

7 DOKUMENTE

7.1 DOKUMENTE

Die Idee zu dieser Einheit geht auf eine Vorlesung „Informatik A“ von Till Tantau zurück (siehe <http://www.itheoi.mu-luebeck.de/lehre/2008-ws/info-a/wiki/Vorlesung>, 10.11.08).

Im alltäglichen Leben hat man mit vielerlei Inschriften zu tun: Briefe, Kochrezepte, Zeitungsartikel, Vorlesungsskripte, Seiten im WWW, Emails, und so weiter. Bei vielem, was wir gerade aufgezählt haben, und bei vielem anderen kann man drei verschiedene Aspekte unterscheiden, die für den Leser eine Rolle spielen, nämlich

- den *Inhalt* des Textes,
- seine *Struktur* und
- sein *Erscheinungsbild*, die (äußere) *Form*.

Inhalt
Struktur
Erscheinungsbild,
Form

Stünde die obige Aufzählung so auf dem Papier:

- den INHALT des Textes,
- seine STRUKTUR und
- sein ERSCHEINUNGSBILD, die (äußere) FORM.

dann hätte man an der Form etwas geändert (Wörter in Großbuchstaben statt kursiv), aber nicht am Inhalt oder an der Struktur. Hätten wir dagegen geschrieben:

„[...] den *Inhalt* des Textes, seine *Struktur* und sein *Erscheinungsbild*, die (äußere) *Form*.“

dann wäre die Struktur eine andere (keine Liste mehr, sondern Fließtext), aber der Inhalt immer noch der gleiche. Und stünde hier etwas über

- *Balaenoptera musculus* (Blauwal),
- *Mesoplodon carlhubbsi* (Hubbs-Schnabelwal) und
- *Physeter macrocephalus* (Pottwal),

dann wäre offensichtlich der Inhalt ein anderer.

Von Ausnahmen (welchen?) abgesehen, ist Autoren und Lesern üblicherweise vor allem am Inhalt gelegen. Die Wahl einer bestimmten Struktur und Form hat primär zum Ziel, den Leser beim Verstehen des Inhalts zu unterstützen. Mit dem Begriff *Dokumente* in der Überschrift sollen Texte gemeint sein, bei denen man diese drei Aspekte unterscheiden kann.

Dokumente

Wenn Sie später beginnen, erste nicht mehr ganz kleine Dokumente (z. B. Seminararbeiten oder die Bachelor-Arbeit) selbst zu schreiben, werden Sie merken, dass die Struktur, oder genauer gesagt das Auffinden einer geeigneten Struktur, auch zu Ihrem, also des Autors, Verständnis beitragen kann. Deswegen ist es bei solchen Arbeiten unseres Erachtens sehr ratsam, *früh* damit zu *beginnen*, *etwas aufzuschreiben*, weil man so gezwungen wird, über die Struktur nachzudenken.

*ein Rat für Ihr
weiteres Studium*

Programme fallen übrigens auch in die Kategorie dessen, was wir hier Dokumenten nennen.

Oft ist es so, dass es Einschränkungen bei der erlaubten Struktur von Dokumenten gibt. Als Beispiel wollen wir uns zwei sogenannte *Auszeichnungssprachen* (im Englischen *markup language*) ansehen. Genauer gesagt werden wir uns dafür interessieren, wie Listen in L^AT_EX aufgeschrieben werden müssen, und wie Tabellen in XHTML.

Auszeichnungssprache
markup language

7.2.1 L^AT_EX

L^AT_EX, ausgesprochen „Latech“, ist (etwas ungenau, aber für unsere Belange ausreichend formuliert) eine Erweiterung eines Textsatz-Programmes (iniT_EX), das von Donald Knuth entwickelt wurde (<http://www-cs-faculty.stanford.edu/~knuth/>, 12.11.08).

Es wird zum Beispiel in der Informatik sehr häufig für die Verfassung von wissenschaftlichen Arbeiten verwendet, weil unter anderem der Textsatz mathematischer Formeln durch T_EX von hervorragender Qualität ist, ohne dass man dafür viel tun muss. Sie ist deutlich besser als alles, was der Autor dieser Zeilen jemals in Dokumenten gesehen hat, die mit Programmen wie z. B. openoffice.org o. ä. verfasst wurden. Zum Beispiel liefert der Text

```
\[ 2 - \sum_{i=0}^k i 2^{-i} = (k+2) 2^{-k} \]
```

die Ausgabe

$$2 - \sum_{i=0}^k i 2^{-i} = (k + 2) 2^{-k}$$

Auch der vorliegende Text wurde mit L^AT_EX gemacht. Für den Anfang dieses Abschnittes wurde z. B. geschrieben:

```
\subsection{Struktur von Dokumenten}
```

woraus L^AT_EX die Zeile

```
7.2 STRUKTUR VON DOKUMENTEN
```

am Anfang dieser Seite gemacht hat. Vor dem Text der Überschrift wurde also automatisch die passende Nummer eingefügt und der Text wurde in einer Kapitälchenschrift gesetzt. Man beachte, dass zB die Auswahl Schrift *nicht* in der Eingabe mit vermerkt ist. Diese Angabe findet sich an anderer Stelle, und zwar an *einer* Stelle, an der das typografische Aussehen *aller* Abschnittsüberschriften (einheitlich) festgelegt ist.

Ganz grob kann ein L^AT_EX-Dokument z. B. die folgende Struktur haben:

```
\documentclass[11pt]{report}
% so schreibt man Kommentare
% dieser Teil heißt Präambel des Dokumentes
% Unterstützung für Deutsch,
% z.B. richtige automatische Trennung
\usepackage[german]{babel}
% Angabe des Zeichensatzes, in dem der Text ist
\usepackage[latin1]{inputenc}
% für das Einbinden von Grafiken
```

```

\usepackage{graphicx}
\begin{document}
  % und hier kommt der eigentliche Text .....
\end{document}

```

Eine Liste einfacher Punkte sieht in \LaTeX so aus:

Eingabe	Ausgabe
<code>\begin{itemize}</code>	
<code>\item Inhalt</code>	• Inhalt
<code>\item Struktur</code>	• Struktur
<code>\item Form</code>	• Form
<code>\end{itemize}</code>	

Vor den aufgezählten Wörtern steht jeweils ein dicker Punkt. Auch dieser Aspekt der äußeren Form ist *nicht* dort festgelegt, wo die Liste steht, sondern an *einer* Stelle, an der das typografische Aussehen *aller* solcher Listen (einheitlich) festgelegt ist. Wenn man in seinen Listen lieber alle Aufzählungspunkte mit einem sogenannten Spiegelstrich „–“ beginnen lassen möchte, dann muss man nur an einer Stelle (in der Präambel) die Definition von `\item` ändern.

Wollte man die formale Sprache L_{itemize} aller legalen Texte für Listen in LaTeX aufschreiben, dann könnte man z. B. zunächst geeignet die formale Sprache L_{item} aller Texte spezifizieren, die hinter einem Aufzählungspunkt vorkommen dürfen. Dann wäre $L_{\text{itemize}} = \{\text{\begin{itemize}}\} \left(\{\text{\item}\}_{L_{\text{item}}} \right)^+ \{\text{\end{itemize}}\}$. Dabei haben wir jetzt vereinfachend so getan, als wäre es kein Problem, L_{item} zu definieren. Tatsächlich ist das aber zumindest auf den ersten Blick eines, denn ein Aufzählungspunkt in \LaTeX darf seinerseits wieder eine Liste enthalten. Bei naivem Vorgehen würde man also genau umgekehrt für die Definition von L_{item} auch auf L_{itemize} zurückgreifen (wollen). Wir diskutieren das ein kleines bisschen ausführlicher in Unterabschnitt 7.2.3.

7.2.2 HTML und XHTML

HTML ist die Auszeichnungssprache, die man benutzt, wenn man eine WWW-Seite (be)schreibt. Für HTML ist formaler als für \LaTeX festgelegt, wie syntaktisch korrekte solche Seiten aussehen. Das geschieht in einer sogenannten *document type definition*, kurz *DTD*.

Hier ist ein Auszug aus der DTD für eine Version von XHTML. Sie dürfen sich vorstellen, dass das ist im wesentlichen eine noch striktere Variante von HTML ist. Das nachfolgenden Fragment beschreibt teilweise, wie man syntaktisch korrekt eine Tabelle notiert.

```

<!ELEMENT table (caption?, thead?, tfoot?, (tbody|tr+))>
<!ELEMENT caption %Inline;>
<!ELEMENT thead (tr)+>
<!ELEMENT tfoot (tr)+>
<!ELEMENT tbody (tr)+>

```

```

<!ELEMENT tr      (th|td)+>
<!ELEMENT th      %Flow;>
<!ELEMENT td      %Flow;>

```

Wir können hier natürlich nicht auf Details eingehen. Einige einfache aber wichtige Aspekte sind aber mit unserem Wissen schon verstehbar. Die Wörter wie `table`, `thead`, `tr`, usw. dürfen wir als bequem notierte Namen für formale Sprachen auffassen. Welche, das wollen wir nun klären.

Die Bedeutung von `*` und `+` ist genau das, was wir als Konkatenationsabschluss und ε -freien Konkatenationsabschluss kennen gelernt haben. Die Bedeutung des Kommas `,` in der ersten Zeile ist die des Produktes formaler Sprachen. Die Bedeutung des senkrechten Striches `|` in der sechsten Zeile ist Vereinigung von Mengen.

Das Fragezeichen ist uns neu, hat aber eine ganz banale Bedeutung: In der uns geläufigen Notation würden wir definieren: $L^? = L^0 \cup L^1 = \{\varepsilon\} \cup L$. Mit anderen Worten: Wenn irgendwo $L^?$ notiert ist, dann kann an dieser Stelle ein Wort aus L stehen, oder es kann fehlen. Das Auftreten eines Wortes aus L ist also sozusagen optional.

Nun können Sie zur Kenntnis nehmen, dass z. B. die Schreibweise `<!ELEMENT tbody (tr)+ >` die folgende formale Sprache festlegt:

$$L_{tbody} = \{<tbody>\} \cdot L_{tr}^+ \cdot \{</tbody>\}$$

Das heißt, ein Tabellenrumpf (*table body*) beginnt mit der Zeichenfolge `<tbody>`, endet mit der Zeichenfolge `</tbody>`, und enthält dazwischen eine beliebige positive Anzahl von Tabellenzeilen (*table rows*). Und die erste Zeile aus der DTD besagt

$$L_{table} = \{<table>\} \cdot L_{caption}^? \cdot L_{thead}^? \cdot L_{tfoot}^? \cdot L_{tbody}^+ \cdot \{</table>\}$$

das heißt, eine Tabelle ist (*table*) von den Zeichenketten `<table>` und `</table>` umschlossen und enthält innerhalb in dieser Reihenfolge

- optional eine Überschrift (*caption*),
- optional einen Tabellenkopf (*table head*),
- optional einen Tabellenfuß (*table foot*) und
- eine beliebige positive Anzahl von Tabellenrümpfen (siehe oben).

Insgesamt ergibt sich, dass zum Beispiel

```

<table>
  <tbody>
    <tr> <td>1</td> <td>a</td> </tr>
    <tr> <td>2</td> <td>b</td> </tr>
  </tbody>
</table>

```

eine syntaktisch korrekte Tabelle.

In Wirklichkeit gibt es noch zusätzliche Aspekte, die das Ganze formal verkomplizieren und bei der Benutzung flexibler machen, aber das ganz Wesentliche haben wir damit jedenfalls an einem Beispiel beleuchtet.

7.2.3 Eine Grenze unserer bisherigen Vorgehensweise

Dass wir eben mit Hilfe von Produkt und Konkatenationsabschluss formaler Sprachen in einigen präzise Aussagen machen konnten, hing auch der Einfachheit dessen, was es zu spezifizieren galt. Es wurde, jedenfalls in einem intuitiven Sinne, immer etwas von einer komplizierteren Art aus Bestandteilen einfacher Art zusammengesetzt.

Es gibt aber auch den Fall, dass man sozusagen größere Dinge einer Art aus kleineren Bestandteilen zusammensetzen will, die aber von der gleichen Art sind. Auf Listen, deren Aufzählungspunkte ihrerseits wieder Listen enthalten dürfen, hatten wir im Zusammenhang mit \LaTeX schon hingewiesen.

Ein anderes typisches Beispiel sind korrekt geklammerte arithmetische Ausdrücke. Sehen wir einmal von den Operanden und Operatoren ab und konzentrieren uns auf die reinen Klammern. Bei einer syntaktisch korrekten Klammerung gibt es zu jeder Klammer auf „weiter hinten“ die „zugehörige“ Klammer zu. Insbesondere gilt:

- Man kann beliebig viele korrekte Klammern konkatenieren und erhält wieder eine korrekte Klammerung.
- Man kann um eine korrekte Klammerung außen herum noch ein Klammerpaar schreiben (Klammer auf ganz vorne, Klammer zu ganz hinten) und erhält wieder eine korrekte Klammerung.

Man würde also gerne in irgendeinem Sinne L_{Klammer} mit L_{Klammer}^* und mit $\{ \{ \} \cdot L_{\text{Klammer}} \cdot \{ \} \}$ in Beziehung setzen. Es ist aber nicht klar wie. Insbesondere würden bei dem Versuch, eine Art Gleichung hinzuschreiben, sofort die Fragen im Raum stehen, ob die Gleichung überhaupt lösbar ist, und wenn ja, ob die Lösung eindeutig ist.

7.3 ZUSAMMENFASSUNG

In dieser Einheit haben wir über *Dokumente* gesprochen. Sie haben einen *Inhalt*, eine *Struktur* und ein *Erscheinungsbild*.

Formale Sprachen kann man z. B. benutzen, um zu spezifizieren, welche Struktur(en) ein legales, d. h. syntaktisch korrektes Dokument haben darf, sofern die Strukturen hinreichen einfach sind. Was man in komplizierten Fällen machen kann, werden wir in einer späteren Einheit kennenlernen.