1 Einführung

1.1 Was ist das Internet?

Das Internet ist ein riesiges Computernetzwerk, dessen Computer auf der ganzen Welt verteilt sind. Innerhalb dieses Netzwerkes gibt es verschiedene Dienste für die unterschiedlichsten Aufgaben.

In einem Netzwerk ist der Datenaustausch zwischen den Rechnern durch ein Protokoll geregelt. Im Internet heißt dieses Protokoll *TCP/IP*.

1.2 Das World Wide Web

Das *World Wide Web*, kurz *WWW* oder *Web*, ist sicherlich der am schnellsten wachsende Internetdienst. Viele, die vom Internet reden, meinen das WWW. Der derzeitige Internetboom ist wohl auch überwiegend auf die Entwicklung und Verbreitung des WWW zurückzuführen.

Da die Daten nicht nur in Textform vorliegen, sondern auch mit Bildern und sogar Filmen und Sound unterlegt werden können, sind die Daten leicht verständlich aufbereitet.

Grundbegriffe des WWW

Die URL

Beim Reden mit bereits versierten Internetnutzern werden Sie häufiger den Begriff *URL (Uniform Resource Locator)* hören. Zum einen meinen sie damit ganz allgemein ein bestimmtes Angebot, zum anderen ist das die Adresse eines *Servers* bzw. der genaue Punkt, wo Sie bestimmte Daten finden.

Hinter einem Hyperlink auf einer WWW-Seite verbirgt sich immer ein Verweis auf eine URL. Diese kann auf dem gleichen oder einem anderen Server liegen.

Sie haben auch die Möglichkeit, über Ihren Web-Browser direkt eine bestimmte URL anzuwählen. Von dort aus können Sie dann mit Hilfe der Hyperlinks, die meist themenbezogen sind, browsen oder eine neue URL direkt anwählen.

URLs sind nicht nur die Adressen im WWW, sondern auch bei den anderen Internetdiensten. Eine typische URL im WWW ist folgendermaßen aufgebaut: *http://Server/Verzeichnis/Datei*. Wenn Sie die Datei nicht angeben, können drei Dinge passieren:

1. Ein Dokument ist zum automatischen Laden definiert, und Sie erhalten die Startseite des Angebots.
2. Sie erhalten eine *Verzeichnisstruktur* der URL und können daraus die gewünschten Dateien durch Doppelklick mit der Maus aufrufen.
3. Sie erhalten eine Fehlermeldung, daß kein Zugriff erlaubt ist.

Das einfachste Beispiel einer URL wäre etwa http://www.ibm.com. Obwohl weder ein zu ladender Dateiname, noch das Verzeichnis, das die Datei enthält angegeben ist, wird in diesem Fall die Startseite automatisch geladen.

Das HTTP-Protokoll

HTTP heißt *Hypertext Transfer Protocol* und ist das Übertragungsprotokoll des WWW. Bei Angabe der URL muß das Protokoll immer in Kleinbuchstaben geschrieben werden; sollten Sie hier einen Fehler machen, korrigiert der Netscape Communicator dies allerdings selbsttätig.

HTTP ist ein schnelles Protokoll, da es wenige Daten für eine Anfrage benötigt. Es wird von Ihnen nur die URL an den gewünschten WWW-Server gesendet, und dieser schickt die gewünschten Daten dann zu Ihnen zurück. So werden nur die wirklich angeforderten Informationen gesendet.

Als erstes werden vom Server Daten über die Art des angeforderten Dokumentes an Sie gesendet, und anschließend folgt das Dokument selber. Ist die angegebene URL fehlerhaft, erhalten Sie eine entsprechende HTTP-Fehlermeldung.

HTML

HTML heißt *Hypertext Markup Language* und ist die Programmiersprache, in der WWW-Seiten erstellt werden. HTML ist jedoch mehr eine Seitenbeschreibungs- bzw. Layoutsprache als eine Programmiersprache und recht einfach zu erlernen.

Suchsysteme

Ein wichtiges Hilfsmittel zum Auffinden bestimmter Informationen im WWW sind die sogenannten Suchsysteme. In den Suchsystemen können Sie einen Begriff eingeben, und in der Datenbank des Suchsystems wird nach dem entsprechenden Begriff gesucht. Als Ergebnis wird Ihnen eine Liste mit URLs geliefert, von der Sie per Klick mit der Maus direkt die gewünschte URL anwählen können.

Die Suchsysteme arbeiten auf zwei verschiedenen Arten. Zum einen gibt es die *Webspider*: Das sind Programme, die nichts anderes tun, als immer nur Links durch das WWW zu folgen, die also automatisch surfen und die dabei gesammelten Informationen in ihrer Datenbank speichern.

Die andere Art ist das manuelle Zusammentragen von URL-Informationen und deren Eingabe in eine Datenbank. Das hat vor allem für die Betreiber einer *WWW-Site* den Vorteil, schnell im Suchsystem vertreten zu sein, da er sich selber im Suchsystem bekannt machen kann, und nicht darauf warten zu müssen, bis ein Spider mehr oder weniger durch Zufall auf diese Seite stößt. Die meisten Suchsysteme arbeiten mit einer Kombination aus beiden Methoden.

Adressen von Suchsystemen

Name	Sprache	Adresse (URL)
Alta Vista	Englisch	www.altavista.telia.com
Infoseek	Englisch	www.guide.infoseek.com
Aladin	Deutsch	www.aladin.com
Crawler	Deutsch	crawler.de
Dino	Deutsch	www.dino-online.de
Lycos	Deutsch	www.lycos.de
Yahoo!	Deutsch	search.yahoo.de
Web	Deutsch	vroom.web.de
Flipper	Deutsch	flp.cs.ti-berlin.de/flipper

Tab. 1.1: Die Adressen einiger Suchdienste

2 Grundlagen

2.1 Was ist HTML?

Die *HyperText Markup Language*, was *HTML* ausgeschrieben heißt, ist ganz kurz gesagt die Programmiersprache des World Wide Web. Ganz präzise ist das allerdings nicht. Eigentlich ist *HTML* eine *Auszeichnungssprache*, die definiert, wie das Erscheinungsbild eines Textdokumentes aussieht. *HTML* wurde aus *SGML* (Standard Generalized Markup Language) entwickelt.

Der Text, den ein *HTML-Dokument* enthält, wird *Quelltext* genannt. Dies ist das gleiche, was bei anderen Programmiersprachen *Listing* genannt wird.

Im Gegensatz zu Textdokumenten einer bestimmten Textverarbeitung wird der Text in allgemeingültigem *ASCII-Format* gespeichert, das auf nahezu allen bekannten Betriebssystemen standardisiert ist. Dadurch haben wir einen *Quelltext*, der neben dem Text noch die Formatierungshinweise enthält und der zudem noch auf allen Computern, gleich mit welchem Betriebssystem sie arbeiten, verwendet werden kann. Das einzige, was dazu benötigt wird, ist ein Programm, das dieses Dokument interpretiert: Der *Web-Browser*.

Egal ob für PC-basierende Windows oder OS/2-Systeme, Atari, Amiga, Apple Macintosh oder Unix-Rechner auf unterschiedlichster Hardware basierend für alle gibt es *Web-Browser*. Und da in allen Betriebssystemen der *ASCII-Zeichensatz* bekannt ist, lassen sich auch auf all diesen Plattformen *HTML-Dokumente* erstellen.

Ein *HTML-Dokument* enthält also zwei Arten von Informationen:

- Den Inhalt, der aus Text oder Verweisen auf zu ladende Dateien, z. B. Grafiken, besteht.
- Die Formatierungshinweise und sonstige Markierungen, wie Verknüpfungen zu anderen Dokumenten, die Tags genannt werden und sozusagen die Programmierbefehle sind.

Der Inhaltstext wird vom Web-Browser unter Beachtung der *Tags*, das sind die *HTML-Befehle*, dargestellt, d. h. der Browser interpretiert die *Tags*, damit die Darstellung des Inhaltes in gewünschter Form erfolgt.

HTML 4.0

Gegenüber den früheren Versionen von *HTML* hat sich einiges verändert. Es gibt *Tags*, die völlig neu eingeführt wurden, andere fallen raus. Manche *Tags* sollen Sie nicht mehr verwenden, dürfen dies aber noch.

Daraus folgt eine Klassifizierung in verbotene und unerwünschte *Tags*. Bedenken Sie immer, daß die Übergänge fließend sind, nicht alle neuen *Tags* verstehen die Browser sofort, und die verbotenen *Tags* verstehen die Browser zur Zeit noch.

Allerdings sollten Sie Ihre Programmierweise möglichst bald umstellen, denn Sie wissen ja, wie schnell die Evolution im Computerbereich vorangeht.

htm und html

HTML-Dokumente haben die Dateiendung htm. Das ist richtig, die ursprüngliche Dateiendung lautet jedoch html. Auf PCs mit DOS als Betriebssystem war es nicht möglich, Dateiendungen, die sogenannten Extensions, mit vier Buchstaben zu verwenden, diese mußten aus drei Zeichen bestehen.

Deshalb wurde aus html einfach htm. Im WWW finden Sie html dennoch sehr häufig, da dort Unix-basierende Computer vorherrschen, die diese Einschränkung nicht haben. Kurz und bündig, es handelt sich in beiden Fällen um HTML-Dateien.

2.2 *HTML* – Die Geschichte

Nachdem das Internet zunächst nur Dienste anbot, die für Experten gedacht waren und damit auch nicht sonderlich komfortabel waren, wurden gegen Ende der 80er Jahre, also kurz nach der Steinzeit des Computerzeitalters, zwei einfacher zu bedienende Dienste entwickelt: *Gopher* und das *World Wide Web*.

Mit diesen zwei Diensten wurden im Internet erstmals Dienste eingeführt, die nicht befehlszeilenorientiert, sondern mit Hilfe einer Benutzeroberfläche arbeiten. Ziel war es, daß die Wissenschaftler nicht mehr zuerst eine komplizierte Nutzung des Computers lernen mußten, um auf Datenfang zu gehen, sondern daß sie sich voll auf das für sie Wichtige konzentrieren können.

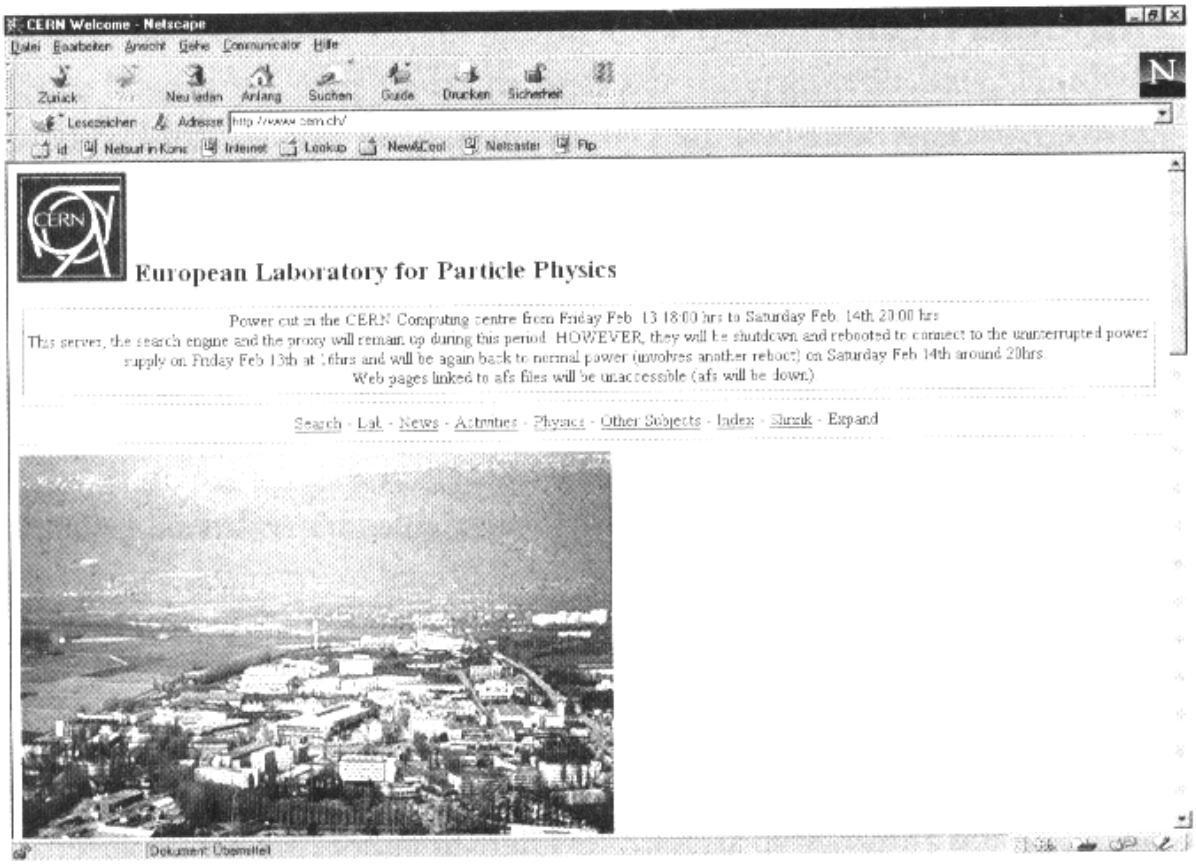

Abb. 2.1: CERN – hier fing alles an

Schnell zeigte sich, daß gegenüber Gopher das World Wide Web, das am *CERN*, dem *europäischen Kernforschungszentrum* in Genf, entwickelt wurde, das Rennen machen werde.

Die Programmiersprache, mit deren Hilfe WWW-Anwendungen erstellt werden, ist *HTML*, die *HyperText Markup Language*.

Einen sprunghaften, fast schon explosionsartigen Anstieg der WWW-Nutzer gab es 1993, und seitdem steigt die Anzahl der Nutzer permanent; die *Bandbreiten* für die Übertragungswege können gar nicht in dem Maße ausgebaut werden, wie die Userzahl wächst.

Auslöser dieses Booms war vermutlich das Erscheinen des ersten Web-Browsers mit einer vollgrafischen Benutzeroberfläche: *NSCA Mosaic*. Das Erscheinen des zu aller Freude auch noch kostenlosen Browsers machte

das Navigieren durch das Internet zum wahren Kinderspiel. Der Programmierer war ein Student Namens Marc Andreessen, später der Firmengründer von Netscape, der Firma, die den erfolgreichsten Web-Browser, den *Netscape Navigator,* herstellt und vertreibt.

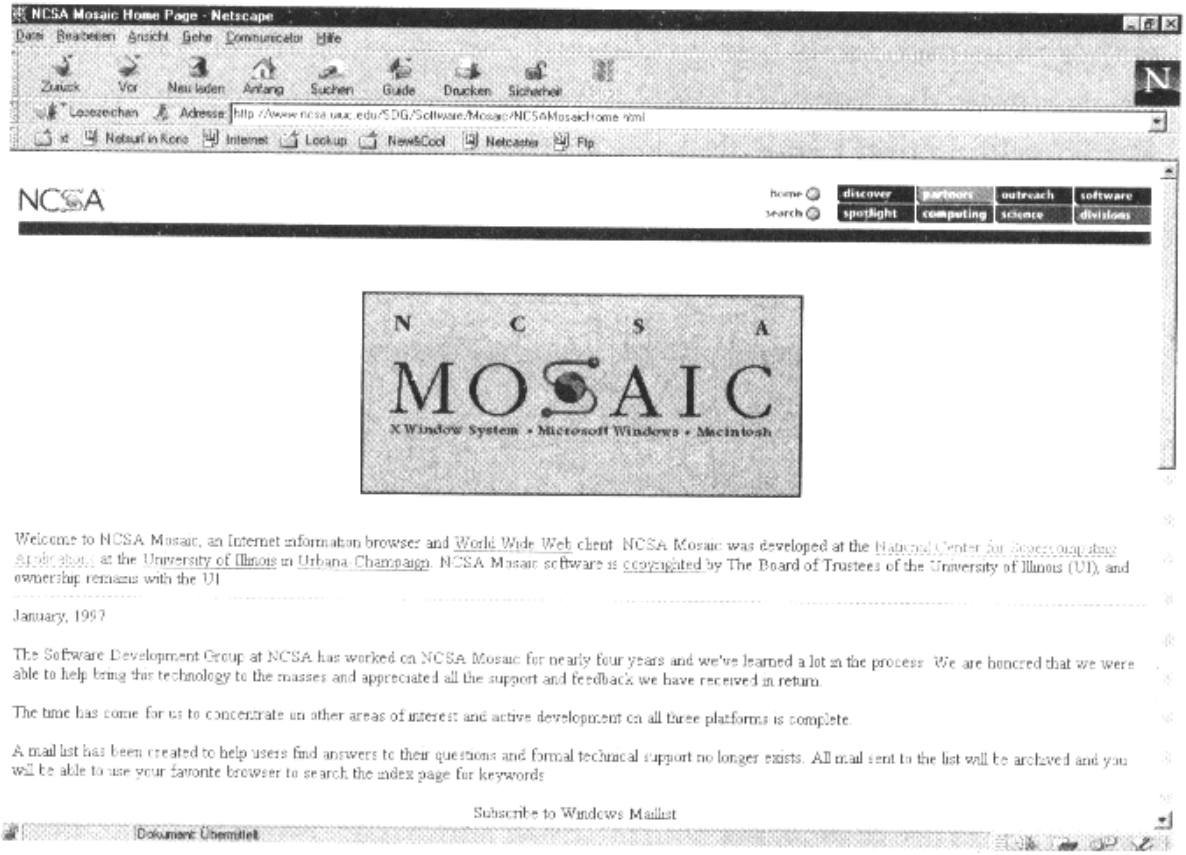

Abb. 2.2: ...und von hier kam der Erfolg

2.3 Wer entwickelt *HTML*?

Inzwischen hat sich nicht nur das WWW extrem ausgebreitet, auch die dazugehörige Sprache, *HTML*, wurde in ihrem Leistungsumfang stark erweitert und weiterentwickelt.

Dies und auch die starke Kommerzionalisierung des Internet haben dazu geführt, daß längst nicht mehr das *CERN* die Koordinierungszentrale des WWW ist. Auch die Koordinierung der Standardisierung und Weiterentwicklung von *HTML* wird nicht mehr vom *CERN* gesteuert.

Diese Aufgaben werden jetzt vom *W3-Consortium* (http://www.w3c.org) wahrgenommen, daß nur zu diesem Zweck entstanden ist. Im *W3-Consortium*, das kurz mit *W3C* bezeichnet wird, sind sowohl Vertreter aus Forschung und Lehre als auch Vertreter der namhaften Software- und Hardwarehersteller vertreten. So übt der kommerzielle Bereich inzwischen massiven Einfluß auf das ehemals unkommerzielle Internet aus.

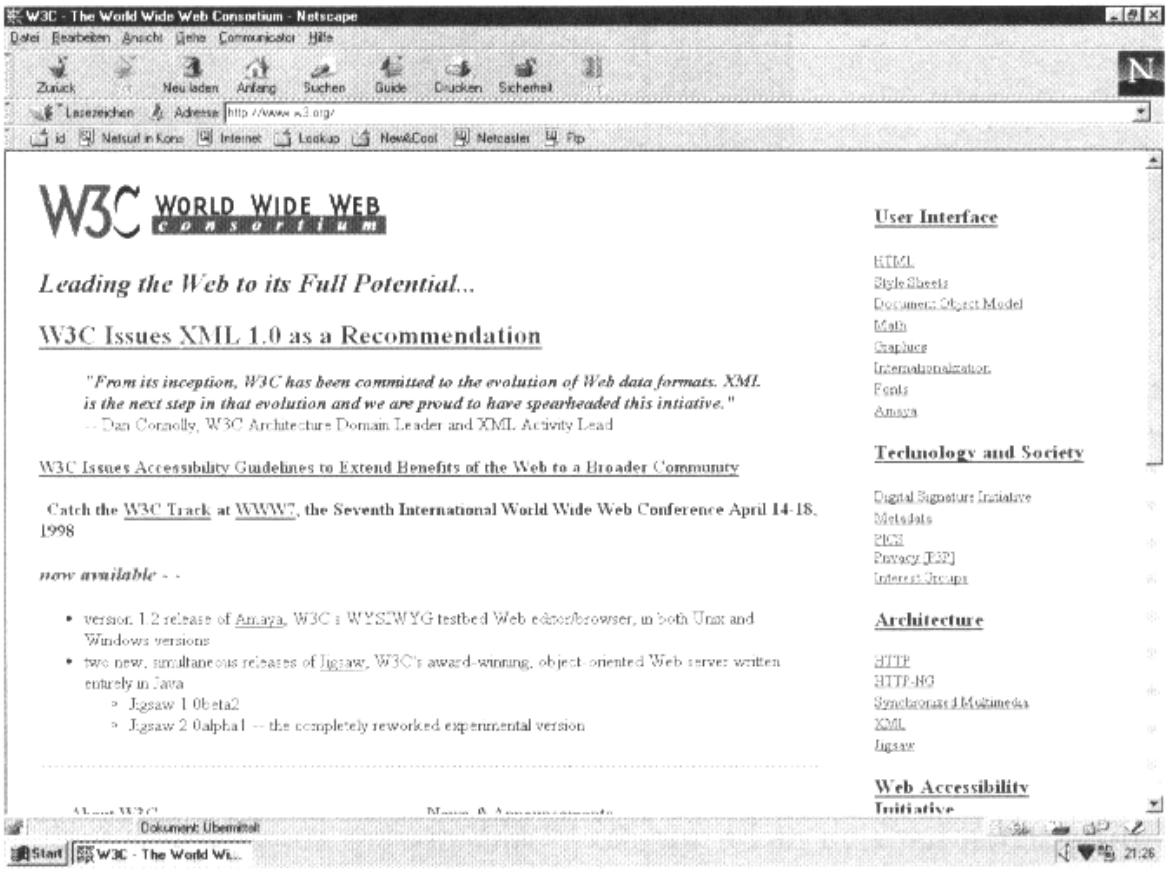

Abb. 2.3: Die Web-Site des W3C

Hier werden die Normen festgelegt und die Entwicklung neuer Standards koordiniert. So befaßt sich das *W3C* nicht nur mit *HTML*, auch *VRML*, *Java* und *XML* – um nur einige Beispiele zu nennen – werden hier in der Entwicklung koordiniert. Hier wird daran gearbeitet, die verschiedenen Ansätze für neue *HTML-Tags* unter einen Hut zu bringen und die Firmen dazu zu bewegen, sich an die vereinbarten Standards zu halten. Der aktuelle Sproß dieser Bemühungen ist *HTML 4.0*.

Die Zukunft?

Die Zukunft heißt wahrscheinlich nicht *HTML*, sondern *XML*. *XML* ist wie *HTML* eine Abkürzung für *eXtensible Markup Language*.

Vor kurzem wurde der Entwurf für *XML* 1.0 auf den Webseiten des *W3-Consortiums* vorgestellt und von seiten professioneller *HTML-Entwickler* erfreut angenommen.

Zunächst beteiligten sich viele namhafte Hersteller an der Entwicklung von *XML*: Novell, Microsoft, IBM und Hewlett-Packard waren neben Sun, dem »Erfinder« von *XML*, die ersten großen Unterstützer, die *XML* als Standardsprache für professionelle *Webapplikationen* etablieren wollten. Lediglich Netscape, allerdings Marktführer bei WWW-Browsern, blockte völlig ab.

Im Frühjahr 1997, während der heißen Entwicklungsphase des Netscape Comunicator 4, entschied sich Netscape völlig unerwartet, nun doch auf den Zug aufzuspringen. Das war wohl der Punkt, an dem klar wurde, daß die Zukunft von *HTML XML* heißt.

XML ist ein drastisches Rückbesinnen auf den Ursprung: auf *SGML*. Die Möglichkeiten von *XML* sind nahezu unbegrenzt, wenn man *XML* mit *HTML* vergleicht.

Gerade dies war auch der Grund, *XML* ins Leben zu rufen, denn nicht die langsamen Übertragungsmöglichkeiten sind nach Einschätzung vieler Webprofis das Hauptproblem des WWW, sondern die beschränkten Möglichkeiten von *HTML*.

Bereits jetzt werden schon erste Stimmen laut, nicht länger die Zeit mit der Weiterentwicklung von *HTML* zu vertun, sondern statt dessen *XML* voranzutreiben. Sicher wird *XML* bald auch ein wichtiger Bestandteil der Webprogrammierung sein, die Browser werden aber problemlos auch weiterhin *HTML-Quelltext* interpretieren.

Für viele Webseiten ist *XML* auch gar nicht nötig, und so werden *HTML-Dokumente* sicher noch lange im Web zu finden sein. Denn einen großen Vorteil hat *HTML*: Es ist ohne langjährige Programmiererfahrung recht leicht in den Griff zu bekommen.

2.4 Was benötige ich zum Programmieren in *HTML*?

Sie benötigen nicht viel: Ein einfacher Texteditor reicht schon aus. Wenn Ihre gewohnte Textverarbeitung die Möglichkeit hat, den Text im *ASCII-Format* zu speichern, dann können Sie diese auch verwenden.

Besonders komfortabel ist die Möglichkeit, *HTML-Dokumente* mit Hilfe eines sogenannten *HTML-Editors* zu erstellen. *HTML-Editoren* sind Programme, die Ihnen Hilfe beim Erstellen von *HTML-Dokumenten* geben. Bei einigen brauchen Sie nicht viel mehr zu machen, als Text und Grafiken im Seitenlayout zu plazieren, andere helfen Ihnen nur dadurch, daß Sie das fertige Ergebnis gleich vor sich haben oder zumindest ein sehr ähnliches Bild. *HTML* zu beherrschen ist jedoch auch hilfreich, wenn Sie einen sogenannten *WYSIWYG-Editor* verwenden, denn wenn es zu Darstellungsproblemen kommt, kann Ihnen nur noch Ihr eigenes Wissen weiterhelfen.

Eine große Hilfe kann also auch ein *WYSIWYG-Editor* sein; meist enthält dieser auch einen *ASCII-Editor*, mit dem Sie dann den *Quelltext* direkt editieren können.

Ganz gleich, ob Sie mit einem Texteditor oder einem speziellen *HTML-Editor* arbeiten, speichern Sie vor dem Betrachten immer erst die aktuelle Version des Dokumentes ab, sonst sehen Sie die zuletzt abgespeicherte Version und wundern sich über die fehlende Wirkung der gemachten Änderungen.

Natürlich brauchen Sie auch noch ein Programm zum Betrachten des Ergebnisses: einen *Web-Browser*. Hier gibt es eine ganze Reihe am Markt. Am meisten verbreitet sind der Internet Explorer von Microsoft (http://www.microsoft.com) und der Navigator oder das Komplettpaket Communicator von Netscape (http://www.netscape.com). Der Netscape Communicator bietet den Vorteil, daß er mit dem Composer auch gleich noch einen guten *HTML-Editor* integriert hat.

Anstatt eines *Web-Browsers* können Sie auch eine Textverarbeitung nehmen, die *HTML-Dokumente* interpretieren und darstellen kann, wie z. B. StarWriter aus dem Paket Star Office (http://www.stardivision.de) oder Microsoft Word 97.

Wenn Sie in Ihre *HTML-Dokumente* Grafiken einbinden möchten, sollten Sie sich auch noch ein Grafikprogramm zulegen. Selbst wenn Sie nur fertige Grafiken oder Bilder verwenden möchten, können Sie mit einem Grafikprogramm die Farben oder die Auflösung wenn möglich reduzieren, um so Ihr Dokument schneller zu machen.

2.5 Wie kommt die Seite ins WWW?

Damit Ihre *HTML-Dokumente* dann auch nicht nur auf Ihrem Computer existieren, sondern auch andere Ihre Meisterwerke bewundern können, müssen diese auf einen sogenannten *Web-Server* abgespeichert werden.

Web-Provider

Dazu brauchen Sie einen eigenen Internet-Zugang und einen Anbieter von *Web-Space*, d. h. der Ihnen Speicherplatz auf einem Web-Server zur Verfügung stellt.

Oft haben Sie mit dem Zugang zum Internet auch schon die Möglichkeit, eigene *HTML-Dokumente* auf einem Web-Server abzulegen. Sehr viele *Provider* bieten an, in der Regel jedoch nur für den nichtkommerziellen Bereich, private Homepages zu veröffentlichen. Dies ist die günstigste Möglichkeit, da sie nicht mit weiteren Kosten verbunden ist.

Haben Sie diese Möglichkeit nicht, dann müssen Sie sich einen entsprechenden Provider suchen. Holen Sie sich dazu verschiedene Angebote ein, das Preis- Leistungsverhältnis schwankt extrem. Vorsicht, der Anbieter mit dem niedrigsten Preis ist nicht immer der günstigste, da es sein kann, daß »low-cost«-Anbieter nur eine sehr schlechte Anbindung haben, so daß interessierte Benutzer durch Wartezeiten oder langsame Übertragungen abgeschreckt werden.

Von diesem Provider erfahren Sie dann, wie Sie Ihre Dokumente auf den Server bekommen.

3 Der Seitenaufbau

Dieses Kapitel wird Ihnen einen Einblick in den Grundaufbau einer *HTML-Seite* gewähren. Sie lernen das nötige Grundwissen, um erfolgreich und zielorientiert eigene *HTML-Dokumente* zu erstellen.

3.1 Die Syntax

Jede Sprache hat ihre Grammatikregeln und ihre Rechtschreibung. Das trifft auch auf Programmiersprachen und damit auch auf *Seitenbeschreibungssprachen* zu.

Bei einer einfachen *Seitenbeschreibungssprache* wie *HTML* sind diese Regeln sehr einfach und schnell zu beherrschen. Allerdings müssen Sie genau auf die richtige Verwendung achten, damit die Funktion und die korrekte Darstellung des Seiteninhalts gewährleistet sind.

Die Befehle

Die wichtigste Regel ist die, daß die Befehle – die bei HTML *Tags* oder auch *Elemente* genannt werden – immer in spitzen Klammern stehen, z. B.:

```
<U> Unterstrichener Text </U>
```

Zwischen diesen beiden spitzen Klammern, die auf der Tastatur als die Zeichen »größer als« und »kleiner als« zu finden sind, steht dann der gewünschte Befehl, der meist die Abkürzung eines englischen Begriffs ist.

Wie bei unserem ersten Beispiel mit dem unterstrichenen Text werden die meisten Tags auch wieder geschlossen, da sie eine Formatierung einlei-

ten, die nicht bis zum Ende der Seite gültig ist, sondern durch andere Formatierungen abgelöst wird.

Wird das *Endtag* nicht gesetzt, gilt die Formatierung für den ganzen Rest der Seite. Eine saubere Programmierung bedingt jedoch auch in diesem Fall das Setzen des *Endtags*.

Das *Endtag* wird durch einen Schrägstrich, dem sogenannten *Slash*, zwischen der ersten spitzen Klammer und dem eigentlichen Befehl gesetzt (z. B.: </U> beendet unterstrichenen Text).

Die Schreibweise

Auf Groß- und Kleinschreibung innerhalb eines Tags brauchen Sie nicht zu achten. Wegen der besseren Übersicht empfiehlt es sich jedoch, sich an eine einheitliche Schreibweise zu gewöhnen. Üblich ist es, *Tags* in Großbuchstaben zu schreiben.

Im Gegensatz zur Groß- und Kleinschreibung sollten Sie mit der Verwendung von Leerzeichen innerhalb eines Tags sehr vorsichtig umgehen. Ein Leerzeichen direkt vor oder nach einer spitzen Klammer eines *Tags* führt bei manchen Browsern zur Ignorierung dieses *Tags*.

Das Problem daran ist, daß bei einer Überprüfung Ihr eigener Browser die Seite unter Umständen korrekt darstellt, ein fremder Betrachter mit einem anderen Browser dann eventuell Layoutmüll zu Gesicht bekommt.

Bevor wir nun zum Inhalt eines Dokuments kommen, müssen wir erst einmal das Grundgerüst der Seite erstellen. Die nötigen Grundlagen erfahren Sie im folgenden Abschnitt.

ten, die nicht bis zum Ende der Seite gültig ist, sondern durch andere Formatierungen abgelöst wird.

Wird das *Endtag* nicht gesetzt, gilt die Formatierung für den ganzen Rest der Seite. Eine saubere Programmierung bedingt jedoch auch in diesem Fall das Setzen des *Endtags*.

Das *Endtag* wird durch einen Schrägstrich, dem sogenannten *Slash*, zwischen der ersten spitzen Klammer und dem eigentlichen Befehl gesetzt (z. B.: </U> beendet unterstrichenen Text).

Die Schreibweise

Auf Groß- und Kleinschreibung innerhalb eines Tags brauchen Sie nicht zu achten. Wegen der besseren Übersicht empfiehlt es sich jedoch, sich an eine einheitliche Schreibweise zu gewöhnen. Üblich ist es, *Tags* in Großbuchstaben zu schreiben.

Im Gegensatz zur Groß- und Kleinschreibung sollten Sie mit der Verwendung von Leerzeichen innerhalb eines Tags sehr vorsichtig umgehen. Ein Leerzeichen direkt vor oder nach einer spitzen Klammer eines *Tags* führt bei manchen Browsern zur Ignorierung dieses *Tags*.

Das Problem daran ist, daß bei einer Überprüfung Ihr eigener Browser die Seite unter Umständen korrekt darstellt, ein fremder Betrachter mit einem anderen Browser dann eventuell Layoutmüll zu Gesicht bekommt.

Bevor wir nun zum Inhalt eines Dokuments kommen, müssen wir erst einmal das Grundgerüst der Seite erstellen. Die nötigen Grundlagen erfahren Sie im folgenden Abschnitt.

3.2 Die Seite

Das Programm, das unser *HTML-Dokument* darstellt (in der Regel ein Web-Browser), muß erkennen, daß es sich bei der Datei, die es lädt, um ein *HTML-Dokument* handelt. Dazu gibt es grundlegende Befehle zur Erstellung eines *HTML-Dokuments*.

Eine einfache Seite

Das Grundgerüst einer leeren *HTML-Seite* sieht immer gleich aus:

```
<HTML>

<BODY>

</BODY>

</HTML>
```

Sie sehen, so einfach ist das. Dies ist bereits das Grundgerüst einer funktionierenden *HTML-Seite*, die allerdings noch leer, ohne Inhalt ist.

Eine *HTML-4.0*-konforme Seite

In der Praxis wird das Gerüst allerdings etwas erweitert sein. Deshalb nachfolgend die sinnvollere Variante, die ich Ihnen gleich im Anschluß erklären werde.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">

<HTML>

<HEAD>

<TITLE>
```

```
</TITLE>

</HEAD>

<BODY>

</BODY>

</HTML>
```

Und das bedeuten die einzelnen Zeilen:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">

Hier wird festgelegt, nach welcher Spezifikation das HTML-Dokument erstellt wurde, das heißt, welcher Standard dieser Seite zugrunde liegt.

<HTML>

Hier wird das eigentliche Dokument geöffnet. Dies ist das erste *Tag* in einem HTML-Dokument, es wird erst am Ende des Dokuments wieder aufgehoben.

<HEAD>

Nun wird der Kopf geöffnet.

<TITLE>

Innerhalb des Kopfes wird der Titel der Seite angegeben, der dann in der Titelleiste des Browsers angezeigt wird.

</TITLE>

Dies beendet den Titel.

</HEAD>

Nun ist auch der Seitenkopf beendet.

<BODY>

Der Rumpf der Seite wird geöffnet. Hier befindet sich der eigentliche Seiteninhalt.

</BODY>

Nun wird der Rumpf mit dem Inhalt wieder geschlossen.

</HTML>

Hier wird nun das Seitenende festgesetzt.

Document Type Declaration

Für gewöhnlich beginnt jede Seite mit dem *Tag* <!DOCTYPE>. Dies ist keine Notwendigkeit, aber die Spezifikationen von *HTML 4.0* sehen dies so vor.

Die *Document Type Declaration*, kurz *DTD*, ist ein *SGML*-Befehl und definiert, nach welchem Standard das folgende Dokument erstellt wurde.

Sauberes Programmieren

In *HTML* sollten Sie gemäß den Spezifikationen folgende Zeile zur *DTD* verwenden:

- Wenn Sie sich absolut an die Spezifikationen gehalten haben und in Ihrem Dokument keine *Tags* stehen, die zwar zugelassen sind, die Sie aber möglichst nicht mehr benutzen sollten, wie z. B. die *Tags* <FONT> und <STRIKE>, dann verwenden Sie folgende Zeile:

  ```
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">
  ```

- Haben Sie sich an die Spezifikation gehalten, aber die unerwünschten *Tags* dennoch verwendet, dann benutzen Sie einfach nachfolgende Zeile:

  ```
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN//">
  ```

- Wenn Sie Ihre Seite nicht nach den Richtlinien von *HTML 4.0* erstellt haben, geben Sie die dazugehörige *DTD* an. Bis zur *HTML-Version 3.0* braucht man keine DTD anzugeben. In diesen Fällen wird immer *HTML 2.0* zugrunde gelegt. Aufgrund der raschen Entwicklung von Browsern und auch von *HTML* sollten Sie jedoch immer aktuell sein und *HTML 4.0* konform programmieren.

- Wenn Sie *HTML-Seiten* mit *Frames* erstellen, sollte das Dokument, das den *Frameset* enthält, folgende *DTD* erhalten:

  ```
  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN//">
  ```

  Damit wird angegeben, daß das Frameset den Regeln von *HTML 4.0* entspricht.

- Die neueste Browsergeneration unterstützt bereits die Möglichkeit, daß die Spezifikationen, nach denen das Dokument erstellt wurde, direkt von einer URL abgefragt werden. Dazu kann an die *DTD* eine URL angefügt werden. Haben Sie sich ganz streng an die Spezifikationen für *HTML 4.0* gehalten, würde die Zeile folgendermaßen lauten:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//""http://
www.W3.org/DTD/HTML4-strict.dtd">
```

Head

Direkt unterhalb der Document Type Declaration folgt der Kopf des Dokuments. Hier finden sich Informationen zum Dokument, die nicht direkt das Erscheinungsbild des Dokuments beeinflussen.

Der Kopf des Dokuments wird mit <HEAD> geöffnet und mit </HEAD> geschlossen. Alle Informationen zwischen diesen *Tags* gehören zum Kopf.

Der Dokumententitel

Das sicherlich am häufigsten verwendete *Tag* im Kopfteil eines Dokuments ist das *Tag* <TITLE> </TITLE> zwischen dessen beiden Teilen der Dokumententitel angegeben werden kann, der dann bei den meisten Browsern oben in der Titelzeile angezeigt wird (siehe Abbildung 3.1).

```
<TITLE>Dies ist der Seitentitel</TITLE>
```

Diese Zeile zeigt die Anwendung des <TITLE>-*Tags*, wobei *Dies ist der Seitentitel* durch den gewünschten Titel ersetzt wird.

Abb. 3.1: Titelzeile im Netscape Navigator

Weitere Tags im Kopfteil

Im Kopfteil eines *HTML-Dokuments* werden noch weitere *Tags* plaziert. Es handelt sich dabei um die folgenden:

<BASE>

<BGSOUND>

<ISINDEX>

<META>

<STYLE>

Davon ist für uns das Tag <STYLE> wichtig, auf das ich in Kapitel 10 näher eingehe. Dort geht es dann nur um Stilvorlagen, die mit diesem Tag realisiert werden.

Body

Nun kommen wir zum eigentlichen Dokument. Der Inhalt samt seiner Formatierungen wird zwischen den Tags <BODY> und </BODY> definiert.

Der größte Teil dieses Buches befaßt sich mit dem Teil, der zwischen diesen beiden Tags zu finden ist.

3.3 Kommentare

Überall, wo programmiert wird, ist es üblich, Kommentare im *Quelltext* anzubringen. Das erleichtert einem selber bei späteren Änderungen die Arbeit. Wenn jemand anderes die Überarbeitung vornehmen soll, ist diese Hilfe natürlich noch viel wichtiger.

Kommentare werden in *HTML* durch ein spezielles *Tag* kenntlich gemacht:

```
<!–Hier steht der Kommentar–>
```

wobei *Hier steht der Kommentar* durch eben den gewünschten Kommentar ausgetauscht wird.

Texte innerhalb dieses *Tags* werden von jedem Browser als Kommentar erkannt und völlig ignoriert. Damit lassen sich Kommentare zum besseren Verständnis der Programmierung oder als besonderer Hinweis in den Quelltext einfügen.

```
<HTML> <!– Dieses Tag öffnet ein HTML-Dokument –>

<BODY> <!– Dieses Tag öffnet den Inhalt –>

</BODY> <!– Der Inhalt wird abgeschlossen –>

</HTML> <!– Das ist das Endtag eines HTML-Dokuments –>
```

Dies war ein Beispiel für einen kommentierten Quelltext.

3.4 Entities

Natürlich können Sie den inhaltlichen Text Ihres *HTML-Dokuments* sofort in der Art und Weise schreiben, mit den deutschen Sonderzeichen, wie Sie es gewöhnt sind. Allerdings werden die deutschen Sonderzeichen bei den wenigsten Betrachtern korrekt wiedergegeben. Ob die Wiedergabe richtig ist und wie sie aussieht, hängt vom verwendeten Browser und von den Landeseinstellungen des Betriebssystems ab.

Was sind *Entities*?

Die Lösung dieses Problems sind die *Entities*. Ein *Entity* gibt es nicht nur für die jeweiligen deutschen Sonderzeichen. Alle Sonderzeichen der verschiedenen Sprachen sowie z. B. das Copyrightzeichen, Anführungszeichen und spitze Klammern werden durch die *Entities* abgedeckt.

In *HTML-Dokumenten* ist es ganz besonders wichtig, daß Sie im Text die spitze Klammer oder auch die Anführungszeichen nicht direkt verwenden, sondern als *Entities*, damit es nicht zu Fehlinterpretationen durch den Browser kommt.

Die Syntax eines *Entity*

Der Aufbau der *Entities* ist einheitlich, jedes *Entity* beginnt mit dem Zeichen &, gefolgt von der Umschreibung des Sonderzeichens und dann gefolgt vom Semikolon. Das Ö sieht als *Entity* folgendermaßen aus: *Ö*.

Wenn man das genauer betrachtet, steckt dahinter ein ganz logisches Konzept, das Sie in dieser Art und Weise auf alle deutschen Umlaute anwenden können. Der entsprechende Buchstabe kommt nach dem *&*, und zwar entsprechend als Großbuchstabe oder als Kleinbuchstabe. Angefügt wird einfach *uml* für Umlaut, gefolgt von einem Semikolon. So haben Sie sehr schnell die spezifischen deutschen Sonderzeichen zusammen, mit Ausnahme des *ß*, das folgendermaßen gekennzeichnet werden muß: *ß*.

Anstelle der *Entities* kann auch der numerische Wert aus dem Zeichensatz *ISO 8859* in der Syntax *&#Wert;* angegeben werden.

Die wichtigsten *Entities*

Zum Abschluß noch eine Auflistung der *Entities* und *ISO-8859-Codes*, die auf deutschsprachigen *HTML-Seiten* häufiger benötigt werden:

Sonderzeichen	ISO 8859	*Entity*
ä	&#228;	&auml;
Ä	&#196;	&Auml;
ö	&#246;	&ouml;
Ö	&#214;	&Ouml;
ü	&#252;	&uuml;
Ü	&#220;	&Uuml;
ß	&#223;	&szlig;
Leerzeichen		
"	&#34;	&quot;
&	&#38;	&amp;
<	&#60;	&lt;
>	&#62;	&gt;

Sonderzeichen	ISO 8859	*Entity*
§	&#167;	&sect;
©	&#169;	&copy;
®	&#174;	&reg;
¼	&#188;	&frac14;
½	&#189;	&frac12;
¾	&#190;	&frac34;

Tab. 3.1: Liste der wichtigsten Entities

3.5 Der Inhalt

HTML-Dokumente sind in erster Linie Textdokumente mit Verknüpfungen in andere Stellen des Dokuments oder in andere Dokumente. Es sind sogenannte *Hypertextdokumente*.

Texteingabe

Wenn in das Gerüst, das wir in Kapitel 3.2 kennengelernt haben, Text geschrieben wird, haben wir ein *HTML-Dokument*, das im Browser bereits eine Anzeige bewirkt.

Probieren wir das gleich einmal aus, indem Sie das nachfolgende *Listing* abtippen. Der Text muß zwischen den *Tags* <BODY> und </BODY> stehen. Sie erinnern sich sicher, daß zwischen diesen beiden *Tags* immer der Inhalt der Seite steht. Es ist der später sichtbare Teil.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">

<HTML>

<HEAD>

<TITLE>

</TITLE>

</HEAD>

<BODY>

      Meine erste HTML-Seite !

      So schwer ist das gar nicht !

</BODY>

</HTML>
```

Zeilenumbruch

Wie Sie in der Abbildung 3.2 sehen, ist die Darstellung jedoch recht unschön. *HTML* interessiert es nicht, ob im Listing eine neue Zeile beginnt. Wenn Sie einen Zeilenumbruch an einer bestimmten Stelle möchten, müssen Sie dort das *Tag* <*BR*> einfügen (engl. »break«). Dieses Tag benötigt keine Aufhebung.

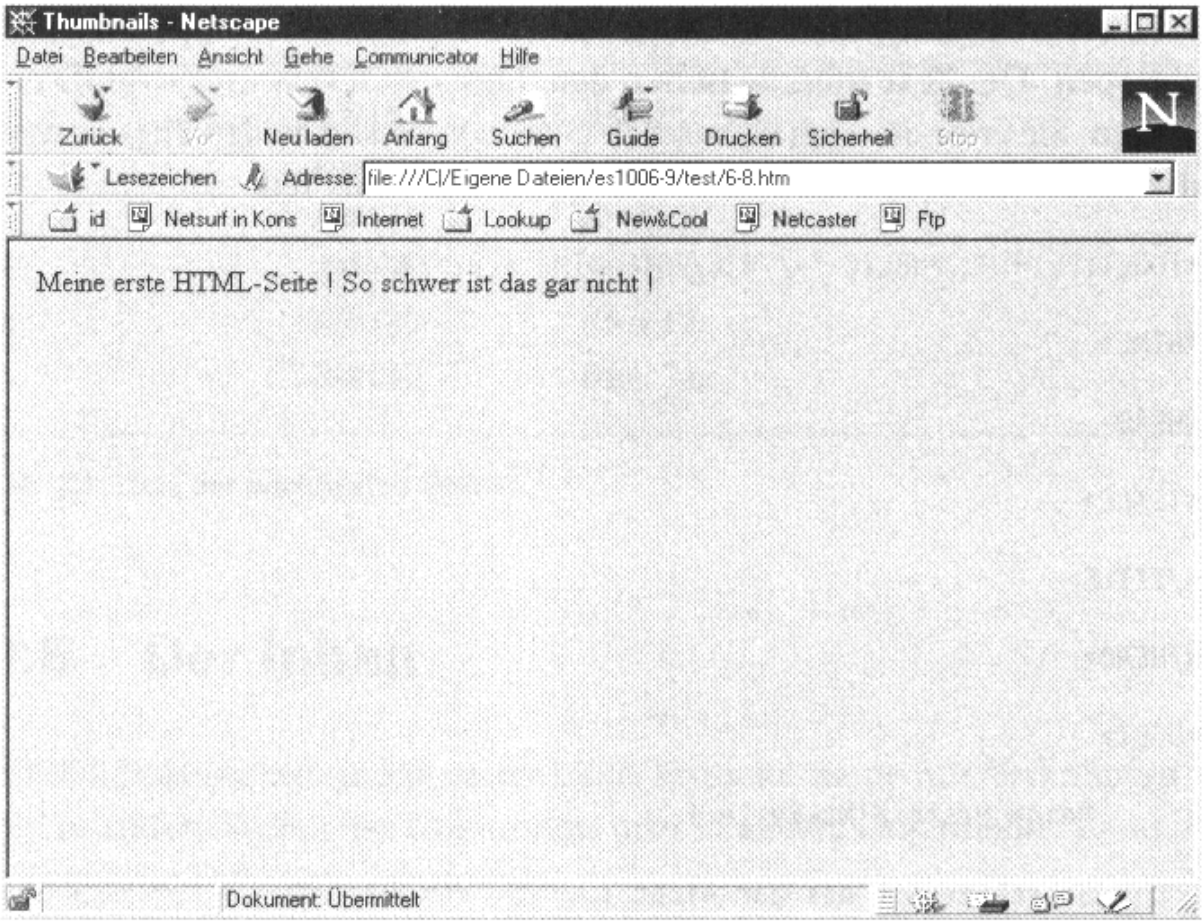

Abb. 3.2: Das erste HTML-Dokument

Wenn Sie nun wie im nachfolgenden Listing das *Tag* <*BR*> nach der ersten Zeile des Textes einfügen, dann erhalten Sie auch bei der Ausgabe durch den Browser den Text in zwei Zeilen.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">

<HTML>

<HEAD>

<TITLE>
```

```
</TITLE>

</HEAD>

<BODY>

      Meine erste HTML-Seite ! <BR>

      So schwer ist das gar nicht !

</BODY>

</HTML>
```

Absatz

In längeren Texten reicht es meist nicht, nur einen Zeilenumbruch einzufügen, es wird ein Absatz benötigt. Gerade lange Texte lassen sich dadurch übersichtlicher strukturieren.

Das *Tag* <*P*> leitet einen Absatz ein. Der Absatz enthält bereits den Zeilenumbruch, allerdings wird die neue Zeile erst nach einer Leerzeile begonnen.

Ersetzen wir im obigen Beispiel einfach das *Tag* <*BR*> durch das *Tag* <*P*> (engl. »paragraph«), dann führt dies bei den meisten Browsern zu der Darstellung, wie in Abbildung 3.3 zu sehen ist.

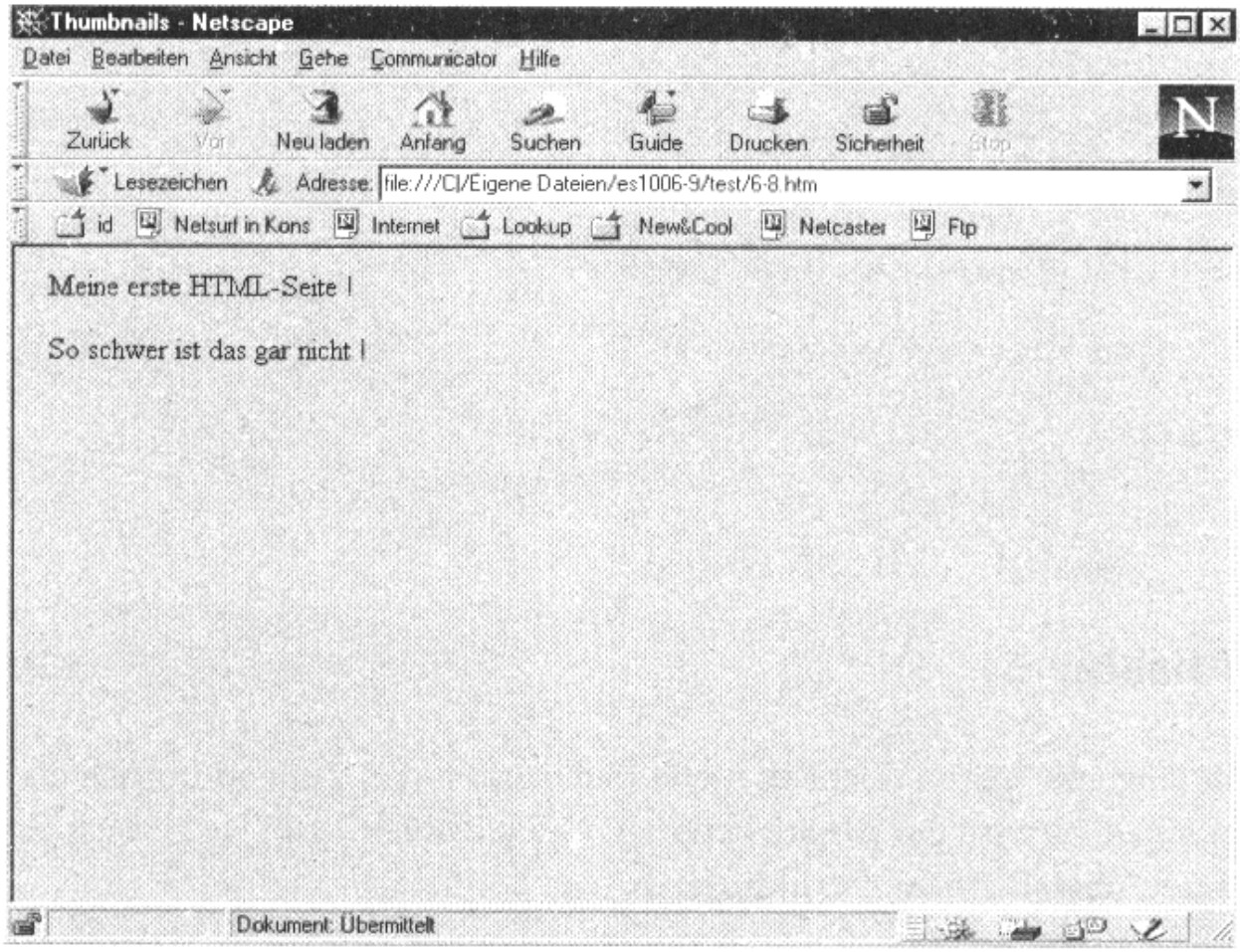

Abb. 3.3: Text nach Einfügen des Tags <P>

Entgegen der inzwischen recht weit verbreiteten Meinung sollte das *Tag* <*P*> nicht alleine stehen. Ein Absatz muß durch das *Tag* </*P*> beendet werden. Die meisten Browser interpretieren es zwar richtig, wenn der Absatz nur gesetzt, aber nicht beendet wird, da das Setzen eines neuen Absatzes den ersten beendet. Bei einigen Browsern kann es allerdings Probleme geben, und Sie wissen nicht, welchen Browser der Betrachter Ihrer Webpage verwendet.

Deshalb muß der Quelltext korrekt lauten:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN//">
<HTML>
<HEAD>
<TITLE>
</TITLE>
</HEAD>
<BODY>
      Meine erste HTML-Seite !
<P>
      So schwer ist das gar nicht !
</P>
</BODY>
</HTML>
```