-URL)
`copyright` für Copyright-Hinweise
`up` oder `top` für die Haupt-Seite (Start-Seite, Titelseite) dieser Information
`begin` oder `first` für die Start-Seite oder das erste Kapitel
`previous` für die vorherige Web-Seite oder das vorherige Kapitel in der vom Autor vorgeschlagenen Lesereihenfolge
`next` für die nächste Web-Seite oder das nächste Kapitel in der vom Autor vorgeschlagenen Lesereihemfolge
`end` oder `last` für die letzte Web-Seite oder das letzte Kapitel

`<link rel=print href="URL" type="...">`
für ein druckfertiges File dieser Information, mit Angabe des Filetyps (z.B. PostScript), 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 durcheinanderbringen 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="htmleinf.ps" type="application/postscript">
```

– – –

Die Link-Befehle werden von einem Teil der Web-Browser berücksichtigt und von den anderen ignoriert. 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.

5.11 Interaktion mit dem Benutzer

Mehr Leben ins World Wide Web!

- →CGI-Prozeduren
- →Zugriffe zählen
- →Formulare <form>
- →Index-Suche <isindex>
- →Image-Maps (ismap <map> usemap)
- →Electronic Mail (mailto)
- siehe auch die Hinweise von →Stefan Münz zu →Java, PostScript und PDF, VRML und Plugin-Programmen sowie zu →JavaScript

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.

5.11.1 CGI-Prozeduren (Common Gateway Interface)

Eine CGI-Prozedur ist ein 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`

Beispiel:

Eine einfache Unix-Shell-Prozedur, die Datum und Uhrzeit anzeigt:

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

– – –

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

CGI-Prozeduren können in verschiedenen Sprachen geschrieben werden (Shell-Scripts, Perl-Scripts, C-Programme u.a.). Die Detail-Informationen dazu finden Sie in der Dokumentation des jeweiligen →WWW-Servers und in den →Referenzen.

5.11.2 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="$1"
logfile=/a/cern/www.log
n=‘grep -i "$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, denn dann würde sich der File-Inhalt bei jedem Zugriff ändern, die oben beschriebene Cache-Methode nicht mehr wirken und der Zugriff auf das File für die Benutzer um Größenordnungen langsamer und teurer werden. Außerdem ist die Anzahl der File-Übertragungen meist nur für den Autor der Information und den Verwalter des WWW-Servers, aber nicht für die Leser interessant.

5.11.3 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 oder einem Mail-Programm 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;rz
<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

Eingabefeld Ich komme.
Eingabefeld Ich komme nicht.

Name: Eingabefeld

Telefonnummer: Eingabefeld

Eingabefeld

– – –

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".

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 Übertragungs-Methoden ("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.

Hier folgt als **Beispiel** die CGI-Prozedur, mit der die Antwort auf das obige Formular erzeugt wird. Sie verwendet die Methode "get" und den CERN httpd Server mit dem Hilfsprogramm "cgiparse" auf einem Unix-Rechner.

Die vom Formular an den Server gesendete Eingabe hat hier einen Aufbau der Form
`name1=value1&name2=value2&name3=value3`
und muß erst mit "cgiparse" (oder einem anderen geeigneten Hilfsprogramm) in die einzelnen Felder aufgeteilt werden.

```
#!/bin/sh
name="`/usr/local/bin/cgiparse -value wer`"
kommt="`/usr/local/bin/cgiparse -value kommt`"
echo "Content-Type: text/html"
echo ""
echo "<head><title>CGI-Test Geburtstag</title></head><body>"
echo "<h1>CGI-Test Geburtstag</h1>"
echo "<p>Hallo, $name !<p>"
if [ "$kommt" = "ja" ]
   then echo "Es freut mich, da&szlig; Du "
   else echo "Es tut mir leid, da&szlig; Du nicht "
fi
echo "zu meiner Geburtstagsfeier kommen willst. "
echo "<p>(Nein, nicht wirklich, aber "
echo "<b>Danke fuer den Test.</b>)"
echo "<p></body>"
exit
```

Bei komplexeren Anwendungen werden oft die Methode "post" und die Programmiersprache Perl verwendet, und es gibt jeweils verschiedene Hilfsprogramme für das Herausholen der einzelnen Felder aus dem an den Server gesendeten Eingabe-String.

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.

5.11.4 Index-Suche <isindex>

Fast alle →Web-Browser haben eine eingebaute Funktion für "Index-Suche", die dem Benutzer die Eingabe eines "Suchbegriffs" ermöglicht – so ähnlich wie ein →Formular, das nur aus einem einzigen Eingabe-Feld besteht.

Die Ausgabe der Aufforderung zur Eingabe des Suchbefehls, die Verarbeitung der Benutzereingabe und die Ausgabe der Suchergebnisse müssen durch eine auf dem →WWW-Server laufende →CGI-Prozedur erfolgen.

Die Angabe des HTML-Befehls `<isindex>` im →`<head>` aktiviert diese Funktion am →Web-Browser. Der Benutzer kann dann den entsprechenden Browser-Befehl eingeben (bzw. das entsprechende Feld am Bildschirm auswählen) und den Suchbegriff eingeben.

Beispiel:

Eine einfache CGI-Prozedur für die Suche in einem Telefonverzeichnis:

```
#!/bin/sh
echo "Content-Type: text/html"
echo ""
echo "<head><title>Telefonverzeichnis</title>"
if [ $# -eq 0 ]
then : # <isindex> Such-Maske
     echo "<isindex></head><body>"
     echo "<h1>Suche im Telefonverzeichnis</h1>"
     echo "Geben Sie den Befehl f&uuml;r Index-Suche "
     echo "und einen Suchbegriff ein.</body>"
else : # Suche durchfuehren
     echo "</head><body>"
     echo "<h1>Such-Ergebnis f&uuml;r $1</h1>"
     echo "<pre>"
     grep -i "$1" /usr/lib/telefon.liste
     echo "</pre></body>"
fi
exit
```

– – –

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

5.11.5 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.
- In HTML 3 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, wird aber noch nicht von allen Web-Browsern unterstützt.

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 ist in →HTML 2 vorgesehen und wird von den meisten graphikfähigen 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">
<img src="austria.gif" ismap></a>
```

– – – bewirkt eine Darstellung wie

→

und je nachdem, wo in diesem Bild Sie "klicken", bekommen Sie eine entsprechende Antwort.

Die zugehörige **CGI-Prozedur** *hat folgenden Aufbau:*

```
#!/bin/sh
echo "Content-Type: text/html"
echo ""
echo "<head><title>CGI-Test Image-Map</title></head><body>"
echo "<h1>CGI-Test Image-Map</h1><p>"
x=‘echo $1 | sed -e 's/,.*$//'‘
y=‘echo $1 | sed -e 's/^.*,//'‘
echo "x = $x (von links nach rechts)<p>"
echo "y = $y (von oben nach unten)<p>"
```

```
if [ "$y" -lt 16 -o "$y" -gt 31 ]
then echo "Farbe = rot<p>"
else echo "Farbe = weiss<p>"
fi
echo "<p></body>"
exit
```

– – –

Ein etwas sinnvolleres Beispiel finden Sie an der →Universität Wien, wo Sie durch Anklicken der Städte in einer Österreich-Karte Informationen über die jeweilige Stadt erhalten.

Da nicht alle Web-Browser Inline-Bilder und Image-Maps unterstützen und da manche Benutzer mit langsamen oder teuren Netzverbindungen die Darstellung von Bildern in ihren persönlichen Einstellungen ausgeschaltet haben, sollte man bei einer Image-Map immer zusätzlich auch eine gewöhnliche →Liste zur Auswahl von Hypertext-Links in Text-Form anbieten, damit *alle* Benutzer die betreffenden Informationen erreichen können.

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 in →HTML 3 vorgesehen und wird bereits von ein paar 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 Web-Browser solche Image-Maps unterstützen, muß man bei einer solchen 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 die betreffenden Informationen erreichen können.

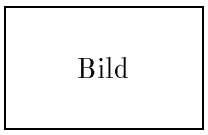
Beispiel:

– – – *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">
<img src="austria.gif" alt="oder hier!" usemap="#map3"></a>
```

– – – *bewirkt die Ausgabe von*

Klicken Sie auf eines der drei Felder der Fahne → 

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;nnen ... 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.

5.11.6 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.

Mailto-URLs werden nicht von allen Web-Browsern unterstützt. "Richtige" Mail-Programme sind für das Versenden von Electronic Mail im allgemeinen besser geeignet, insbesondere für wichtige oder längere Mail-Texte und für das Lesen von Antworten.

Deshalb ist es empfehlenswert, die Mail-Adresse nicht nur "versteckt" im URL sondern auch sichtbar im Text anzugeben.

Beispiel:

– – – Die Eingabe von

```
Wenn Ihnen meine HTML-Einf&uuml;hrung gef&auml;llt,
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

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" erklärt 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 angehören. Derzeit wird an einer neuen HTML-Norm gearbeitet, die gegenüber →HTML 2 erweiterte Möglichkeiten für spezielle Darstellungsformen und Layout-Varianten bietet und die von verschiedenen Software-Firmen herausgebrachten →HTML-Varianten unter einen Hut bringen soll. Auch an anderen Orten wird an vielversprechenden Weiterentwicklungen gearbeitet, so zum Beispiel an →Hyper-G, →PDF, →VRML und →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 →Suchhilfen zu einem wichtigen Hilfsmittel geworden, und schon gibt es auch Hilfsmittel zum Auffinden von Suchhilfen. Ein möglicherweise zukunftsweisendes Forschungs-Projekt 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" (TEX), 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 →"LaTeX" (LATEX) 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.

7. Die roten Fäden der Ariadne

Hinweise zur Online-Verwendung der Hypertext-Version dieses Handbuches

7.1 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.
- →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 wild **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 eigentlich etwas ganz **anderes.**

7.1.1 Ich möchte den kompletten Text lesen.

1. →Vorwort
2. →Wörterbuch
3. →Grundlagen
4. →Textelemente
5. →Hypertext-Links
6. →Bilder und Töne
7. →Layout
8. →Spezialeffekte
9. →Interaktion mit dem Benutzer
10. →Geschichte und Geschichten
11. →Referenzen
12. →Liste der HTML-Befehle

7.1.2 Ich möchte das komplette Handbuch auf Papier ausdrucken.

Für das Ausdrucken der HTML-Einführung auf Papier gibt es zwei Möglichkeiten:

- Entweder Sie lesen die HMTL-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 90 Seiten.
  Eine mit Gnuzip komprimierte Version ist auch über FTP erreichbar:
  →ftp://mail.boku.ac.at/www/htmleinf.ps.gz (ca. 250 kBytes)

Bitte, beachten Sie in beiden Fällen die →Copyright-Bestimmungen.

7.1.3 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.

7.1.4 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 etwas im WWW veröffentlichen?

7.1.5 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. →Aufbau eines HTML-Files
7. →Filename und Directory
8. →Textelemente
9. →Hypertext-Links
10. →Bilder und Töne
11. →Layout und Spezialeffekte
12. →Geschichte und Geschichten

7.1.6 Ich möchte rasch eine kurze Information im WWW veröffentlichen.

1. →Vorwort
2. →Aufbau eines HTML-Files
3. →Absätze und Zeilenumbruch
4. →Buchstaben und Sonderzeichen
5. →Wie kann ich mein HTML-File im WWW verfügbar machen?

7.1.7 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 mein HTML-File im WWW veröffentlichen?

7.1.8 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

7.1.9 Ich möchte, daß meine Informationen bunt und wild aussehen

1. →Vorwort
2. →Frisch geplant ist halb gewonnen!
3. →Inhalt und Form
4. →Richtige HTML
5. →Bilder und Töne
6. →Farben
7. →Trennlinien
8. →hervorgehobene Wörter
9. →hervorgehobene Absätze
10. →Überschriften
11. →Listen und Aufzählungen
12. →Layout und Spezialeffekte

7.1.10 Ich möchte, daß meine Informationen von vielen Leuten gelesen werden.

1. →Vorwort
2. →Frisch geplant ist halb gewonnen!
3. →Wie kann ich mein HTML-File 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. →Schönes Layout
8. →Günstige Aufteilung
9. →Richtige HTML
10. →Inhalt und Form
11. →Zugriffe zählen

7.1.11 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

7.1.12 Ich möchte etwas anderes.

- →Wer suchet, der findet...
- →...ein Auto im Dschungel?

7.2 Referenzen

- →Internet-Handbuch von Hubert Partl (BOKU Wien)
- →weitere Informationen über das Internet
- →Informationen über WWW und Clients
- →Informationen über WWW-Server und HTML
- →"HTML-Dateien selbst erstellen" von Stefan Münz.
- →W3-Consortium:
  - →allgemeine Informationen über WWW und HTML, WWW-Server, Clients, Editoren und Hilfsprogramme, Spezifikationen von HTML 2, HTML 3 und Style-Sheets
- →Übersicht über die neuesten HTML-Spezifikationen (H.A.Driscoll) und →die neuesten HTML-Internet-Drafts (IETF)
- HTML-Prüfprogramme (Validatoren):
  - →HALsoft Validator (Webtechs, USA, oder →Eunet Österreich)
  - →Weblint
  - →Kinder Gentler Validator
  - →HTML-Check
  - →MOMspider
  - →Liste von weiteren HTML-Validatoren
- →WWW-Server in aller Welt
- →Suchhilfen in aller Welt

7.3 Liste der HTML-Befehle

Alphabetische Liste der in der vorliegenden HTML-Einführung beschriebenen HTML-Befehle ("Tags"):

<a> <area>

<b> <big> <blink> <blockquote> <body> <bq>

 <code>

<dd> <dl> <dt>

<font> <form> <frame> <frameset>

<head> <hr> <html> <h1> <h2> <h3>

<i> <img> <input> <isindex>

 <link>

<map>

<noframes> <note>

<object>

<p> <pre>

<small> <style> <sub> <sup>

<tab> <table> <td> <th> <title> <tr> <tt>

<u>

<!doctype> <!-- -->

Happy Web-Weaving!