1.4.2. Auszeichnungssprachen

Die Auszeichnung von Texten ist keine neue Erfindung: In der Form von Satz- und Korrekturanweisungen war sie ein unentbehrliches Hilfsmittel für Verfasser und Lektoren, um Schriftsetzern Anweisungen für die Umsetzung eines Manuskriptes mitzuteilen. In *Abb. 1.4/6* sind Beispiele solcher Anweisungen aus dem „Chicago Manual of Style“ dargestellt.

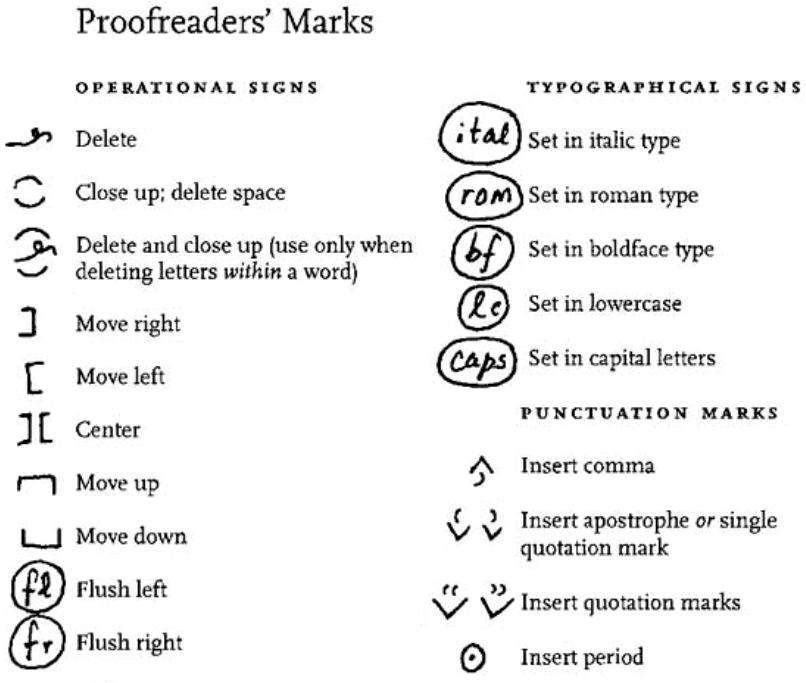

Abb.: 1.4/6 Korrektur- und Satzanweisungen als Beispiel einer Auszeichnung

Schon die Verwendung von Groß- und Kleinschreibung, Sperr- und Fettschrift ist eine Form der Auszeichnung. Genau genommen handelt es sich dabei um eine einfache Form *physischer Auszeichnung*. Physische Auszeichnung bedeutet, den Dokumentinhalt mit bestimmten Attributen zur visuellen Darstellung zu beschreiben und gehört somit zur **Layoutgestaltung**

In der Datenverarbeitung nennt man das Einbinden von „Tags“ in eine konventionelle Textdatei ebenfalls ***Auszeichnen***[1]. Bei Tags handelt es sich um Anweisungen, die aus zwei Teilen (dem Start- und dem Endtag) bestehen. Dazwischen befindet sich entweder normaler Text oder wieder ein Tag. Im letzteren Fall spricht man von einer sogenannten *Schachtelung* von Tags. Da Tags ebenfalls aus Textzeichen bestehen, ist es notwendig, diese vom „restlichen“ Text unterscheiden zu können. Dazu dienen Trenn- bzw. Deliminatorzeichen. Üblicherweise werden dafür in **Auszeichnungssprachen**, wie in HTML das Größer- und Kleinerzeichen verwendet. Das Gegenteil einer Auszeichnungssprache sind Dateiformate, die

[1] Genaugenommen handelt es sich hierbei um ein *beschreibendes Auszeichnen (descriptive markup),* das im Gegensatz zum *prozeduralen Auszeichnen (prozedural markup)* steht.

ein *spezific coding*[2] verwenden. Beispielsweise kann man mit speziellen Tags in HTML einen Text mit einer bestimmten Schriftart, Farbe und Größe versehen. Die physische Auszeichnung kann sich aber auch auf das gesamte Dokument erstrecken. So gehören etwa Tags für die Angabe der Seitenbreite ebenfalls dazu. Davon unterscheidet sich die *logische Auszeichnung* eines Textes, die die **Struktur** eines Dokuments (wozu auch Hypertexte gehören) beschreibt. Diese umfasst beispielsweise Überschriften, Aufzählungen, Absätze, Rahmen, Listen und *Verweise*. Es ist natürlich für eine Aufnahme durch den Menschen sinnvoll, diese Textelemente auch unterschiedlich optisch darzustellen. Der Begriff Auszeichnung („Markup“) ist auch in der Bezeichnung HTML enthalten: Hypertext Markup Language. Aus dieser Bezeichnung ist weiters ersichtlich, dass mit dieser Sprache sogenannte Hypertexte, das sind nicht lineare Texte, ausgezeichnet werden. Die Verbindungen zwischen einzelnen Hypertexten, sogenannte Links, stellen einen Teil der **Struktur** eines Hypertextes dar.

Ziel der logischen Auszeichnung ist neben der Strukturierung vor allem auch eine computergestützte Weiterverarbeitung des Hypertextes. Für verschiedene Ausgabegeräte wirken bestimmte physische Darstellungen unterschiedlich. Aus diesem Grund ist auch eine Trennung von Strukturbeschreibung und optischer Darstellung sinnvoll. Dafür stehen in HTML die Cascading Style Sheets (CSS) zur Verfügung. Diese saubere Trennung von Layout und Struktur ist in der täglichen Praxis bei vielen HTML-Seiten nicht gegeben. Grund dafür ist, dass alte Browserversionen CSS nicht unterstützen und der Einsatz von Style-Sheets noch immer nicht so verbreitet ist[3].

Für einen möglichst flexiblen Einsatz ist eine Trennung[4] von Struktur, Inhalt und Format Voraussetzung. Die Vermischung von Struktur und Format in HTML wurde trotz der offensichtlichen Nachteile anfänglich als nicht weiter störend empfunden. Die Benutzung von Formatierungsbefehlen innerhalb eines Dokuments entsprach auch eher der gewohnten WYSIWYG[5]-Sichtweise und man dachte auch noch nicht an eine Aufbereitung für verschiedene Ausgabegeräte. In den Anfangszeiten des Internets war ein leicht zu erlernendes und einfach anzuwendendes Werkzeug auch wichtiger. Somit trugen diese „Designschwächen“ von HTML wesentlich zur Verbreitung des WWW bei. Aber spätestens beim arbeitsteiligen Erstellen von Internetseiten, der Aufbereitung gleicher Inhaltsquellen für unterschiedliche Ausgabegeräte, der Anwendung von unternehmensweit geltenden Style Guides, Versionsverwaltung und ab bestimmten Projektgrößen ist eine saubere Trennung von Struktur, Inhalt und Layout unabdingbar[6].

---

2 Darunter versteht man direkt in eine Datei geschriebene Formatierungsbefehle und Makros. Das Gegenteil davon ist *generic coding*, das als der Vorläufer moderner Auszeichnungssprachen angesehen werden kann.

3 Das Hardcodieren von Layoutinformationen ist nach wie vor im Amateurbereich üblich, während in professionellen Umgebungen die Verwendung von Style Sheets mittlerweile gang und gäbe ist.

4 Es ist nicht Voraussetzung, dass diese Teile auch physisch, d. h. über mehrere Dateien getrennt werden, siehe dazu das Beispiel HTML und CSS im nachfolgendem Kapitel.

5 WYSIWYG = **W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et. Eine aus Textverarbeitungsprogrammen bekannte Funktionalität, die den Text am Bildschirm nicht nur im fertigen Layout anzeigt, sondern auch ein direktes Editieren in dieser Ansicht ermöglicht.

6 Diese Aufgaben werden unter dem Begriff des WCM (Web Content Management) zusammengefasst.

Es erscheint etwas restriktiv, wenn immer von Texten bzw. von Hypertexten die Rede ist, handelt es sich doch bei Internetseiten heutzutage durchgehend um multimediale Darstellungen. Der Begriff Hypertext lässt jedoch auf eine lediglich monomediale Darstellung schließen. Grund für die weitere Verwendung des Begriffs „Text“ ist, dass die Beschreibung multimedialer Internetseiten ebenfalls durch einfache Texte erfolgt. Dieses Textformat ist die Grundlage für die Einbindung von Multimediaobjekten.

Neben HTML gibt es noch andere Möglichkeiten, um multimediale Dokumente zu strukturieren und zu speichern. Gemeinsam ist diesen Datenformaten, dass sie hauptsächlich auf eine physische Beschreibung eines Dokuments abzielen:

- *Textbearbeitungsprogramme*, die die Integration von Multimediaobjekten und Hyperlinks ermöglichen.
- *Seitenbeschreibungssprache*n, wie Postscript und PCL (für den Druckbereich)
- *Dokumentformate,* wie beispielsweise PDF (Portable Document Format der Firma Adobe), das auch Hyperlinks auf Stellen im Dokument und auf Internetressourcen integriert.

Im Unterschied zu HTML handelt es sich bei diesen Dateiformaten jedoch um keine Auszeichnungssprachen und die Struktur, der Inhalt und die Daten werden in einer gemeinsamen Datei gespeichert (*specific coding*).

Die Möglichkeiten zur physischen Auszeichnung in HTML haben seit CSS 2.0 einen Stand erreicht, der sogar Hoffnungen nährt, dass HTML in naher Zukunft als universelle Seitenbeschreibungssprache und als universelles Dokumentformat eingesetzt werden wird. Ein mögliches Anzeichen dafür ist sicherlich auch, dass HTML mittlerweile von allen Office-Pakten als alternatives Exportdateiformat[7] angeboten wird. Im Gegensatz zur Layoutgestaltung sind die Tags zur Inhaltsstrukturierung in HTML weniger weit fortgeschritten. Dies ist nicht unbedingt als Nachteil zu sehen, da die Einbindung von Tags zur physischen Präsentation und der geringe Sprachumfang mit ein Grund für die leichte Erlernbarkeit von HTML gewesen sind. Darüber hinaus kann HTML im Unterschied zu vielen Textverarbeitungsdateiformaten, zu Desktop-Publishing Dateiformaten und zu Dokumentformaten als ein *plattformunabhängiges* Dateiformat betrachtet werden. Will man jedoch all diese Vorteile für andere Zwecke als zur Gestaltung von Internetseiten nutzen, stößt man bald an die Grenzen des Machbaren bzw. des Praktikablen. Aufgrund der eingeschränkten Möglichkeiten zur Strukturierung von Inhalten ist der Einsatz von HTML zur Erstellung von Dokumenten[8] für eine informationstechnische Weiterverarbeitung wenig sinnvoll. Die Befehle zur Strukturierung eines Dokuments sind auf Internet-Hypertexte ausgelegt und verbindlich definiert, ohne dass eigene Erweiterungen möglich sind. Für das Publizieren von einfachen Doku-

---

7 Davon zu unterscheiden ist das **interne** Speicherformat der Office-Suiten. Star Office 6.0 verwendet hiefür schon durchgehend XML.

8 Mit dem Begriff Dokument sind hier nicht nur Textverarbeitungsdateien gemeint, sondern strukturierte und semistrukturierte Daten im Allgemeinen: Texte, Tabellen, Diagramme, technische Zeichnungen bis hin zu Transaktionen in ERP-Systemen und compound Documents.

menten im Internet macht dies durchaus Sinn. Typische Internetdokumente stellen keine allzu großen Ansprüche in Hinsicht auf Strukturierungsmöglichkeiten. Das Hinzufügen eigener Tags in HTML ist auch nicht wünschenswert bzw. nicht sinnvoll, da sich ja die Internetclients verschiedenster Hersteller auf einen definierten Standard der Sprache beziehen sollen.

Anders als bei diesen **layoutorientierten** Aufgaben sieht es jedoch bei **datenzentrierten** Aufgaben aus. Dabei handelt es sich um Probleme, die hauptsächlich bei der Speicherung und Übertragung von Daten entstehen, die im betrieblichen Alltag anfallen. Diese Daten besitzen eine komplexere innere Struktur und sollen nicht nur für eine Präsentation im Internet aufbereitet, sondern auch möglichst einfach elektronisch weiterverarbeitet werden können. Ein möglichst komfortabler Austausch von Dokumenten und eine einfach zu handhabende informationstechnische Weiterverarbeitung sind aber auch für layoutzentrierte Dokumente wünschenswert. Immer öfters besteht für das betriebliche Tagesgeschäft die Notwendigkeit, dass solche layoutorientierten Softwaresysteme innerhalb heterogener EDV-Infrastrukturen, aber auch zwischen den EDV-Landschaften kooperierender Organisationen zusammenarbeiten sollen. Neue technische Anforderungen an solche Systeme kommen auch aus den Führungsbereichen der Organisationen selbst: Es besteht ein vitales Interesse daran, daten- und dokumentzentrierte Quellen mit unterschiedlichen Formaten für das *„Knowledge-Management“* und darauf aufbauende Systeme zusammenzuführen und zu verarbeiten. Aber auch für ein intelligenteres „Suchen und Finden“ im Internet, beispielsweise für Suchmaschinen und Softwareagenten, ist HTML nicht der Weisheit letzter Schluss.

Für diese Anwendungsbereiche ist nicht nur ein plattformübergreifendes Format erforderlich, sondern es muss zudem von einem Anwendungsbereich unabhängig sein und auch die Möglichkeit vorsehen, beliebige bedeutungsspezifische (d.h. semantische) Informationen in ein Dokument zu integrieren. In HTML existieren zwar Ansätze dazu, wie etwa Metatags zur Beschreibung eines Dokumentes, doch sind diese grundsätzlich nicht erweiterbar. Wünschenswert ist eine Auszeichnungssprache, die sich nicht auf eine vorgegebene Grammatik und ein nicht erweiterbares Vokabular beschränkt, sondern mit der es möglich ist, eigene Auszeichnungssprachen zu kreieren, mit denen sich beliebig komplexe Dokumentstrukturen abbilden lassen. Eine Auszeichnungssprache, die diese wünschenswerten Eigenschaften erfüllt, wird als *Metasprache* bezeichnet. Der bisher umfassendste Ansatz für eine solche Metasprache stellt **SGML** (**S**tandard **G**eneralized **M**arkup **L**anguage) dar, die aufbauend auf die Arbeiten zu GML[9] (**G**eneralized **M**arkup **L**anguage) im Jahre 1986 standardisiert wurde. GML seinerseits geht bereits auf Arbeiten aus dem Jahre 1969 zurück, als Charles Goldfarb für IBM eine Auszeichnungssprache für komplexe juristische Dokumente entwarf. Damals schon stand die elektronische Weiterverarbeitung (hauptsächlich Information Retrieval Anwendungen) und nicht nur ein *generic coding* (im Unterschied zum *specific coding)* im Mittelpunkt der Arbeiten. Im Jahre 1974 schließlich entwickelte Goldfarb SGML, das zwölf Jahre später unter seiner Führungsarbeit als ISO-Standard verabschiedet wurde. Das Konzept der generischen Auszeichnung war in der Druckindustrie schon seit den frühen 60er Jahren bekannt. Das Revolutionäre an diesen Arbeiten war die Übertragung dieser Idee auf das elektronische Dokumentenmanagement.

---

9 „Zufälligerweise“ kann man das Akronym GML auch aus den Anfangsbuchstaben der Familiennamen der Erfinder zusammengesetzt sehen: **G**olfarb, **M**osher, **L**orie

SGML wird in zahlreichen Industrien für das Dokumentmanagement eingesetzt. Erwähnenswert sind vor allem die Automobil- und die Rüstungsindustrie, die beide aufgrund der komplexen Natur der Dokumente (Versionsverwaltung, Multilingualität, Interdependenzen, etc.) frühzeitig SGML für ihre Anwendungsbereiche adaptierten. Der Nachteil von SGML ist, dass es als sehr komplex und schwierig gilt. Abhilfe dafür verspricht die Extensible Markup Language[10] (XML), die mit Hilfe von SGML definiert wurde. Die Spezifikation von XML 1.0 umfasst lediglich einige wenige Seiten und gilt so wie HTML als leicht erlernbar, was u. a. ein Grund dafür ist, dass XML in den vergangenen Jahren soviel Aufmerksamkeit zuteil wurde. Der wahre Grund liegt aber vor allem im Potenzial von XML als Integrationstechnologie für alle Bereiche der Informationsverarbeitung und Übermittlung strukturierter Daten.

In den nächsten beiden Kapiteln werden die grundlegenden Sprachelemente von HTML und XML behandelt. Auch wenn Dokumente nicht (mehr) mit Hilfe der einzelnen Befehle dieser Sprachen direkt erstellt werden sondern mit Anwendungsprogrammen, ist ein Grundverständnis der Funktionsweise dieser Sprachen bei der Verwendung von Tools (z.B. HTML- und XML Editoren) hilfreich.

1.4.2.1 Grundkonzepte der Hypertext Markup Language

Im Vergleich zu einem herkömmlichen Desktop-Publishing Programm sind den Gestaltungsmöglichkeiten im W3 noch gewisse Grenzen gesetzt. Die Entwicklung und Normung von HTML ist aber einem steten Erweiterungsprozess unterworfen, wobei neben dem aktuellen Standard HTML 4.01 bzw. xHTML 1.0 von einzelnen Client-Entwicklern vom Standard abweichende Zusatzfeatures implementiert werden. Mit einer weitgehenden Unterstützung der durch das W3C bereits standardisierten **CCS-2** ist dadurch echte Seitengestaltung bei Web-Seiten möglich sein.

Genau betrachtet müsste es eigentlich Hypermedia Markup Language heißen, da viele Dokumente im W3 hypermedial, mit eingebetteten Grafiken, Links auf Videos, Animation, Sound und dynamischen Inhalten (siehe Kap. 1.4.2.1.2) gestaltet sind. Ein weiterer oft benutzter Ausdruck ist Dynamic-HTML (DHMTL). Dabei handelt es sich aber um keine neue Sprachversion oder einen Standard, sondern um eine marketingorientierte Bezeichnung der großen Browserhersteller. Gemeint sind damit die Möglichkeiten, dynamische Inhalte in einfachen HTML-Code einzubinden. Mit der Abkehr von proprietären Entwicklungen und der zunehmenden Unterstützung des gesamten HTML-Sprachumfanges, von Stylesheets, standardisierten Skripting bzw. Programmiersprachen in aktuelle Webbrowser ist das Schlagwort DHMTL als Mittel zur Produktdifferenzierung jedoch in den Hintergrund geraten. Anhand von einfachen Beispielseiten im Darstellungsmodus (vgl. *Abb. 1.4/6*) und im Quellcode (vgl. *Abb. 1.4/7*) werden die wichtigsten HTML-Befehle erklärt.

Überschrift

[10] Zum Selbststudium besser geeignet ist die von Tim Bray **kommentierte Version** http://www.xml.com/-axml/testaxml.htm

Es folgt ein normaler Text, der auch die deutschen Umlaute (ÄäÖöÜüß) darstellen kann, sowie eine Reihe von Sonderzeichen ((©Ñ_) enthält. An einfachen Textformatierungs-möglichkeiten stehen **Fett**, *kursiv* oder **Hervorhebung** zur Verfügung. Ein Text kann auch `Vorformatiert`
angezeigt werden und Textteile durch eine Linie getrennt werden.

Zentrierte Absatzüberschrift

Um Texte zu strukturieren, können Aufzählungslisten verwendet werden.

- Item 1
- Item 2
- Item 3

Mit einem Anker kann ein Link auf eine bestimmte Textstelle in einem Hypertextdokument erzeugt werden. Hier folgt ein Link auf ein Dokument auf einen anderen HTTP-Server.

Abb. 1.4/7: Hypertextdokument im Präsentationsmodus

Wie aus dem Quellcode ersichtlich, könnte das HTML-Dokument mit einem beliebigen Texteditor oder einer Textverarbeitung erstellt worden sein. Jedes HTML-Dokument beginnt mit dem Tag <HTML> und endet mit dem Tag </HTML>. Eine zu formatierende Textstelle wird grundsätzlich mit einem Anfangs- und Ende-Tag markiert, wobei dem Ende-Tag ein Schrägstrich (/) vorgestellt wird. Diese, einen Text umfassenden Tags, werden auch *Container* genannt. Das Gegenteil dazu sind Tags, die kein Endtag haben, wie beispielsweise das erste Tag im Beispiel aus *Abb. I-2.7*. Das Tag <DOCTYPE> ist optional und verweist auf die verwendete HTML Version und Sprache.

Eine weitere Untergliederung erfolgt in den zwischen die Tags <HEAD> und </HEAD> eingeschlossenen Header und dem eigentlichen Textelement zwischen den Tags <BODY> und </BODY>. Der Titel <TITLE> eines HTML-Dokuments kann in Datenbanken für Suchzwecke verwendet werden und wird im Normalfall in der ersten Zeile des W3-Clients angezeigt.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
<HTML>
<HEAD>
<TITLE>Ein HTML-Dokument </TITLE>
</HEAD>
<BODY>
<H1>Überschrift</H1>
Es folgt ein normaler Text, der auch die deutschen Umlaute (&Auml;&Ouml;&Uuml;
&auml;&ouml;&uuml;&szlig;) darstellen kann, sowie eine Reihe von Sonderzeichen
(&copy;&Ntilde;&THORN;) enth&auml;lt. An einfachen Textformatierungs m&ouml;glichkeiten stehen
<STRONG>Fett </STRONG>,<I> kursiv</I> oder <B>Hervorhebung</B> zur Verfügung. Ein Text kann
auch<PRE> Vorformatiert</PRE> angezeigt werden und Textteile durch eine Linie getrennt
werden.<BR><HR></HR>
<CENTER><H2>Absatzüberschrift</H2></CENTER>
Um Texte zu strukturieren, können Aufzählungslisten verwendet werden.
<UL><LI> Item 1 </LI>
<LI>Item 2 </LI>
```

</OL>

```
<LI>Item 3 </LI></UL>

Mit einem<A HREF="#Anker1/dokument1.html"> Anker</A> kann ein Link auf eine bestimmte
Textstelle in einem Hypertextdokument erzeugt werden. Hier folgt ein <A HREF=
"http://www.idv.edu/home.html">Link</A> auf ein Dokument auf einen anderen HTTP-Server.

</BODY></HTML>
```

Abb. 1.4/8: HTML-Quellcode der Darstellung aus Abb. 1.4/6

Im eigentlichen Textteil <BODY> gibt es wesentlich mehr Möglichkeiten, Tags einzusetzen. **Überschriften** können in sechs verschiedenen Größen definiert werden, wobei die Festlegung der Schriftart und Größe für die einzelnen <Hx>-Tags an den W3-Clients eingestellt werden kann. Die Syntax lautet **<Hx>text </Hx>**, wobei x eine Zahl zwischen 1 und 6 ist und die Überschriften-Ebene definiert.

Umlaute und **Sonderzeichen** werden durch spezielle Codes in der Form von **&xxx;** dargestellt und von den meisten HTML-Editoren automatisch konvertiert. Die entsprechenden Werte können aus Tabellen entnommen werden.

Zeilenschaltungen (CR) werden von W3-Clients ignoriert, d. h. der Text wird der Fenstergröße des Clients entsprechend umgebrochen. Um einen erzwungenen Zeilenumbruch durchzuführen, wird der Tag
 verwendet. Der Tag <P> Paragraf hat eine ähnliche Funktion, jedoch wird eine Leerzeile zusätzlich hinzugefügt.

Um Textteile durch ein **horizontale Linie** zu trennen, wird der <HR> Tag eingesetzt.

Zur Zeichenformatierung werden sogenannte **Stil-Tags** eingesetzt, deren wichtigste folgende sind:

- <EM> einfache Hervorhebung (Emphasize)
- <STRONG> starke Hervorhebung
- <B> Fettdruck (Bold)
- <I> Kursiv (Italics)

Um einen Text genauso darzustellen, wie er eingegeben wurde, inklusive aller Leerzeichen, Zeilenschaltungen, Tabulatoren etc. wird der <PRE> (preformated) Tag verwendet.

```
<DL><DD> Linz <BR>

Linz ist die Hauptstadt von Oberösterreich und die drittgrößte Stadt Österreichs. Einst
römisches Kastell, im Mittelalter Residenz deutscher Kaiser, heute dynamisches Industrie-,
Wirtschafts- und Handelszentrum mit über 1000jähriger Geschichte und Kultur.

<DD> Steyr <BR>
```

```
Steyr gehört zu den ganz wenigen historischen Städten, die eine weitgehend geschlossene mit-
telalterliche Bausubstanz aufweisen. Es ist nahezu das ganze Ensemble der Altstadt erhalten.
Die Grundlage für den wirtschaftlichen Aufschwung der Stadt war immer der Handel und die
Verarbeitung von Eisen und Stahl. Steyr wurde deshalb auch die alte Eisenstadt genannt.

</DL>
```

Abb. 1.4/9: Quellcode einer Definition List

Listen können als aufzählende Listen, numerierte Listen oder beschreibende Listen ausgeführt werden. Eine aufzählende Liste startet mit dem <UL> Tag (**U**nnumbered **L**ist), die einzelnen Listeneinträge werden so wie bei allen anderen Listenformen mit dem <LI> Tag eingetragen und enden mit dem </UL> Tag (siehe Beispiel im Quellcode der Abb. I-2.8). Sollen die einzelnen Listeneinträge durchnumeriert werden, so wird mit dem <OL> Tag (**O**rdered **L**ist) begonnen und mit dem </OL> Tag abgeschlossen. Einen Sonderfall stellt die beschreibende Liste <DL> (**D**efinition **L**ist) dar, deren Listeneinträge mit dem <DD> Tag gekennzeichnet werden.

Ein weiteres Strukturelement von HTML-Seiten sind **Tabellen**. Tabellen, deren Rahmen und Gitternetzlinien nicht sichtbar sind (sogenannte transparente Tabellen), stellen ein einfaches Mittel zur Positionierung von Elementen zur Verfügung. Tabellen werden ebenfalls mittels Tags realisiert. Wegen der Möglichkeit zur Schachtelung von Tags kann eine Tabellenzelle wiederum Elemente wie Listen, Tabellen, Bilder, Links etc. enthalten und es ist ein Positionieren der Elemente einfach möglich. Ein pixelgenaues Positionieren ist in HTML damit nicht möglich und ist von der, der Sprache zugrundeliegenden Philosophie[11] auch nicht beabsichtigt. Für diese typischen Seitengestaltungsaufgaben sind die Cascading Style Sheets (CSS) vorgesehen, die auch eine auf das Ausgabemedium hin optimierte Gestaltung[12] ermöglichen. CSS werden detaillierter später dargestellt.

Die volle Hypertextfunktionalität in HTML kommt aber erst durch die Möglichkeit des Erzeugens von **Links** (Querverweise) zur Geltung. Links werden mit dem <A> Tag (**Anchor**) hergestellt. Um externe Links zu realisieren, wird das Attribut **HREF** (**H**yper**T**ext **R**eference) verwendet.

<A HREF= ”Page2”> Seite 2 </A>: Dieser Link verweist auf ein Dokument ”Page2”, welches sich im gleichen Verzeichnis wie das aktuelle Dokument befindet. Am Bildschirm erscheint der Text ”Seite 2”, der wie alle Links farblich hervorgehoben oder unterstrichen ist.

Linz

Linz ist die Hauptstadt von Oberösterreich und die drittgrößte Stadt Österreichs. Einst römisches Kastell, im Mittelalter Residenz deutscher Kaiser, heute dynamisches Industrie-, Wirtschafts- und Handelszentrum mit über 1000jähriger Geschichte und Kultur.

11 Eine Auszeichnungssprache sollte idealerweise unabhängig vom Ausgabemedium sein

12 Wenn oftmals auch nur mit Tricks wie z. B. mittels Grafiken

Steyr

Steyr gehört zu den ganz wenigen historischen Städten, die eine weitgehend geschlossene mittelalterliche Bausubstanz aufweisen. Es ist nahezu das ganze Ensemble der Altstadt erhalten. Die Grundlage für den wirtschaftlichen Aufschwung der Stadt war immer der Handel und die Verarbeitung von Eisen und Stahl. Steyr wurde deshalb auch die alte Eisenstadt genannt.

Abb. 1.4/10: Ergebnis der Definition List aus Abb. 1.4 /9

Mit *<A HREF= "#Linkname"> Linkname </A>* kann auf eine bestimmte Stelle innerhalb eines Dokuments verwiesen werden, die mit *<A NAME= "Linkname"> Linkname*[13] *</A>* gekennzeichnet wurde.

Um Links zu Dokumenten auf anderen Servern zu erstellen, werden die in der Einleitung schon beschriebenen URL-Adressen verwendet. Ein Link auf die Homepage des idv-HTTP-Servers hätte somit folgendes Format:

<A HREF="http://www.idv.edu/default.html">idv Home-Page

Eingebettete Grafiken (Inline Images) und Multimediaobjekte**.** Ein Bild sagt mehr als tausend Worte und so ist auch im W3 die Verwendung von in das HTML-Dokument eingebetteten Grafiken ein wesentliches Gestaltungselement. Mit dem <IMG SRC="image_URL" ALIGN = TOP/MIDDLE/BOTTOM> Tag können Grafiken im GIF-, JPEG- oder X-Bitmap Format in ein Dokument eingebunden werden. Mit dem optionalen ALIGN wird festgelegt, ob der folgende Text oben, in der Mitte oder unten an das Bild gestellt werden soll.

Multimediaobjekte wie Videosequenzen und Sounddateien werden nicht von allen W3-Clients abgearbeitet, sondern müssen mit einer externen Helper-Application oder einem Browser-Plug-In ausgeführt werden. So ist z. B. für das Abspielen eines Quicktime-Movies das Vorhandensein des Apple Quicktime-Players notwendig.

Eine spezielle Art der Grafikeinbindung sind sogenannte ***Clickable Images***; das sind Grafiken, durch deren Anklicken an verschiedenen Bereichen unterschiedliche Links ausgelöst werden können. Ein Beispiel (vgl. *Abb. 1.4/11*) ist die Grafik von Österreich[14] mit eingezeichneten Bundesländergrenzen, in der durch Anklicken auf die Fläche eines Bundeslandes der Link zur betreffenden Leitseite ausgeführt wird. Der Link wird durch ein Serverscript, in dem die dazugehörigen Koordinaten der Grafikbereiche eingetragen sind, hergestellt. Im HTML-Dokument selbst erfolgt nur eine Kennzeichnung durch den <ISMAP> Tag, der z. B. folgenden Aufbau hat: * src= "/Doris-/Oesterreich. gif" ismap> *. Die HTML3 Spezifikation definiert Tags für Clickable-Images, die am W3-Client abgearbeitet werden.

Frames werden schon seit langem von den gängigsten Browsern unterstützt und sind nunmehr auch ab Version 4.0 (1997) Bestandteil des HTML-Sprachumfanges. Zuvor hatten

13 Der Linkname ist optional.

14 Vgl. http://www.oesterreich.com (20. 1. 1999)

die Browserhersteller jeweils eigene Sprachkonstrukte in die Internet-Clients integriert. Bei Frames handelt es sich um eine zusätzliche Gestaltungsmöglichkeit von Hypermedia-Dokumenten. Mit Hilfe von Frames kann eine Seite in mehrere Fenster unterteilt werden. Interessant sind Frames vor allem deshalb, da der Inhalt eines Fensters in Abhängigkeit eines gewählten Links aus einem anderen Fenster variiert werden kann. Typische Einsatzbereiche für Frames sind Inhaltsverzeichnisse und Hilfefunktionen. Im Unterschied dazu teilen eingebettete Rahmen (**iFrames**[15]) den Bildschirm nicht in unterschiedliche Rahmen auf, sondern stellen Bereiche dar, in denen fremde Inhalte angezeigt werden können.

Abb. 1.4/11: Clickable Image und Frames am Beispiel Oesterreich.com

Formulare in HTML-Dokumenten sind eine weitere Möglichkeit der Interaktion im W3. Die Abarbeitung der eingegebenen Daten erfolgt wie bei den Clickable-Images durch ein Serverscript, wo dann z. B. auch eine Host-Datenbank abgefragt werden kann.

1.4.2.1.1. Cascading Style Sheets

Ein wesentliches Merkmal einer Auszeichnunsgssprache ist die Trennung von Inhalt, Struktur und Layout der Dokumente. Dieses Prinzip wurde anfänglich in HTML nicht konsequent umgesetzt: Es existieren zahlreiche Tags zur Layoutgestaltung. Zur Trennung von Layout und Struktur dienen *Cascading Stye Sheets* (CSS), die mit den HTML-Dokumenten verknüpft werden oder in eine HTML- Datei direkt eingebunden werden können. Die Funktionsweise von CSS1 (standardisiert 1996) ist in etwa vergleichbar mit Druckformatvorlagen in Textverarbeitungsprogrammen. Die Möglichkeiten bei der Verwendung von CSS2 (standardisiert bereits 1998) gehen dabei, soweit dies der Client unterstützt, im Bereich des

15 iFrames seit HTML 4.0 Teil des Sprachumfanges.

seitenorientierten Layouts weit über die Tags zur optischen Präsentation in HTML hinaus und in Kombination mit HTML eignet sich CSS auch als vollwertige Seitenbeschreibungssprache, wie beispielsweise Postscript. Die Ausgabe ist aber nicht nur auf eine optische Präsentation begrenzt, es existieren auch Befehle für die akustische Ausgabe.

```
<HTML>

   <HEAD>

    <STYLE type="text/css">

     H1,P { font-size:12pt; font-style:bold; }

     P.gross { font-size:24pt; }

     P.klein { font-size: 8pt; }

    </STYLE>

   </HEAD>

   <BODY>

    <H1>Überschrift 1. Ordnung</H1>

    <P class="gross">Grosser Textabsatz</P>

    <P class="klein">Kleiner Textabsatz</P>

   </BODY>

</HTML>
```

Abb. 1.4/12: In HTML Quellcode eingebundene Style Sheets mit Formatunterklassen

Style Sheets kann man im Kopf einer HTML-Datei definieren, wobei diese Formatangaben lediglich für diese eine Datei gelten. Will man ein *konsistentes Layout* über mehrere Dateien, ist es erforderlich, diese Formatdefinitionen in einer eigenen Datei anzugeben. Die Verwendung mehrerer CSS-Dateien ist auch möglich. Dies ist dann erforderlich, wenn man verschiedene Style-Sheets für mehrere Ausgabegeräte bereitstellen möchte. Dies gilt nicht nur für Ausgabegeräte, die unterschiedliche Medien unterstützen (z. B. Sprache und Schrift), sondern auch für Ausgabegeräte, die das selbe Medium benutzen. So wirkt beispielsweise eine optische Ausgabe auf einem Drucker anders als auf einem Monitor.

Im Beispiel der *Abb. 1.4/12* werden Formatvorlagen direkt in den Kopf der HTML-Datei eingebunden. Dies erfolgt mittels des HTML-Tags <*style*>. Das Attribut type="text/css" gibt an, dass die Formatvorlagen mit Hilfe von CSS definiert werden. Bis zum abschließenden Endtag erfolgt nun die Verwendung von CSS Befehlen. In diesem Beispiel wird ein Sytee-Sheet für das HTML header-Tag erster Ordnung definiert. Zuerst wird das gewünschte HTML-Tag angegeben, für das das Style-Sheet definiert werden soll, gefolgt von Attributen in geschwungenen Klammern. Jedesmal, wenn nun in der HTML-Datei dieses Tag codiert

wird, gibt der Browser diese Überschriften in einer Größe von 12 Punkten und in Fettdruck aus. Die Möglichkeiten zur optischen Aufbereitung sind freilich um einiges umfangreicher als in diesem einfachen Lehrbeispiel. Mit einem Beistrich getrennt erfolgt die Formatangabe für Tags, die dieselben Formateigenschaften haben sollen. In dem Beispiel handelt es sich dabei um Textabsätze, die genauso wie Überschriften erster Ordnung dargestellt werden sollen. Die nächste Formatvorlage gilt wiederum für Absätze, dabei erfolgt jedoch eine Unterscheidung in die zwei Klassen „gross“ und „klein“, mit denen Textabsätze in verschiedenen Formatierungen ausgegeben werden können. In diesem Beispiel unterscheiden sich die beiden Formatierungen lediglich in der Schriftgröße. Wäre kein Style-Sheet für einen „klassenlosen“ Absatz definiert worden, würde die Ausgabe mit den Browservoreinstellungen erfolgen. Die Verwendung in HTML erfolgt durch Angabe einer definierten Klasse als Attribut.

1.4.2.1.2 JavaScript, Java und ActiveX

Die rasche Verbreitung der Programmiersprache Java[16] wurde in erster Linie durch das WWW, wo die Einbettung von kleinen systemunabhängigen Programmen (executable content) in HTML-Dokumente einen neuen Qualitätsschub hervorbrachte, gefördert. Java wurde 1990 von einer Projektgruppe der Firma Sun für den Bereich Consumer-Electronic entwickelt. Portabilität, das heißt Unabhängigkeit von Betriebssystemen und Hardware auf der Endgeräteseite, war die oberste Prämisse bei der Entwicklung. Dadurch war es naheliegend, Java auch als Programmiersprache am Internet einzusetzen, und so veröffentlichte Sun 1995 Java und einen Java-fähigen WWW-Browser.

Von der Programmiersprache Java zu unterscheiden ist JavaScript[17], eine Makrosprache, die zwar auf Java basiert, aber nur einen kleinen Teil der Funktionen von Java implementiert. JavaScript wird als Quellcode direkt in die HTML-Seite eingebettet. JavaScript wurde von der Firma Netscape für den Navigator 2.0 unter dem Namen LiveScript entwickelt. Ein häufiges Einsatzgebiet für JavaScript ist die Steuerung und Überprüfung von Formulareingaben in HTML-Seiten.

1.4.2.2. Exkurs: Grundkonzepte von XML

Wie in anderen (beschreibenden) Auszeichnungssprachen auch werden in XML Tags zur Kennzeichnung (Auszeichnung) von Textelementen verwendet. Dadurch ist es sowohl für den Menschen[18] lesbar als auch maschinell einfach weiterzuverarbeiten. Ein typisches XML –Dokument[19] ist in *Abb. 1.4/13* dargestellt. In diesem Beispiel werden Kundenauftragsdaten für einen Web-Shop mittels XML codiert, mit der Absicht, diese vom e-Mall Betreiber an den Händler zu übertragen.[20]

---

16 Vgl. http://java.sun.com/ (25. 1. 1999)

17 Vgl. http://developer.netscape.com/docs/manuals/javascript.html (25. 1. 1999)

18 Dieser Vorteil zeigt sich vor allem beim Programmieren von Schnittstellen, wo Programmierer und Analytiker oftmals Originaldateien positionsweise lesen müssen.

19 XML ist im Unterschied zu HTML case-sensitiv, d. h. es wird zwischen Groß- und Kleinschreibung unterschieden.

20 Yahoo bietet diese Möglichkeit auch für kleine Händler an und offeriert dazu auch die verschlüsselte Übertragung sensibler Daten mittels s-http sowohl im Pull als auch im Push-Verfahren. Details dazu unter: http://store.yahoo.com/

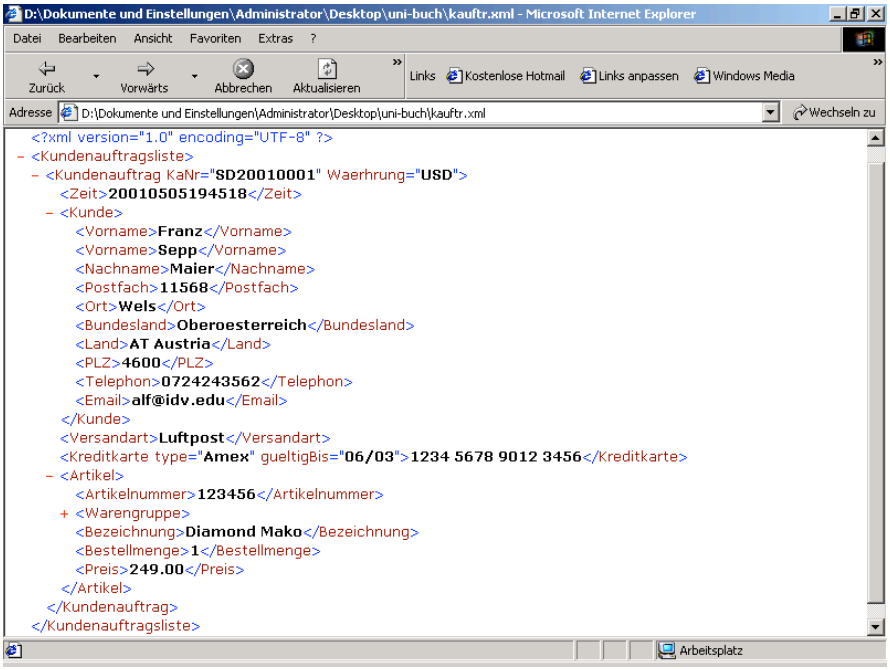

Abb. 1.4/13: Kundenauftragsdaten eines Webshops in XML dargestellt im MS Explorer 6.0

Der Quellcode des XML-Dokuments wird mit Hilfe eines Browsers[21] dargestellt. Die Darstellung unterscheidet sich von der Darstellung in einem Texteditor dadurch, dass geschachtelte Tags *aufgerissen* werden können[22]. Dies ist an den Minus- und Pluszeichen im Screenshot aus *Abb. 1.4/13* ersichtlich. Diese Zeichen sind nicht Bestandteil des Quellcodes, sondern werden durch den Internetclient hinzugefügt. Eine optisch ansprechendere Dokumentdarstellung in einem Internetbrowser kann man durch den Einsatz von Style-Sheets erreichen. In XML gibt es dafür die E**x**tensible **S**tylesheet **L**anguage (**XSL**), die gemeinsam mit der Stylesheet Erweiterung CSS von HTML verwendet werden kann.

Das Beispiel aus *Abb. 1.4/13* ist für den Menschen um einiges einfacher zu lesen als ein konventionelles Text- oder gar Binärformat. Weiters kann man nach Bedarf eigene Tags, wie beispielsweise <Preis> definieren. Dies bedeutet aber noch nicht, dass es dadurch auch automatisch in konkreten betrieblichen Anwendungsprogrammen weiterverarbeitet werden kann. Grund dafür sind die meist divergierenden Datenstrukturen zwischen den Zielformaten der betrieblichen Anwendungen und den Austauschformaten. Ein einfaches Beispiel dafür ist das Tag <Preis>, das in diesem Fall die Zeichenkette „249.00“ enthält. Einem Menschen unseres Kulturraumes ist die Semantik des Wortes „Preis“ bekannt, was aber für den Begriff einer „Warengruppe“ nicht mehr gelten muss. Für das Element Preis sind auch die möglichen Ausprägungen des Inhalts im Dokument intuitiv (positive Dezimalzahlen, zwei

21 Daran ist ersichtlich, dass XML-Dokumente im Unterschied zu HTML nicht primär auf eine Präsentation im Internet abzielen. XML zielt nicht darauf ab, HTML als Sprache des Internetpublishing abzulösen. Für die Präsentation im Internet werden XML Seiten auch in Zukunft in HTML (bzw. xHTML) transformiert werden.

22 Ein weiterer, in der Abbildung erkennbarer Unterschied zur Darstellung in einem Texteditor besteht darin, dass der Verweis auf die externe DTD nicht angezeigt, sondern dafür auf den Quelltext verwiesen wird.

Nachkommastellen) ersichtlich. Der Begriff der „Warengruppe“[23] kann dagegen nicht als allgemein bekannt vorausgesetzt werden und auch die Bezeichnung ist nicht einheitlich. So kann etwa das Synonym „Materialgruppe“ verwendet werden. Der mögliche Wertebereich für eine Warengruppe ist auch nicht per-se bekannt und es werden in den Firmen die verschiedensten Kategorisierungen verwendet. Genauso ist es notwendig, festzulegen, in welcher Reihenfolge die Informationen stehen dürfen, d. h. den Kommunikationspartnern muss die Struktur der auszutauschenden Dokumente bekannt sein. Es muss also ein gemeinsames Verständnis über die Semantik und die Struktur der verwendeten Dokumente geschaffen und in maschinen- und menschenlesbarer Form festgehalten werden: Dies ermöglicht die **D**ocument **T**ype **D**efinition (**DTD**).

Im Unterschied zu HTML und konventionellen Dateien[24] existiert zu einem XML-Dokument zusätzlich ein optionaler Teil, in dem definiert wird, welche Tags und Attribute wie und an welcher Stelle im Dokument verwendet werden können: Die *Document Type Definition* (DTD). Eine DTD kann sich direkt am Anfang des XML-Dokuments befinden. In diesem Fall spricht man von einer internen DTD. In der DTD stehen die formalen Regeln, die Syntax zur korrekten Gestaltung von XML-Dokumenten. Die Syntax wird darin mit Hilfe der EBNF[25] beschrieben. Mit deren Hilfe lassen sich die verwendeten Tags, die dazu erlaubten Attribute und auch die Reihenfolge bzw. erlaubte Verschachtelungen definieren. Neben der direkten Einbindung einer DTD in das XML-Dokument kann eine DTD auch getrennt als eigene Datei existieren. Die Verknüpfung zu einer externen DTD wird mit Hilfe des Tags <DOCTYPE>[26] am Beginn des Dokuments hergestellt. Die Einbindung des Verweises zu einer DTD für eine Kundenauftragsliste eines Yahoo Internetshops sieht beispielsweise so aus:

```
<!DOCTYPE OrderList SYSTEM "http://store.yahoo.com/lib/vw/OrderList.dtd">
```

Der Verweis erfolgt durch eine http-URL. Genauso gut könnte es sich auch um eine File-URL handeln, die auf eine Datei in einem LAN oder auf demselben Rechner verweist. *Abb.1.4/14* zeigt einen Ausschnitt, der zum XML-Dokument (*Abb. 1.4/13*) gehörenden externen DTD. In dieser DTD wird die Syntax der Kundenauftragsliste definiert.

So wird beispielsweise festgelegt, dass ein dazugehörendes XML-Dokument für eine Liste von Kundenaufträgen aus beliebig vielen Aufträgen bestehen kann. Ein einzelner Kundenauftrag besteht aus den Kundenstammdaten und beliebig vielen Auftragspositionen. Bei den Kundenstammdaten kann optional eine von der Rechnungsadresse abweichende Lieferadresse angegeben werden.

---

23 Welcher Warengruppe ein Kunde einen Artikel zuordnet, ist für den Lieferanten in der Regel nicht von Bedeutung und wird deshalb auch normalerweise in einem Kundenauftrag, wie im Lehrbeispiel, nicht angeführt.

24 Um in konventionellen Dateien Strukturen mit einer variablen Anzahl an Elementen abbilden zu können, verwendet man Satztypen zur Identifikation von Elementen.

25 Die Extended Backus Naur Form (EBNF) ist eine Erweiterung der Backus Naur Form (BNF), die von J. Backus und P. Naur zur Beschreibung der Programmiersprache Algol definiert wurde. Mithilfe der EBNF kann ganz allgemein die Syntax einer Sprache, also auch einer Auszeichnungssprache, beschrieben werden.

26 In Abb. I-2.8 steht dieses Tag am Beginn des HTML Dokuments. Genaugenommen handelt es sich um xHTML 1.0, das ein mithilfe von XML definiertes HTML 4.01 darstellt.

```
<!ELEMENT KundenauftragsListe[27] (Kundenauftrag*) >

<!--erstellt für didaktische Zwecke , T. Filsecker, 2001-->

<!ATTLIST KundenauftragsListe WebShop CDATA #REQUIRED>

<!ELEMENT KundenAuftrag (Zeit, Kunde, Versandart?, Kreditkarte?, Artikel+ )>

<!ATTLIST KundenAuftrag KaNr ID #REQUIRED Waehrung CDATA #REQUIRED >

<!ELEMENT Kunde (Vorname*, Familienname, ( Postfach | Strasse) , Ort, Bundesland,
Land?, PLZ, Telephon?, Email?, Kreditkarte)>

<!ELEMENT Land (#PCDATA)> [28]

<!-- Der Inhalt beginnt mit dem zweistelligen ISO-3166 code, gefolgt von einem

    Leerzeichen und dem ausgeschriebenen Laendernamen

    Beispiel: <Country>US United States</Country> -->

<!ELEMENT               Kreditkarte (#PCDATA)>

<!ATTLIST     Kreditkarte   type (Amex|Visa|MC) #REQUIRED

                                                gueltigBis CDATA #IMPLIED
```

Abb. 1.4/14: Ein Ausschnitt aus der DTD für eine Kundenauftragsliste

Für die DTD stehen zur Beschreibung der logischen Struktur die Schlüsselwörter[29,30] <!ELEMENT ... >, <!ATTLIST ... > zur Verfügung, mit denen eigene Tags und Attribute definiert werden können. Daneben können auch wie schon aus HTML bekannt, <!-- Kommentare --> eingefügt werden. Mittels <!ELEMENT ...> werden die sogenannten Elementtypen definiert, die im XML-Dokument dann als konkretes Element (Tags) in Erscheinung treten. Ein Elementtyp darf nur einmal definiert werden. Mit <!ATTLIST> wird die Attributliste eines Elementtyps deklariert.

Im dargestellten Beispiel wird für die Kundenstammdaten der **Elementtyp** *Kunde* definiert. In der DTD werden die Elementtypen und deren Verwendungsregeln definiert. Im XML-Dokument treten diese Elementtypen dann als konkrete Elemente in Form von Tags und deren umschlossene Inhalte auf. Mit Hilfe der DTD werden also nicht nur die im Dokument erlaubten Tags definiert, sondern auch deren mögliche Inhalte. Als Inhalt eines Elementes

---

27 Das erstdefinierte Element wird als Wurzel *(root)* bezeichnet.

28 Dieses Beispiel für das Element *Land* wurde bewusst aus der Yahoo DTD für Online-Shop Betreiber übernommen. Eleganter und vor allem richtig wäre es natürlich, diese Struktur auch in der DTD zu definieren, etwa durch zwei Elemente *CountryCode* und *CountryName* als Subelement von *Country*.

29 Man spricht von Schlüsselwörtern und nicht von Tags, auch wenn diese mit den Deliminatorzeichen so ähnlich aussehen. Grund dafür ist, dass die DTD selbst nicht mit einer Auszeichnungssprache, sondern mittels einer EBNF-ähnlichen Notation definiert wird.

30 Daneben gibt es noch zur Beschreibung der physischen Struktur die Schlüsselwörter <!ENTITY ... > und <!NOTATION ... >, die jedoch im Rahmen dieses propädeutischen Textes nicht weiter behandelt werden. Der interessierte Leser sei auf die XML Spezifikation verwiesen (siehe Fußnote 13).

kommt nicht nur reiner Text, sondern auch ein anderes Element oder eine Gruppe von Elementen in Betracht. Diese *Kindelemente* können ein- oder mehrfach vorkommen. In der Elementtypdefinition kann man mit Hilfe von *Sequenz*, *Alternative* und *Option* den Inhalt eines Elementes beschränken.

Am Beispiel aus Abb. 1.4/14 wird dies näher erläutert: Dem Kundennamen folgen in Klammern weitere Informationen, wie Adresse, Telefonnummer, E-Mail etc. Dabei handelt es sich um sogenannte Kindelemente von *Kunde,* für die es in der DTD ebenfalls eine Elementtypdeklaration gibt. Die als Inhalt von *Kunde* erlaubten Kindelemente werden durch Beistrich voneinander getrennt und ergeben so die Reihenfolge (Sequenz) ihrer Verwendung. Diese Reihenfolge gilt dann im XML-Dokument für die Verwendung der entsprechenden Tags im XML Dokument. So muss beispielsweise das Element *Namen* vor dem Element *Land* verwendet werden. Die Kindelemente werden als Elementtype definiert, wie an Hand der Elementtypdefinitionen für das Land und die Kreditkartennummer im Beispiel ersichtlich ist. Ein Fragezeichen hinter einem Element bedeutet, dass dieses optional ist *(Option),* wie z. B. die E-Mail Adresse. Im Beispiel der Adressangabe sind die Elemente für Straße und Postfach zusätzlich geklammert und voneinander durch einen vertikalen Strich „|“ getrennt. Diese Schreibweise steht für *Alternative* und bedeutet, dass genau eines der Elemente verwendet werden kann und muss. Man kann durch Hinzufügen eines Fragezeichens eine optionale Alternative daraus machen. Das bedeutet, dass genau ein Element aus der Aufzählung verwendet werden kann, aber nicht verwendet werden muss.

Eine andere Funktion erfüllt das Asterisk-Zeichen *, das für „beliebig viele“ Elemente steht. Im Beispiel kommt dieses Zeichen hinter den Elementen *Kundenauftrag* und *Vorname* vor. Dies bedeutet, dass das XML-Dokument aus beliebig vielen (auch null) Kundenaufträgen bestehen und ein Kunde beliebig viele Vornamen haben kann. Das Pluszeichen hinter dem Kindelement Artikel hat eine ähnliche Funktion. Es bewirkt in diesem Fall, dass ein Kundenauftrag beliebig viele Artikel umfassen kann, aber im Unterschied zum Asterisk muss mindestens ein Element enthalten sein. Dies sind Beispiele für die Strukturierung von Elementen, die wiederum aus Elementen bestehen. Im „Nutzdatenteil“ des XML-Dokuments erkennt man dies an den geschachtelten Tags, die sich aus dieser Elementdefinition ableiten. Bei nicht weiter verschachtelten Tags bzw. auf der untersten Ebene der Elementtypdefinition besteht der Inhalt eines Elementes aus einer Zeichenkette (String). Dafür existiert das Schlüsselwort PCDATA, das für einen beliebig langen String steht. Darüber hinaus kann eine Elementtypdefinition auch einen *gemischten* Inhalt zulassen, bei dem Elemente dieses Typs Zeichenketten enthalten können, die optional mit Kindelementen gemischt werden können. Ebenfalls für eine beliebige Zeichenkette steht das Schlüsselwort CDATA[31], allerdings wird dieses im Zusammenhang mit Attributen verwendet. Attribute sind Zusätze zu Elementen, die diese näher beschreiben. Im dargestellten Beispiel werden für das Element *Kreditkarte* die beiden Attribute *type* und *gueltigBis* definiert. Der Zusatz #REQUIRED zu *type* bedeutet, dass das Attribut erforderlich ist, während #IMPLIED besagt, dass das „Ablaufdatum“ der Kreditkarte nicht unbedingt erforderlich ist. Weiters unterscheiden sich die beiden Attribute in der Art der erlaubten Inhalte. Für das Attribut *type* wurde eine Aufzählung mit erlaubten

[31] PCDATA steht für *Parsed Chraracter Data*, das sind vom Parser zu analysierende und zu verarbeitende Zeichenketten, während CDATA direkt an die folgende Anwendung ohne Zwischenverarbeitung durch den Parser weitergereicht werden.

Werten angegeben, aus denen genau eine der angegebenen Zeichenketten ausgewählt werden darf. Im Gegensatz dazu steht das CDATA für einen beliebigen String. Ähnlich einem Identifikationsschlüssel in einer Datenbank wirkt bei einem Attribut der Zusatz ID. Damit erzwingt man, dass ein String nur einmal im gesamten XML-Dokument für das zugehörende Element verwendet werden darf. In der Attributliste zum Element *Kundenauftrag* wird ein Attribut *KaNr* [32] definiert und diesem der Typ *ID* zugewiesen. Dadurch wird erreicht, dass ein bestimmter Inhalt eines Elementes (z. B. der Kundenauftrag mit der Nummer SD20010001) im gesamten Dokument einzigartig ist und keine Kundenauftragsnummer mehrfach vergeben werden kann.

Einer der viel gepriesenen Vorteile von XML ist seine kurze und prägnante Definition und eine damit einhergehende leichte Erlernbarkeit; ganz im Gegensatz zum umfangreichen und schwer erlernbaren SGML. In der Definition von XML 1.0[33] „fehlen“ einige wichtige Konzepte, um es als allgemeine Integrationstechnologie auch für den Bereich der klassischen[34] betrieblichen Informationsverarbeitung zu nutzen. Dieser „Mangel“ wird durch Spracherweiterungen (Die Konzepte heißen *Namensräume*, *Verweise*, *Schemata* (bzw. Schemasprachen), und *Style Sheets)* Bei diesen Erweiterungen handelt es sich aber keineswegs um die nachträgliche Bereinigung von Versäumnissen oder Unzulänglichkeiten der Sprachdefinition. Man wollte die Definition möglichst kompakt halten und hatte schon von Anbeginn die Erweiterung um fundamentale Konzepte, wie beispielsweise die Namensräume, im Hinterkopf.

32 Als sprechende Abkürzung für Kundenauftragsnummer.

33 Vgl. http://www.w3.org/TR/2000/REC-xml-20001006 (28. 3. 2001)

34 Im Sinne einer transaktionsorientierten Informationsverarbeitungsinfrastruktur