Dokumentensatz: TEX & LATEX

Patrick Pekczynski

Proseminar: Programmiersysteme WS 2003/04
Dozent: Gert Smolka, Prof. Dr. rer. nat
Betreuer: Marco Kuhlmann, MSc
FR 6.2 Informatik
Lehrstuhl für Programmiersysteme
Universität des Saarlandes, Saarbrücken
31. März 2004 - 02. April 2004
Gesetzt mit LATEX2ϵ <2001/06/01>

Gliederung

TEX - Etymologie

Bedeutung & Aussprache

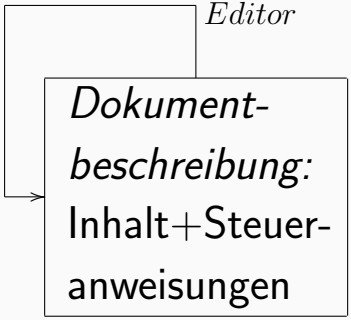
- $\tau\epsilon\chi$ (griech.: *Kunst, Technologie*) [Knu84]
- 'X' steht für 'χ' $\Rightarrow$
  - schottisch: *'Loch'*
  - deutsch: *'ach'*
  - spanisch: *'Juan'*
  - russisch: *'kh'*
- TEX (gesprochen *'Tech'*) [Kop00]

Was ist TEX ? - (1)

- **Dokumentensatzsystem** [Kop00]
  - „TEX [is] a new typesetting system, intended for the creation of beautiful books [...] that contain a lot of mathematics.“[Knu84]
  - nahezu **alle** Aufgaben lösbar, die dem traditionellen Beruf des Setzers entsprechen
  - typographische Qualität mit den weltbesten Setzern zu vergleichen
  - Erzeugung **qualitativ hochwertiger, technischer Manuskripte**

Was ist TEX ? - (2)

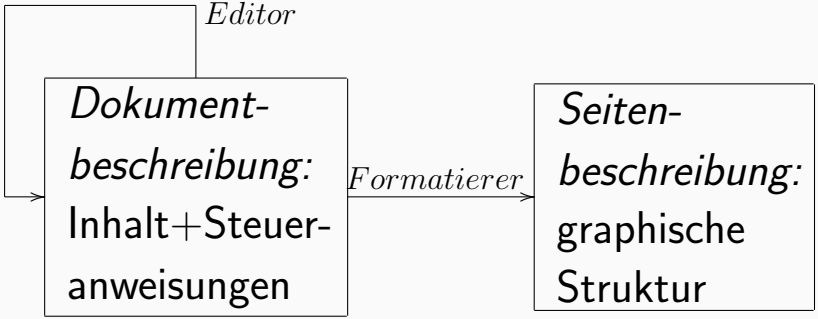
- **physische Markup-Sprache** [1]
  - **Kein** interaktives *WYSIWYG*-System auf der Basis eines *Wortprozessors*
  - **3-stufiges** *Batch-System* auf Basis von:
    1. Editor
    2. Formatierer
    3. Visualisierer



[1] visuelles Markup, Präsentationsmarkup

Was ist TEX ? - (2)

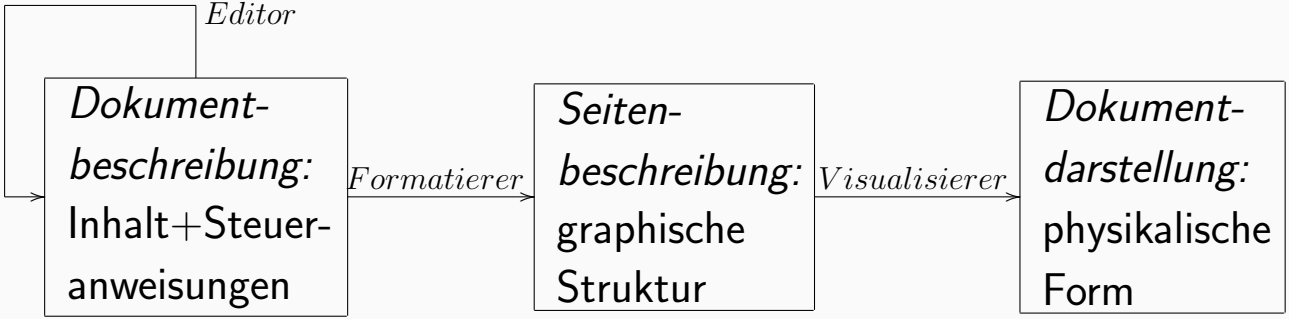
- **physische Markup-Sprache** [1]
  - **Kein** interaktives *WYSIWYG*-System auf der Basis eines *Wortprozessors*
  - **3-stufiges** *Batch-System* auf Basis von:
    1. Editor
    2. Formatierer
    3. Visualisierer



[1] visuelles Markup, Präsentationsmarkup

Was ist TEX ? - (2)

- **physische Markup-Sprache** [1]
  - **Kein** interaktives *WYSIWYG*-System auf der Basis eines *Wortprozessors*
  - **3-stufiges** *Batch-System* auf Basis von:
    1. Editor
    2. Formatierer
    3. Visualisierer



[1] visuelles Markup, Präsentationsmarkup

Was ist TEX ? - (3)

- **Makroprogrammmiersprache**

  Mächtigkeit von TEX als Programmiersprache basiert auf

  – Verwendung von **Makros**

  – Prozess der **Makroexpansion**

- **Minimalistischer Ansatz**

  – Es existieren zahlreiche Makropakete, Tools, Style-Files oder zusäztliche Fonts für TEX

TEX - Entwicklungsumgebung

- erste Version von TEX [TEX78]:

  SAIL [*Stanford Artificial Intelligence Language*]

  [**ALGOL 60** ähnlich]

- spätere Versionen von TEX : **WEB** [2]
  - von Donald Knuth entwickelt als **literate programming**
  - an TEX und **Pascal** gebundene Metasprache
  - Untermenge von **Pascal**, erweitert um Präprozessor [Eij92]
  - in Quellcode integrierte Dokumentation

    [*effizientes Warten von Code*]

[2] CWEB für C, noweb für jede Programmiersprache

TEX-Varianten/Erweiterungen

- ϵ-TEX [`etex`]

  TEX , das „rechts-nach-links “- Satz unterstützt

  $\Rightarrow$ arabische Sprachen

- pdfTEX [`pdftex`]

  TEX , das als Ausgabe direkt **PDF** erzeugt statt **DVI**

- Omega [`omega`]

  – Interne Benutzung von 16-Bit-Unicode-Zeichen

  – Dokumentensatz in allen nahezu allen Sprachen
  [*Arabisch, Chinesisch, Sanskrit, Griechisch, . . .*]

Was ist LaTeX ? - (1)

LaTeX *[kurzw.: Lamport TeX]* ist:

- das bekannteste Makropaket für TeX (S. 30)
- komfortables TeX-Zugangspaket
- Abstraktion der TeX-Befehle auf **high-level** Befehle
- benutzerfreundlichere Oberfläche mit Fokus auf Dokumentstruktur
- sowohl **generisches**, als auch **logisches** Markup [3] durch Kontrollbefehle der Form \<*control sequence*>

[3] deskriptives Markup

Was ist LaTeX ? - (2)

Logische Markup-Sprache:

- Stärkere Trennung von Inhalt [*logische Struktur*] und Layout [*graphische Repräsentation*]

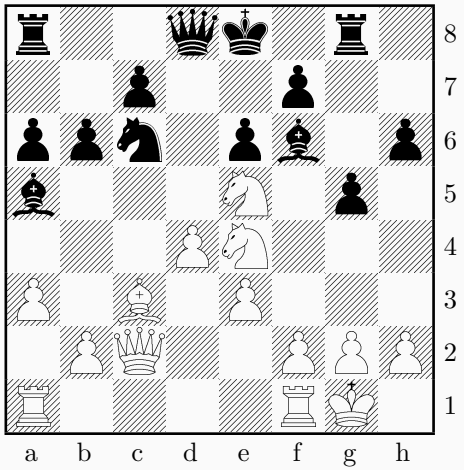
  zentrale Änderungen für das gesamte Dokument möglich
- Durch Ähnlichkeiten mit anderen Markupsprachen Konvertierung möglich, z.B. Latex $\rightarrow$ HTML
- Hervorhebung der logischen Struktur durch Befehle wie z.B.:

  `\section{`<*Abschnitt*>`}` oder `{\em` <*text* >`}`
- Klammerung ausgewählter Teile zu Umgebungen vom Typ T `\begin{T}...\end{T}` und Überprüfung der korrekten hierarchischen Klammerung

Was ist LaTeX ? - (3)

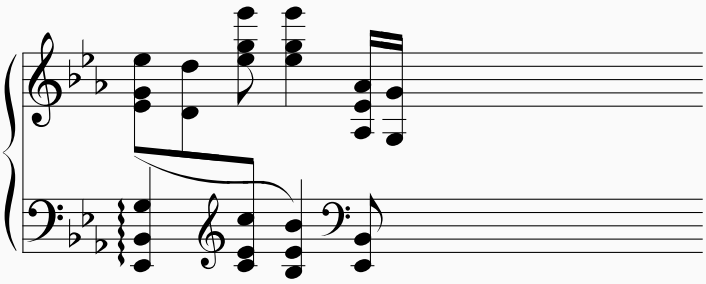
LaTeX ist Grundlage für zahlreiche Ergänzungspakete wie z.B.:

- Schachdokumentation mit *chess*



Was ist LaTeX ? - (4)

- Notensatz mit *musixtex*



ConTEXt

- Makropaket, das auf plain-TEX aufbaut
- vollwertige Alternative zu LATEX
- einheitliches Konzept für Makros und Schnittstellen
- Erstellen hochwertiger [*auch* **interaktiver**] PDF-Dokumente
- METAPOST-Code kann direkt verarbeitet werden
- **XML-Code** kann interpretiert und dargestellt werden [*integrierter XML-Parser*]
- arbeitet mit **PERL**-Skripten wie *TEXUTIL* [4] und *TEXEXEC* [5]

[4] Hilfsprogramm *[Register, Listen, Referenzen ...]*
[5] ConTEXt und PDF-Hilfsprogramm und *Batch-Prozessor*

Gliederung







Der TEX-Prozessor

Die Verarbeitung von TEX-Dateien erfolgt auf 4 Ebenen:

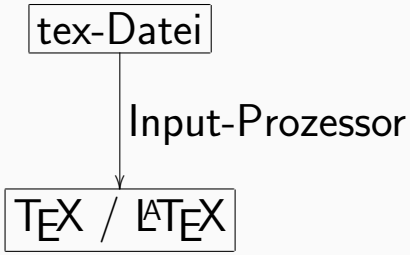
1. Input-Prozessor
2. Expansions-Prozessor
3. Ausführungs-Prozessor
4. Visueller Prozessor



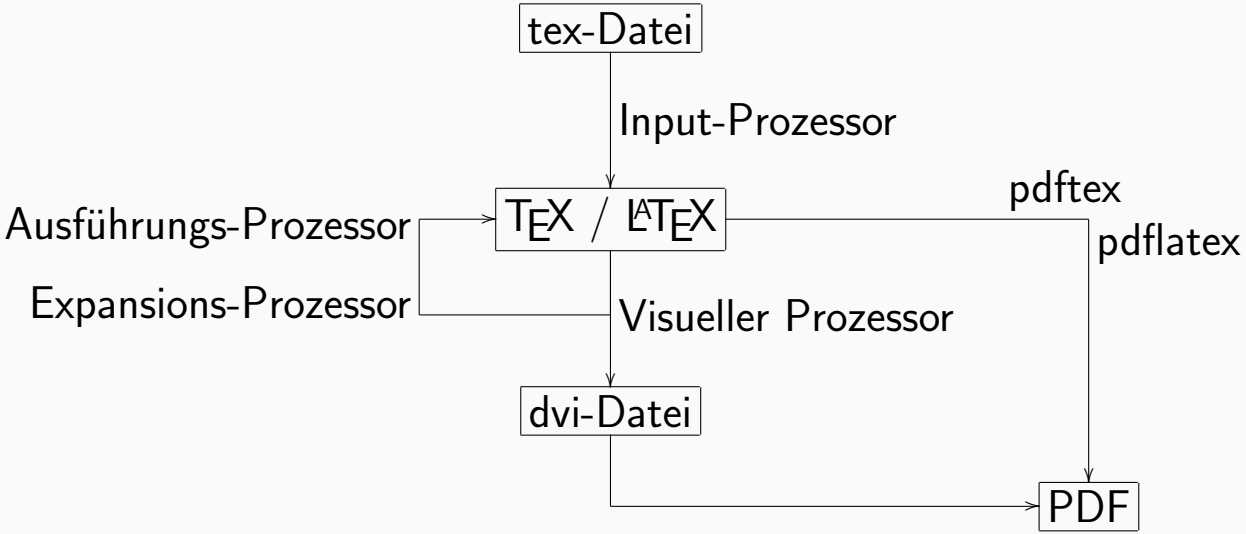




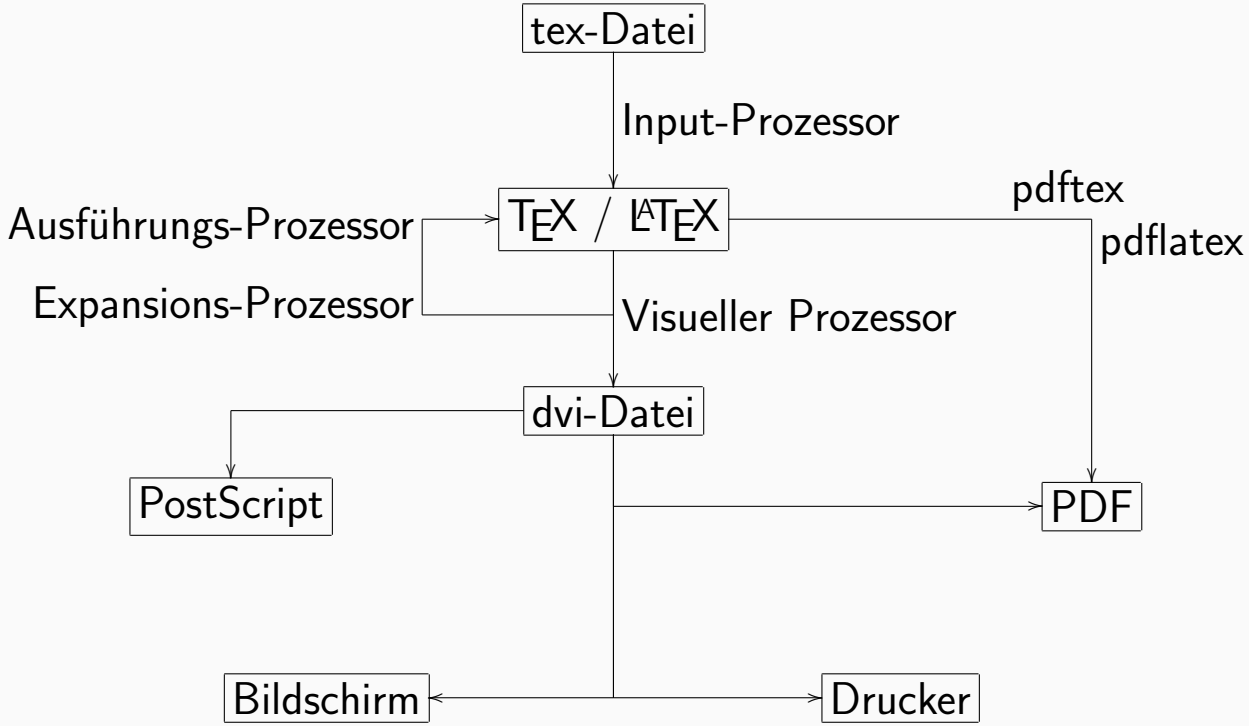
Skizze - TEX-Prozess



Skizze - TEX-Prozess



Skizze - TEX-Prozess



Der Input-Prozessor - (1)

- zeilenweises Übersetzen der Zeichen aus der Eingabedatei in Token [= *interne Objekte von TEX*]
- Ausgabe ist ein Stream aus Token, genauer eine Liste von Token
- 2 Kategorien von Token:
  1. Token für den Satz [*character tokens*]
     Einteilung der Satz-Token in 16 Kategorien (S. 30)
  2. Befehle die weiter verarbeitet werden [*control sequence tokens*]

Der Expansions-Prozessor - (2)

- Erhält als Eingabe Token des Input-Prozessor
- Expansion des Eingabestroms aus Token bis ein Token gefunden wird, das nicht weiter expandiert werden kann
- wesentlicher Vorgang = Prozess der **Makroexpansion**
- durch spezielle Befehle kann die Expansionsreihenfolge verändert werden:
  - `\expandafter`<*token1*><*token2*>
    $\rightsquigarrow$ <*token1*>< *Expansion von token2*>
  - `\noexpand`<token> nächstes Token nicht expandierbar

Der Ausführungs-Prozessor - (3)

- erhält nicht weiter expandierbare Token aus dem Expansions-Prozessor als Eingabe:
  - Token, die eine Zuweisung bewirken [*inklusive Makrodefinitionen*]
  - Token für horizontale, vertikale oder mathematische Listen:
    - Zeichen [*characters*] der Kategorien 11 und 12
    - Boxen [*boxes*] 
    - Kleber [*glue*]
- besondere Anweisung: `\relax` $\rightsquigarrow$ keine Ausführung

Der Visuelle Prozessor - (4)

Im Visuellen Prozessor werden auf erstellte Listen folgende Verfahren angewendet:

- Umbruch von Absätzen
- Ausrichtung
- Seitenumbruch
- mathematischer Textsatz
  [*reimplementiert in* **SML** *von Reinhold Heckmann* [WH97]]
- Erzeugen der DVI-Ausgabedatei [**D**e**v**ice **I**ndependent file]

Gliederung

- ☞ **Programmiersprachliche Aspekte**
- ☞ **TEXniken**
- ☞ **Probleme**
- ☞ **Ausblick**

Makroprogrammierung - (1)

TEX ist eine **makrobasierte** Sprache. Zentrale Konzepte sind:

- **Makros**:
  - TEX's definitorische Abkürzung von Befehlssequenzen
  - Kontrollsequenz, in der mehrere Befehle abkürzend zusammengefasst werden
  - Makros sind **parametrisierbar**
  - **Makros $\approx$ Prozeduren**

Beispiel

```
\def\abc#1{defg [[#1]]}
  \abc X ⇝ defg [[X]]
```

Makroprogrammierung - (2)

- **Makroexpansion**:
  - übernimmt zentrale Rolle bei der Verarbeitung des Dokumentes
  - trotz Verschiedenheit der Konzepte ist **Expansion** in TEX am ehesten eine Analogie für **prozedurale Aufrufe** in anderen Programmiersprachen







Makroprogrammierung - (3)

TEX = **Turing-vollständige** [6] Sprache

If-then-else

`\if` <*condition*><*then-branch* > `\else` <*else-branch*> `\fi`

loop

```
\def\loop#1\repeat{\def\body{#1}\iterate}
\def\iterate{
  \let\next\relax \body \let\next\iterate \fi\next}
```

`\body`$\rightsquigarrow$ `\if`<*abort*><*then-branch* > `\else`<*iteration*> `\repeat`

[6] Wikipedia-http://en.wikipedia.org/wiki/TeX

Gliederung

- ☞ TEXniken
- ☞ Probleme
- ☞ Ausblick







Falten von Listen

SML-Definition von *foldl*:

```
fun foldl f s nil = s
  |  foldl f s (x,xr) = foldl f (f(x,s)) xr
```

Naive **TEX**-Defintion von *foldla* für Listen über integers : [7]

```
\def\foldla(#1//#2//#3){
    \hd(#3(as)\x) \tl(#3(as)\xr)
    \ifx\nil\x
    \else
        #1(\x,#2)
        \count250=\func \countdef\buffer=250
        \foldla(#1//\number\buffer//\xr)
    \fi}
```

[7] *foldl*-Implementierung s. Ausarbeitung

Beispiele

- **Neue Liste definieren:**
  `\newlist(\examp=4,2,23,8,10,50,N)`
- **Reversieren mit foldle [Listen über Token]:**
  `\rev(\examp)`= [50,10,8,23,2,4]
- **Summe aller Elemente:**
  `\foldla(\sum//0//\examp)`=97
- **Dezimaldarstellung:**
  `\foldla(\litonum//0//\examp)`=422381050
- **Map modulo 3:**
  `\map(\mapmod3//\examp)`= [1,2,2,2,1,2]
- **Floats über Listen ($\frac{833719}{265381} \approx \pi$):**
  [3,1,4,1,5,9,2,6,5,3,5,8,1,0,7,7,7,7,1,2,0,4,4,1,9,3,0,6,5,8,1,8,5,7,7,8,1]

Gliederung

- ☞ **Probleme**
- ☞ **Ausblick**

Probleme? - (1)

- **umfangreicher Befehlssatz:**
  - TEX selbst basiert auf 300 Basisbefehlen + 600 daraus abgeleiteten Makrobefehlen [Kop00]
- Trotz hoher Qualität, Stabilität und Portabilität der Software

  $\Rightarrow$ **komplizierte Benutzung**
- **Minimalisitischer Ansatz** [*Segen und Fluch*]:
  - Viele verschiedene Stylefiles
  - Viele effektive aber noch nicht standardisierte Makropakete

Probleme? - (2)

- **Robustheit:**
  - **Ungetypte Sprache:**
    Makroaufruf mit Parametern möglich, die nicht mit Makrodefinition beabsichtigt wurden
  - Fehlersuche sehr umständlich, **Tracing** lediglich über Extraschalter möglich wie `\tracingonline` [Eij92]
  - Kontrollfluss nur schwer nachvollziehbar
- äußerst komplexe Satz-Algorithmen (S. 18)
  mit **Seiteneffekten** [WH97]

Gliederung

LaTeX3-Projekt

- Langzeit-Forschungsprojekt
- **Ziel** = die nächste Version von LaTeX zu entwickeln [GMS00] [Kop00]
- bis zur Fertigstellung von LaTeX3 ist die Projektgruppe mit der Wartung der aktuellen Version LaTeX2$_\epsilon$ beauftragt

Ziele von LaTeX3

- Verarbeitung eines größeren Umfangs an Dokumenten
  [*z.B:* **SGML**]
- Erweiterung der Dokumentklassen
- automatisierte Übersetzung von **SGML DTD's**
  in LaTeX-Dokumentklassen
- Interface für Style-files überarbeiten und vereinheitlichen
- Robustheit erhöhen [*Fehlersuche und fehlende Tags*]
- Voraussetzungen schaffen für **Hypertextsysteme**
  unter Verwendung von:
  - HTML
  - XML

$\Rightarrow$ Reimplementierung von größeren Programmteilen

LaTeX3-Mitglieder

Aktive Mitlieder	
Johannes	Braams
David	Carlisle
Michael	Downes
Robin	Fairbairns
Frank	Mittelbach
Chris	Rowley
Rainer	Schöpf
Martin	Schröder

Ehemalige Mitlieder	
Denys	Duchier
Alan	Jeffrey

- Wissenschaftliche Arbeitsweise:
  - Vorträge
  - Tagungen
  - Anwenderkonferenzen

Literatur

[Dow02] Michael Downes. *TEX and LATEX2ε*. *Notices of the AMS*, 49(11):1384–1391, 2002. [Download: pdf-file].

[Eij92] Victor Eijkhout. *TEX by Topic, A TEXnician's Reference*. Addison-Wesley, 1992. [Download: pdf-file].

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LATEX Companion*. Addison-Wesley, 1994.

[GMS00] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *Der LATEX Begleiter*. Addison-Wesley, 2000.

[Knu84] Donald E. Knuth. *The TEXbook*, volume A of *Computers and Typesetting*. Addison-Wesley, 1984.

[Kop00] Helmut Kopka. *LATEX - Einführung*, volume 1 of *LATEX*. Addison-Wesley, 2000.

[WH96] Reinhard Wilhelm and Reinhold Heckmann. *Grundlagen der Dokumentenverarbeitung*. Addison-Wesley, 1996.

[WH97] Reinhard Wilhelm and Reinhold Heckmann. *A Functional Description of TEX's Formula Layout*. *Journal of Functional Programming*, 7(5):451–485, 1997. [Download: ps-file].

Vielen Dank für die Aufmerksamkeit

Zusatz-Folien

- Entwicklungsgeschichte TEX
- Entwicklungsgeschichte LATEX
- Übersicht von Makros und Tools
- Übersicht Kategorien der Satz-Token
- Beispiel für Currying in TEX
- Liste TEX-Distributionen
- Fragen







Entwicklungsgeschichte - TEX

1977 **Donald Knuth**, Informatikprofessor an der
Stanford University, beginnt mit der Entwicklung von TEX

1978 erste brauchbare Ergebnisse mit TEX78 = *alpha version* [8]

1982 erste stabile Version mit TEX82 = *beta version*
Knuth erklärt TEX für PD-Software [*public domain*]
für nahezu jeden Rechnertyp und jedes Betriebssystem verfügbar

1983 TEX 1.0 erste TEX-Version wie es heute bekannt ist

1985 TEX 2.0 wird veröffentlicht

1990 TEX 3.0 letzte Hauptversion von TEX wird veröffentlicht

dato aktuelle TEX-Version = TEX 3.14159 [9]

Zurück zur Übersicht

[8] nach [Dow02],[Kop00]
[9] Die Versionsnummer strebt gegen π

Entwicklungsgeschichte - LaTeX

1985 **Leslie Lamport** stellt LaTeX 2.09 fertig

durch den Einsatz verschiedener Formatdateien [*Style-files*] geht Grundeigenschaft der **Rechnerunabhängigkeit** verloren

1989 **Mittelbach**, **Rowley** und **Schöpf** beginnen nach Begegnung mit Lamport mit Implementierung und Erweiterung von LaTeX

$\Rightarrow$ Beginn des **LaTeX3 Projekts**

1994 LaTeX2ϵ offizielle LaTeX-Version

1994-1999 zweimal im Jahr [*01.06. und 01.12.*] erscheint neue LaTeX-Version

2001 Seit 01.06. aktuelle LaTeX2ϵ Version [10]

Zurück zur Übersicht

[10] gem. http://www.latex-project.org/latex2e.html - Stand 27.01.2003

Flexibilität und Mächtigkeit

- Es existieren zahlreiche Makropakete, Tools, Style-Files oder zusätzliche Fonts :

`latex`	bekanntes Makropaket für TEX
`context`	Alternative zu LATEX
`javatex`	Javaimplementierung von TEX
`xmltex`	Parser für XML-Dokumente
`bibtex`	automatische Bibliographie erzeugen
`makeidx`	Indexeinträge
`makeindex`	automatischer Dokumentindex
`tex4ht`	LATEX- HTML/XML - Konverter

Zurück zur Übersicht

16 Kategorien für Zeichen (0-7)

0. Escape-zeichen: Beginn eines Steuerbefehls [\]
1. Beginn einer Umgebung: [{]
2. Ende einer Umgebung [}]
3. Math shift: Formelumgebung [$]
4. Zeilen-, spaltentabulator: [&]
5. Zeilenende: [*code 13 wie* `\endlinechar` *Parameter*]
6. Parameterzeichen für Makros: [#]
7. Superscript in Mathemodus: [^]

Zurück zur Übersicht

16 Kategorien für Zeichen (8-15)

8. Subscript in Mathemodus: [_]
9. Ignorierte Zeichen: werden von der Eingabe entfernt
10. Leerzeichen:
11. Buchstabe:
12. Andere: [*Zeichen, die in keiner anderen Kategorie sind, wie z.B. Zahlen oder Satzzeichen*]
13. Aktive Zeichen: fungieren als TEX-Befehle ohne ein vorangehendes Escape-zeichen
14. Kommentarzeichen: [%]
15. Ungültiges Zeichen: sollte nicht im Input auftreten

Zurück zur Übersicht

Currying (partielle Applikation)

```
\def\mapmul#1#2{
    \count11=#1
    \multiply\count11 by #2
    \def\func{\number\count11} }

\def\foo(#1,#2){#1#2}

\def\mul#1{\foo(\mapmul{12},#1)}
```

$\rightsquigarrow$ Aufruf von `\mul4` ergibt 48

Zurück zur Übersicht

TEX Distributionen

- MikTEX: http://www.miktex.org
- TEXLive: http://www.dante.de/software/cdrom/texlive
- OzTEX: http://www.trevorrow.com/oztex/
- teTEX: http://www.tug.org/teTeX

Zurück zur Übersicht

Fragen?

Zurück zur Übersicht