# Die TEXnische Komödie

DANTE
Deutschsprachige
Anwendervereinigung TEX e.V.

8. Jahrgang Heft 2/1996 August 1996

2/96

# Impressum



Diese Ausgabe wurde mit Hilfe folgender Programme fertiggestellt: `emTeX (tex386), Version 3.14159 [4b]`, `LaTeX2e <1995/12/01> patch level 1`, `GSview 1.3` (für die Bildschirmdarstellung) und `dvipsk 5.58f` (für Korrektur und endgültige Belichtung).

Erscheinungsweise: vierteljährlich
Erscheinungsort: Heidelberg
Auflage: 2500

Herausgeber: DANTE, Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 10 18 40
69008 Heidelberg
Tel.: 0 62 21/2 97 66
Fax: 0 62 21/16 79 06
e-mail: `dante@dante.de`

Druck: PrintArt GmbH
Kirchenstr. 8
67125 Dannstadt



Redaktionsschluß für Heft 3/1996: 30.9.1996

# Editorial

Liebe Leserinnen und Leser,

einem unglaublich fleißigen Redaktionsteam, das sowohl beim Vorbereiten der Artikel als auch bei der endgültigen Korrektur hilft und sogar eigene Beiträge liefert, habe ich es zu verdanken, daß die Phase der schlaflosen Nächte, die ich wegen der Verspätung beim Erscheinen der Mitgliederzeitung habe, ihrem Ende entgegengehen. Sie können sich vielleicht vorstellen, welche Erleichterung das für mich bedeutet.

Das neue Layout der Mitgliederzeitung, das ebenfalls nur dank gemeinsamem Einsatz des ganzen Teams möglich war, wurde im großen und ganzen sehr positiv aufgenommen, was uns natürlich besonders freut. Die Anregungen, die uns erreichten, werden wir diskutieren und bei entsprechender Entscheidung realisieren. Sie werden also bei aufmerksamer Beobachtung immer wieder kleine Änderungen am Aussehen feststellen können.

Einer der erwähnten Beiträge des Teams ist das Kreuzworträtsel, das Sie auf Seite 69 finden. Versuchen Sie sich daran, verzweifeln Sie aber nicht, wenn Sie es nicht vollständig lösen können. Es sind Fragen aus allen Bereichen gestellt, die mit TEX zu tun haben. Der Zugriff auf ein TEX-Buch ist empfehlenswert! Die Auflösung werden wir Ihnen in der nächsten Ausgabe liefern.

Außerdem finden Sie in dieser Ausgabe einen sehr prominenten Namen unter den Autoren – Donald E. Knuth. Wir haben einen Artikel einer seiner WWW-Seiten im Original veröffentlicht, da wir ihn für sehr wichtig halten. Versäumen Sie nicht, ihn zu lesen und seinen Inhalt zu verbreiten.

Ihre Luzia Dietsche

# Hinter der Bühne

## Vereinsinternes

## Grußwort

Liebe Mitglieder,

wie immer an dieser Stelle möchte ich die Ereignisse seit dem Erscheinen der letzten Mitgliederzeitung zusammenfassen.

Nachdem der Kassenbericht für das Jahr 1993 vom Finanzamt anerkannt worden ist, haben wir die Gemeinnützigkeit nun auch für die nächsten drei Jahre zugestanden bekommen. Die positive Entwicklung bei den Kassenberichten setzt sich fort und so findet man in dieser Ausgabe den Kassenbericht von 1994. In der nächsten Woche werde ich ihn beim Finanzamt vorlegen. Mein Wunsch wäre es jetzt, in der nächsten Ausgabe noch den Kassenbericht für 1995 zu veröffentlichen, denn dann wäre ein weiteres Problem gelöst.

Ein Problem, das noch zu lösen ist, besteht darin, daß der Bankeinzug der Mitgliedsbeiträge noch nicht erfolgt ist. Der Grund liegt darin, daß außer der enormen Gebührenerhöhung der Postbank (0,50 DM pro Buchung) das ganze nicht mehr wie bisher gedruckt, sondern auf einem Datenträger maschinenlesbar abgegeben werden muß. Das finanzielle Problem der Buchungen konnte umgangen werden, indem wir den Einzug wie bei Überweisungen über die Volksbank Rhein-Neckar-Mitte abwickeln. Die benötigte Software liegt schon auf meinem Schreibtisch und ich hoffe, sie im Laufe des nächsten Monats installieren zu können. Durch diese Umstellung hat sich in diesem Jahr der Bankeinzug leider verzögert, wird aber spätestens im Oktober durchgeführt.

An vielen Stellen zeigt es sich, daß wir nicht in der Lage sind, wie eine Firma zu agieren. Es darf nicht vergessen werden, daß die meisten Arbeiten freiwillig und ehrenamtlich geleistet werden. Dadurch sind wir in der Lage, mit einem relativ kleinen Budget erstaunlich viel zu bewältigen. Es sollte nicht übersehen werden, daß es uns trotz der Kostensteigerungen (Post, Bank, u. a.) möglich war, den Mitgliedsbeitrag seit vielen Jahren konstant zu halten. Allerdings hoffe ich auf ein gewisses Maß an Verständnis, wenn beispielsweise eine Büchersendung,

durch die Urlaubszeit bedingt, länger als gewohnt auf sich warten läßt oder wenn eine Antwort nicht innerhalb einer oder zwei Wochen erfolgen kann. Oft sind für Antworten auch noch gewisse Recherchen notwendig und unsere Damen am Telefon sollten nicht als *Blitzableiter* benutzt werden. Das bedeutet nicht, daß wir nicht versuchen, die Effektivität des Büros weiter zu verbessern und die Bearbeitungszeit weiter zu verkürzen. Aber wie gewohnt dauert alles seine Zeit.

In dieser Woche haben wir den CTAN-Server gegen eine schnellere Maschine ausgetauscht, um den gestiegenen Anforderungen Rechnung tragen zu können. Die dadurch frei gewordene Maschine, die viele Jahre zuverlässig ihren Dienst erfüllt hat, geht als Spende an die TUG, um einen Ersatz für den abgeschalteten CTAN-Server in Houston, Texas zu schaffen. Sie wird voraussichtlich ihren neuen Platz in Boston finden und von Karl Barry betreut werden. Dadurch hoffe ich, daß es für die TEX-Anwender in Nordamerika, die zur Zeit die CTAN-Server in Europa benutzen mußsen, wieder leichter wird, an die Software zu gelangen. Den Anwendern in Europa hilft diese Entscheidung, da die Netzbelastung erheblich gesenkt wird.

Die unerfreulichste Gegebenheit der letzten Zeit war die Bekanntmachung von Professor Donald E. Knuth, daß jemand seine Computer-Modern-Schriften verändert und damit sein *Copyright* verletzt hat. Besonders ärgerlich war dabei die Tatsache, daß dies in Deutschland erfolgt war, was von einigen unserer „Freunde“ dazu verwendet wurde, einen nicht vorhandenen Zusammenhang mit DANTE e.V. zu konstruieren. Im nachhinein stellte es sich heraus, daß alles auf Mißverständnissen und mangelnder Erfahrung basierte und es von dem Schuldigen sehr bedauert wurde. Betroffen ist das NTEX-Paket der Slackware-Linux-Verteilung. In neuen Versionen des NTEX-Pakets ist der Fehler korrigiert und ich bin mit den Verantwortlichen im Gespräch, um eine CD-ROM mit einer Fehlerkorrektur zusammenzustellen.

Um die Angelegenheit so schnell wie möglich aus der Welt zu schaffen, habe ich versucht, Verbindung mit den wichtigsten Buchverlagen aufzunehmen, die ihren Büchern eine CD-ROM beigelegt haben, die den Fehler ebenfalls enthielten. Frustrierend war es zu sehen, wie diese Verlage auf den Sachverhalt reagierten. Die Spanne der Reaktionen reichte von Ignoranz bis hin zu der Tatsache, daß ich nicht zurückgerufen wurde. Es zeigt sich an dieser Stelle, daß einzig und allein Umsatz und Gewinn zählen. Fast niemand hat interessiert, daß hier eine inkompatible Version eines TEX-Systems in großem Umfang verteilt wird, die ihren Benutzern bestimmt etliche Probleme bescheren kann. Den Verantwortlichen ist dabei ganz entgangen, daß auch ein Verlag verpflichtet ist, wie jedes andere Unternehmen seinen Kunden ein korrektes Produkt zur Verfügung zu

stellen. Man sollte einmal prüfen, ob der Käufer hier nicht ebenfalls das Recht auf Korrektur, Wertminderung oder Rückgabe des Buches hat. In jedem Falle sollte man beim Verlag oder der Buchhandlung intervenieren, daß der Fehler beseitigt und (kostenlos) eine neue CD-ROM zur Verfügung gestellt wird. Die Informationen bezüglich der Reaktionen seitens der Verlage nehme ich gerne entgegen. Sie werden dann von mir ausgewertet und gegebenenfalls veröffentlicht.

Die 500 Exemplare der CD-ROM „CTAN/3“ mit der Kopie des CTAN-Servers sind verteilt. Es können keine mehr bestellt werden, da vorerst eine Neuauflage nicht in Sicht ist. Allerdings sind Überlegungen im Gange, was in der Zukunft in dieser Richtung getan werden kann.

Bleiben mir am Ende noch zwei Dinge zu erwähnen. Herr Ralf Gärtner hat sich freundlicherweise bereit erklärt, die vakante Position des Koordinators für VAX/VMS zu übernehmen. Er wird auf der Mitgliederversammlung in Hamburg anwesend sein und ich freue mich darauf, ihn dort zu treffen. Ferner wurde ich gebeten, nochmals darauf hinzuweisen, daß eine Anmeldung zu dieser Mitgliederversammlung erforderlich ist, damit die lokalen Organisatoren besser planen können.

Mit freundlichen Grüßen verbleibe ich bis in Hamburg und hoffe, dort viele von Ihnen zu treffen.

Joachim Lammarsch
(Präsident)

## Kassenbericht für den Zeitraum 1.1.1994 – 31.12.1994

Friedhelm Sowa

In der letzten Ausgabe war zu lesen, daß wir guter Hoffnung waren, noch in diesem Jahr die Abschlüsse für 1994 und 1995 erstellen zu können. Offensichtlich handelte es sich dabei nicht um eine zu optimistische Annahme, denn auch der Berichtszeitraum 1994 ist abgeschlossen.

Das Jahr 1994 unterschied sich im Hinblick auf den Überschuß von dem der Vorjahre. Er ist deutlich geringer geworden, was auf die gesunkene Mitgliederzahl

zurückzuführen ist. Auch die getätigten Investitionen beim Anlagevermögen haben zu einer Reduzierung des Überschusses beigetragen.

Die Guthaben von DANTE e.V. entwickelten sich im Berichtszeitraum wie folgt:

| Konto bei | Anfangsbestand | | Endbestand | |
|---|---|---|---|---|
| Postgiro Beiträge | 1.1.94 | 53.049,67 DM | 31.12.94 | 33.599,11 DM |
| Postgiro Tagung | 1.1.94 | 9.233,29 DM | 31.12.94 | 11.961,85 DM |
| Postgiro Verkauf | 1.1.94 | 2.350,79 DM | 31.12.94 | 5.945,90 DM |
| Deutsche Bank | 1.1.94 | 8.986,30 DM | 31.12.94 | 9.016,44 DM |
| Volksbank | 1.1.94 | 14.566,82 DM | 31.12.94 | 120.494,52 DM |
| Kasse | 1.1.94 | 3.001,17 DM | 31.12.94 | 3.138,37 DM |
| Wertpapiere | 1.1.94 | 200.000,00 DM | 31.12.94 | 0,00 DM |
| Festgeldkonten | 1.1.94 | 0,00 DM | 31.12.94 | 120.147,48 DM |
| | | 291.188,04 DM | | 304.303,67 DM |

Auf den ersten Blick mag es verwunderlich erscheinen, zum Jahresende eine Auflösung der Wertpapiere festzustellen. Auf den zweiten Blick finden sie sich jedoch in den Festgeldkonten wieder sowie zusätzlich noch ein Betrag von 100 000,–DM auf dem Girokonto der Volksbank. Dieser Betrag wurde wegen des Ablaufs des Anlagezeitraums dem Girokonto gutgeschrieben, erscheint also zum Jahresabschluß nicht auf einem Festgeldkonto.

Um gleich die Antwort auf die Frage „Warum diese Änderung?“ zu geben: Nachdem durch den Umzug in das Büro von DANTE e.V. endlich mehr Platz zur Verfügung stand, um die Unterlagen von vertikaler auf horizontale Ordnung umzustellen, kamen dem Schatzmeister die Auszüge der Festgeldkonten in die Finger und fanden die notwendige buchhalterische Beachtung durch Aufteilung der Festgelder auf Einzelkonten.

Ein „historisches“ Ereignis fand beim Jahresabschluß statt. Die zuerst angeschafften DV-Geräte, ein PC und ein Laserdrucker, wurden jeweils bis auf einen Restwert von 1,–DM abgeschrieben. Zum 31.12.1994 betrug der Buchwert des Anlagevermögens 63 900,10 DM.

| | |
|---|---|
| Betriebsausstattung | 7.271,27 DM |
| DV-Anlagen | 42.539,51 DM |
| Büroeinrichtung | 14.089,32 DM |
| | 63.900,10 DM |

Bezüglich der Zinserträge bei den Einnahmen ist zu erwähnen, daß es sich hier zum Teil um aufgelaufene Beträge des Vorjahres handelt, die aus den Festgeldanlagen resultieren. Zukünftig wird auch hier eine periodengerechte Buchung erfolgen.

| | Ausgaben | Einnahmen |
|---|---|---|
| Druckkosten Komödie | 17.891,89 DM | |
| Gehälter Hilfskräfte | 23.420,00 DM | |
| Porto und Telefon | 29.364,67 DM | |
| Bucheinkauf | 14.659,19 DM | |
| Kontogebühren | 913,00 DM | |
| Reisekosten/Bewirtung | 23.679,65 DM | |
| CD-/Disketteneinkauf | 4.104,91 DM | |
| Tagungskosten | 9.519,29 DM | |
| Betriebs-/Bürobedarf | 3.919,72 DM | |
| Rechtsberatungskosten | 1.711,68 DM | |
| Abschreibungen | 22.009,62 DM | |
| Raumkosten | 15.127,43 DM | |
| Sonstige Kosten | 164,61 DM | |
| | 166.485,66 DM | |
| Beiträge | | 114.419,85 DM |
| Buchverkauf | | 13.636,80 DM |
| Tagungsbeiträge | | 6.480,00 DM |
| Spenden | | 1.560,00 DM |
| Kostenerstattung Disketten | | 9.282,16 DM |
| Kostenerstattung Komödie | | 120,93 DM |
| Zinsen | | 25.481,22 DM |
| Sonstige Erlöse | | 15.301,07 DM |
| | | 186.282,03 DM |
| Überschuß per 31.12.1994 | | 19.796,37 DM |

Abschließend noch eine Erläuterung zu einer Frage, die zum Kassenbericht 1993 gestellt wurde: „Warum ist die Differenz zwischen dem Endbestand und dem Anfangsbestand der Guthaben nicht gleich dem Jahresüberschuß?“

Die kaufmännisch vorbelasteten Mitglieder werden wissen, daß auch der Warenbestand eines Unternehmens zum Vermögen zählt und bilanziert wird. In unseren Kassenberichten geschieht dies nicht, weil Bücher und Disketten, die

im Lager liegen, zum Selbstkostenpreis an die Mitglieder weitergegeben werden und somit ertragsneutral sind. Nichtsdestoweniger ist eine Inventur und Bewertung jederzeit möglich, somit auch die Bilanzierung. Dies wäre jedoch nur dann erforderlich, wenn es sich bei Buch- und Diskettenverkauf nicht um einen Zweckbetrieb handeln würde. Der Jahresüberschuß spiegelt sich also nicht nur auf den Konten wider, sondern wird auch dazu verwendet, um Bücher, Disketten und CD-ROMs für die Mitglieder bereitzuhalten.

# Von fremden Bühnen

Auf Donald E. Knuths WWW-Seite `http://www-cs-faculty.Stanford.EDU/~knuth/cm.html` befindet sich seit einigen Monaten ein Aufruf an die Benutzer des Textsatzsystems TEX. Er weist darin auf zwei Probleme hin, die die von ihm erstellte Computer-Modern-Schriftfamilie betreffen. Das erste Problem, das Anfang Juli entdeckt wurde, betrifft eine nicht von Knuth autorisierte Änderung an den Quelldateien, die zu Inkompatibilitäten führt. Diese geänderten Quelldateien werden mit der Slackware-Linux-Distribution verteilt. Auf das zweite Problem, daß häufig noch veraltete Versionen der Computer-Modern-Schriften verwendet werden, wurde schon mehrmals von Jörg Knappen hingewiesen; zuletzt in der Ausgabe 1/1996, S. 52, der Mitgliederzeitung.

Bitte überprüfen Sie ihre TEX-Installation und besorgen Sie sich, falls Sie veraltete oder veränderte Computer-Modern-Schriften verwenden, die offiziellen Dateien der neuesten Version vom CTAN (Comprehensive TEX Archive Network).

## Important Message to all Users of TEX

### Donald E. Knuth

Two font problems have arisen that need to be corrected before they get even worse. One is a *serious font incompatibility problem*; the other is an *aesthetic problem of obsolete designs*. Both concern only the Computer Modern font family. I'm counting on the traditional goodwill of TEX users everywhere to help resolve these problems.

### The Slackware Linux Distribution Has Bad Fonts

Somebody made unauthorized changes to the Computer Modern source files in 1994 (i.e., to the METAFONT programs that define the Computer Modern fonts), in direct violation of my stipulation on the copyright page of *Computers & Typesetting*, Volume E.

As a result, these fonts produce quite different results on different computer systems, and they will also cause TEX to typeset your old papers with different line breaks, page breaks, overfull boxes, etc. Unfortunately, the unauthorized files somehow got substituted for the correct ones — I'm not sure when. But I am sure angry!

Another person has changed the shapes of the apostrophe and reverse apostrophe in the Computer Modern Typewriter fonts, against my wishes. Instead of naming him here, I ask him to retract his work as soon as possible.

Dear friends, I decided to put these fonts into the public domain rather than to make them proprietary; all I have asked is that nobody change them, *unless the name is changed*, so that every user can obtain equivalent results on all computer systems, now and 50 years from now. I went to enormous efforts to make TEX and METAFONT systems equivalent on hundreds of different computers, and to make them archival as no commercial software has ever been. For many years I have been careful not to make any font changes that would alter TEX's typesetting behavior. If you want to improve the fonts, go ahead, but *don't give your fonts the same name as mine.* I insist that every font named `cmr10` have the same font metrics, so that TEX will choose exactly the same line breaks and page breaks on every computer system in the world. This compatibility must be enforced by peer pressure (boycotts, bad publicity, etc.), to anybody who breaks the rules. The TEX Users Group is now deciding how best to condemn this action and to keep the cancer from infecting too many systems.

The volunteer who helped me discover why my TEX was different from his tells me that he got the adulterated font files from an InfoMagic 4CD set dated March 1995.

Look, I number the Linux folks among my personal heroes; I don't want to campaign against their fine work. They undoubtedly picked up these bad font files from somewhere in all innocence. But now the community must quickly get the word out that the CM fonts distributed with Linux since spring 1995 (at least) are grievously corrupted. The incompatible fonts must be banished from all computer systems that hope to be compatible with legitimate installations of TEX.

As far as we know, the distribution of these damaged font files has been confined to the so-called NTEX distribution, which went out with Slackware Linux. Thomas Esser's teTEX distribution, also widely used with Linux, does not have this problem, nor does the TEX Live CDROM. The directory on CTAN that contained the variant files was removed in July 1996.

### How to Tell If Your Fonts Are Corrupted

Type the following simple instructions to TEX, after the `**` prompt:

```
\setbox 0 = \hbox{ho} \showbox 0
```

Then type `x` to the `?` prompt. If TEX replies that `\box0` is an `\hbox` of width 10.55559, you're in good shape; `\relax`! But if TEX replies that `\box0` has width 10.31947, I'm sorry to say that you've got a big problem. In that case TEX will also show you a `\kern` between the `h` and the `o`.

The source of the errors can be traced to three illegitimate font files. Please get your system administrator to replace the files you have by the officially correct ones that you can find here:

- `roman.mf` (`http://www-cs-faculty.stanford.edu/~knuth/roman.mf`)
- `romlig.mf` (`http://www-cs-faculty.stanford.edu/~knuth/romlig.mf`)
- `punct.mf` (`http://www-cs-faculty.stanford.edu/~knuth/punct.mf`)

Several other files in the Slackware distribution are slightly out of date with respect to the official sources in directory `~ftp/pub/tex/cm` at `labrea.stanford.edu`, but replacing the three files above *and remaking all the fonts* will cure all the serious problems. (Actually you don't need to remake any bitmaps, except for the typewriter-style fonts `cmtt*` and `cmtex*`; all you need otherwise is to remake the font metric files, namely the files with suffix "`.tfm`". You also need to remake the "`.fmt`" files that TEX uses at the start of a job, because these contain preloaded font information.)

## Also, Please Bring Your Fonts Up To Date

If you see that your system produces the symbol

$$\delta \quad \text{instead of} \quad \delta$$

for the Greek lowercase delta, you should tell your system administrator immediately to *upgrade your obsolete version of the Computer Modern fonts.* (In this respect, Linux has the right stuff.)

I made important corrections to all those fonts in the spring of 1992, but alas, I still see many books, journals, and preprints using the old versions. Please help me abolish the old forms from the typefaces of the earth.

Many characters were improved in 1992, notably the arrows, which now are darker and have larger arrowheads, so that they don't disappear so easily after xeroxing. But most of the changes are rather subtle compared to the dramatic improvement in the lowercase delta. In fact, the old delta was so ugly, I couldn't stand to write papers using that symbol; now I can't stand to read papers that still do use it.

When I released the new versions in 1992, I also installed a new and improved procedure for digitizing the letterforms at low resolutions.

The new forms of the characters take exactly the same space on the page as the old, as far as TEX is concerned. So the same line breaks and page breaks will be obtained with the 1992 fonts as with the original fonts of 1986. The only difference will be that your papers will look better and they will be more pleasant to read.

These fonts are never going to change again. The best time to get rid of the old version is now. Correct METAFONT sources are widely available in TEX archives around the world, with dozens of mirror sites; they appear in directory `tex-archives/fonts/cm/mf`. If you use dvips for printing DVI files, it is not necessary to regenerate all the fonts with METAFONT; just delete the old ones, and dvips will make new ones as needed.

You may happen to see new publications that continue to use the old delta. If so, please tell the authors of all such papers about this Web page. Their work deserves to be typeset with more beautiful letterforms.

Incidentally, my little monograph *Axioms and Hulls* was the first publication to use the final form of the Computer Modern fonts. All details of the 1992 font refinements were documented in the fourth printing of *Computers & Typesetting, Volume E: Computer Modern Typefaces*, which came out in March 1993.

While you're upgrading the fonts, you might just as well install the latest version of `plain.tex` (April, 1996, 14 K bytes).

# Bretter, die die Welt bedeuten

## KOMA-SCRIPT – Eine Alternative zu den Standardklassen?

Markus Kohm

Wer Newsgroups wie `de.comp.tex` aufmerksam verfolgt, wird immer wieder Verweise auf KOMA-SCRIPT finden. Selbst in der DE-TEX-/DANTE-FAQ sind inzwischen mehrere Hinweise auf dieses Paket zu lesen. Auf den folgenden Seiten soll daher die Entstehung, die Motivation, die Grundzüge und die Verwendung von KOMA-SCRIPT erläutert werden. Um den Rahmen nicht zu sprengen, werden nur die Klassen `scrartcl`, `scrreprt` und `scrbook` sowie das Satzspiegelpaket `typearea` erläutert. Die Briefklasse `scrlettr` und die ergänzenden Pakete `scrpage`, `scrtime`, `scrdate`, `scraddr` und die zu `scrlettr` gehörenden Beispieldokumente `phone` und `dir` werden hier ausgeklammert. Für die Installation sei auf die Anleitungen, die zum Paket gehören, verwiesen [3].

### Die Geschichte

*Es war einmal ...*

Der Ursprung von KOMA-SCRIPT wurde noch zu LATEX 2.09-Zeiten gelegt. Irgendwann 1992 fiel Frank Neukam an der Universität Karlsruhe die Aufgabe zu, ein Vorlesungsskript zu erstellen. Dabei wurden ihm einige Unzulänglichkeiten der damaligen Standardsatzstile (*Styles*) deutlich. Bereits der Satzspiegel war mit der damaligen Fixierung auf Papier im klassischen, aber hierzulande unüblichen Letter-Format schlicht unbrauchbar und machte den starken Einfluß amerikanischer Gepflogenheiten in den Styles deutlich.

Auch die übergroßen, fetten, serifenbehafteten Überschriften und die Großbuchstaben in den Kolumnentiteln erschienen Frank als absolut unpassend. Bestärkt wurde er darin durch ein Buch von Jan Tschichold [2], das ihm zur gleichen Zeit in die Finger kam.

Begonnen hatte Frank damit, einige Anpassungen an den Standard-Styles vorzunehmen. Schließlich wurde daraus die „Document-Style-Familie SCRIPT“, zunächst in der Version 1.0.

Dieser erste Ursprung von KOMA-SCRIPT wurde von verschiedenen Leuten erweitert und ergänzt. Kurz bevor dadurch ein Versionen- und Unterversionenchaos ausbrach, veröffentlichte Frank Neukam Ende 1993 die offizielle Version 2.0 und verschwand dann in der Versenkung.

Bekanntlich erblickte aber 1994 auch LaTeX $2_\varepsilon$ das Licht der Welt. Zunächst still und leise als Beta-Version und inoffiziell begann der Siegeszug spätestens im April 1994, obwohl die erste offizielle Version erst zwei Monate später, im Juni, erschienen ist. Das war auch der Zeitpunkt, zu dem ich mein System umstellte.

Einige der Dinge, die Frank Neukam mit seiner SCRIPT-Familie eingeführt hatte, fanden sich von Anfang an in den neuen Standardklassen. So werden beispielsweise von vornherein auch die gängigsten ISO/DIN-A-Papierformate unterstützt. Die Ähnlichkeit geht sogar so weit, daß Ränder und Textbereich aus dem Papierformat *berechnet* werden. Gleichzeitig beginnen hier aber bereits wieder die wesentlichen Unterschiede. Denn während bereits SCRIPT 2.0 reiche Varianten an typographisch ausgewogenen Satzspiegeln berechnen kann, sind die Standardklassen noch immer sehr unflexibel und im deutschen Sprachraum aus typographischer Sicht höchst strittig. Auch hatte man sich hierzulande teilweise längst an die Änderungen und Erweiterungen von SCRIPT gewöhnt, so daß der Ruf nach SCRIPT für LaTeX $2_\varepsilon$ laut wurde.

Ein freundlicher Mensch bastelte innerhalb kurzer Zeit eine erste Anpassung und veröffentlichte diese als SCRIPT 2.0e. Diese Anpassung war typisch für das LaTeX $2_\varepsilon$-Umfeld Anfang bis Mitte 1994. Im Prinzip waren lediglich die Inkompatibilitäten beseitigt und aus der Endung `.sty` war `.cls` geworden. Ich will die Leistung, die hinter dieser Anpassung steckte, nicht schmälern, aber das Ganze erschien mir von Anfang an als unzureichend. Vielen Errungenschaften des neuen LaTeX war keinerlei Aufmerksamkeit gewidmet worden. Verständlich war, daß wegen der geringen Unterschiede der alten Stile *book* und *report* diese in SCRIPT nicht getrennt implementiert wurden. Absolut unzureichend war aber beispielsweise die Implementierung von Optionen und Befehlen. Überall fanden sich noch TeX-Hacks auf LaTeX-Ebene.

So begann ich im Juni 1994 die gesamte SCRIPT-Familie neu zu implementieren. Aufgrund der deutlich geänderten Struktur erschien es mir nicht sinnvoll, einfach SCRIPT 2.0 an LaTeX $2_\varepsilon$ anzupassen. Stattdessen ging ich den anderen Weg und paßte die Standardklassen an die typischen SCRIPT-Eigenschaften an.

So entstand KOMA-SCRIPT, das Ende 1995 von Frank Neukam als offizieller Nachfolger der „Document-Style-Familie SCRIPT“ anerkannt wurde.

## Das Typische an KOMA-SCRIPT

*cucullus non facit monachum*[1]

Bei der Verwendung neuer Klassen werden drei grundsätzliche Fragen von potentiellen Nutzern gestellt:

1. Weshalb soll ich diese Klasse verwenden?
2. Was unterscheidet diese Klasse von den Standardklassen?
3. Was ist das Typische dieser Klasse?

Diese Fragen sind erst recht begruendet, wennn – wie bei KOMA-SCRIPT – eine Sammlung von Klassen betrachtet wird.

Die erste Frage ist aus Sicht des Autors rasch beantwortet: „Niemand muss, Jeder darf.“ Anders ausgedrückt, wenn Sie die Antworten auf die übrigen Fragen überzeugen, werden Sie KOMA-SCRIPT verwenden wollen, wenn nicht, sollten Sie die Finger davon lassen.

Die beiden anderen Fragen sind Thema dieses Artikels. Daher zunächst nur soviel zu Frage drei: Vielfach wird von oberflächlichen Betrachtern behauptet, typisch seien die Überschriften und Kolumnentitel. Das ist nur zu einem sehr kleinen Teil richtig. In Wahrheit ist die Flexibilität das, was KOMA-SCRIPT auszeichnet. Der Grundstein hierfür wurde bereits in SCRIPT 2.0 gelegt und seither konsequent weiterverfolgt. Die *typischen* Überschriften nutzen diese Flexibilität lediglich. Sie sind aber, wie noch erläutert werden wird, keineswegs zwingend.

## KOMA-SCRIPT im Vergleich mit anderen Klassen

*Der Apfel fällt nicht weit vom Stamm.*

KOMA-SCRIPT ist aus zwei Hauptquellen entstanden, zum einen aus den Standardklassen, zum anderen diente die SCRIPT-Familie als Vorbild. Es ist nicht weiter verwunderlich, wenn KOMA-SCRIPT von beiden etwas mit auf den Weg bekommen hat.

Obwohl Spötter behaupten, an KOMA-SCRIPT sei alles anders, ist das definitiv nicht richtig. KOMA-SCRIPT ist für den Anwender zunächst genau wie die Standardklassen zu verwenden. Hier wie dort existieren drei Hauptklassen

[1] *Die Kutte macht nicht den Mönch.*

| Standardklasse: | | KOMA-SCRIPT-Klasse: |
|---|---|---|
| article | – | scrartcl |
| report | – | scrreprt |
| book | – | scrbook |
| letter | – | scrlettr |

Tabelle 1: Gegenüberstellung von Standard- und KOMA-SCRIPT-Klassen

und eine Briefklasse. Selbst die Namen ähneln sich, wie Tabelle 1 zeigt. Eine Standardklasse kann jederzeit durch die entsprechende KOMA-SCRIPT-Klasse ersetzt werden. Umgekehrt ist dies auch möglich, allerdings nur, wenn keine KOMA-SCRIPT-Erweiterungen verwendet wurden.

Für die Kompatibilität zur SCRIPT-Familie und deren Namen existieren *Style*-Dateien, die versuchen, den entsprechenden SCRIPT-Style auf eine der Klassen von KOMA-SCRIPT abzubilden. Dabei wird jedoch nicht volle Kompatibilität erreicht. Es sollte aber ausreichen, um alte Dateien mit den neuen Klassen bearbeiten zu können. Ich empfehle jedoch, stattdessen den Dokumentkopf an LaTeX 2ε anzupassen. In den meisten Fällen geht das sehr rasch und reicht bereits aus, um die gewünschten Ergebnisse zu erzielen.

## Der Satzspiegel

*Spieg'lein, Spieg'lein an der Wand ...*

### Die Standardklassen und der Satzspiegel

*Es irrt der Mensch solang er strebt.*

Seit LaTeX 2ε werden die gebräuchlichen ISO/DIN-A-Formate, die bereits in SCRIPT 1.0 eingebaut waren, von den Standardklassen unterstützt. Dabei wird von den Standardoptionen *a4paper* und *a5paper* zunächst nicht die Aufteilung zwischen Text und Rändern, sondern nur die Größe des Papiers festgelegt – auch dies ist etwas, was SCRIPT schon immer so gemacht hat. Die eigentliche Festlegung der Aufteilung zwischen Rändern und Text, des sogenannten Satzspiegels, erfolgt erst in den Dateien der Schriftgrößenoptionen (z. B. `size10.clo`). Diese *Berechnung* ist allerdings recht unflexibel und aus typographischer Sicht nicht sonderlich geglückt. Die Breite der Ränder hängt jeweils von der Schriftgröße ab. Die Papiergröße geht nur insofern in die *Berechnung* ein, als alles außer den Rändern eben Textbereich ist.

### KOMA-SCRIPT und der Satzspiegel

*... wer ist die Schönste im ganzen Land?*

Jan Tschichold beschreibt in [2] eine Satzspiegelberechnung, die abhängig von der Papiergröße ist, aber unabhängig davon funktioniert. Das scheinbare Paradoxon soll hier gelöst werden.

Die grundlegende Konstruktion basiert auf zwei grundlegenden Ideen:

1. Die sichtbaren Ränder sollen links und rechts bzw. innen und außen gleich groß sein.
2. Das Seitenverhältnis des Papiers soll sich im Seitenverhältnis des Textbereichs widerspiegeln.

Die Unterscheidung zwischen *links* und *innen* bzw. *rechts* und *außen* hat ihren Sinn.

Bei einseitigem Druck betrachten Sie jede Seite, jedes Blatt für sich alleine. Sie haben dabei einen optischen Eindruck von einem Rand links und einem Rand rechts. Es wird als angenehm empfunden, wenn beide Ränder, die Sie sehen, gleich groß sind. Der Zusatz „die Sie sehen“ wird noch eine Rolle spielen.

Betrachten Sie nun einmal ein Buch. Dieses ist beidseitig gedruckt. Hier vermitteln linke und rechte Seite zusammen einen Gesamteindruck. Zwar besitzt jede Seite zwei Ränder, der rechte Rand der linken Seite und der linke Rand der rechten Seite verschmelzen optisch jedoch zu einem einzigen Rand. Was sie von diesem *inneren* Rand sehen, sollte genauso groß sein wie jeder der beiden *äußeren* Ränder.

Dieses erste Prinzip ist damit nicht nur unmittelbar einleuchtend, es wird – zumindest zunächst – sogar von den Standardklassen erfüllt.

Das zweite Prinzip leuchtet nicht unmittelbar ein. Experimentiert man jedoch einmal mit gleich großen schwarzen Rechtecken unterschiedlicher Seitenverhältnisse auf einem weißen Blatt Papier, so merkt man rasch, daß das Auge zur Harmonie neigt.

Die Konstruktion ist nun denkbar einfach und basiert auf einer Einteilung der Breite und Höhe der Seite in $n$ gleiche Teile. Das Ergebnis ist je eine Längenangabe für die Horizontale – im folgenden mit HLE (horizontale Längeneinheit) abgekürzt – und die Vertikale (VLE).

*Beispiel:* Für eine DIN-A4-Seite ($210\,\mathrm{mm} \times 297\,\mathrm{mm}$) ergibt sich bei Zehnteilung der Seite ($n = 10$) für die horizontale Längeneinheit $\mathrm{HLE} = 210\,\mathrm{mm} \div$

$10 = 21{,}0\,\text{mm}$. Für die vertikale Längeneinheit ergibt sich $\text{VLE} = 297\,\text{mm} \div 10 = 29{,}7\,\text{mm}$.

Nun wird bei doppelseitigem Druck einfach der Anteil einer Seite am effektiv sichtbaren inneren Rand auf eine Breite von 1 HLE und der äußere Rand auf 2 HLE gesetzt. Soll einseitig gedruckt werden, so wird sowohl der linke als auch der rechte Rand auf 1,5 HLE eingestellt. Analog dazu wird ein oberer Rand von 1 VLE und ein unterer Rand von 2 VLE verwendet. Für den Textbereich bleibt so eine Fläche von $(n-3)$ HLE Breite mal $(n-3)$ VLE Höhe. Randbemerkungen („marginal notes“) stehen im äußeren bzw. rechten Rand und sind auf eine Breite von 1,5 HLE (bei Verwendung der `twoside`-Option) bzw. 1 HLE (sonst) begrenzt.

*Beispiel:* Bei Sechsteilung der Seite ($n = 6$) sind der linke und rechte Rand zusammen 3 HLE und der Textbereich ebenfalls $6 - 3 = 3$ HLE breit. Das gleiche gilt für die vertikalen Größen. Damit ist bei einer Sechsteilung der Seite der Textbereich genau halb so breit und hoch wie die gesamte Seite.

Die TEX-Forderung, daß die Textbereichsbreite (`\textwidth`) ein Vielfaches der Zeichenbreite der `Typewriter`-Schrift sein soll, wird durch die beschriebene exakte Konstruktion nicht erfüllt. Sie wird allerdings selbst von den Original-Klassen nicht unbedingt eingehalten und ist nur bei komplett in `Typewriter`-Schrift geschriebenen Zeilen von Belang. Wie wir noch sehen werden, bietet KOMA-SCRIPT trotzdem eine Möglichkeit, den Satzspiegel für solche Fälle entsprechend anzupassen.

Eine weitere Regel der Satzspiegelkonstruktion, daß `\textheight` (die Höhe des Textbereiches) gleich `\topskip` vermehrt um ein ganzzahliges Vielfaches des Grundlinienabstandes (`\baselineskip`) sein soll, wurde jedoch befolgt. Damit wird erreicht, daß bei komplett mit Fließtext gefüllten Seiten (also Seiten ohne Überschriften, Tabellen, *displayed material* oder ähnliches) die Absätze nicht auseinandergezogen werden müssen, um zu gewährleisten, daß die unterste Zeile mit dem unteren Rand des Textbereichs zusammenfällt. Der Satzspiegel kann dadurch maximal um weniger als eine Zeile höher werden, als dies bei einer exakten Konstruktion der Fall gewesen wäre.

Es bleibt die Frage, in wieviele Streifen denn nun insgesamt aufgeteilt werden soll. Es ist bekannt, daß eine optimale Lesbarkeit gegeben ist, wenn eine Zeile 60 bis 70 Zeichen lang ist. Damit ist klar, daß ein optimales $n$ nur in Abhängigkeit der Schriftgröße gewählt werden kann. Welche Größen KOMA-SCRIPT voreinstellt, geht aus Tabelle 2 hervor. Da bei KOMA-SCRIPT die Flexibilität ein hohes Gut ist, kann man den voreingestellten Wert über eine Klassenoption, die dann

| Schriftgröße: | 10 pt | 11 pt | 12 pt |
|---|---|---|---|
| DIV-Wert ($n$): | 8 | 10 | 12 |

Tabelle 2: Voreingestellte DIV-Werte bei unterschiedlichen Schriftgrößen und ISO/DIN-A4-Seitenformat

an `typearea` weitergegeben wird, verändern. Größere Werte führen dabei zu größeren Textbereichen, kleinere Werte zu kleineren Textbereichen.

### Der effektiv sichtbare Rand

*Da beißt die Maus den Rand noch ab.*

Bereits mehrfach war von *Sichtbarkeit* die Rede. Unter der sichtbaren Seite wird der Teil der Seite verstanden, der nach Bindung und ähnlichen Prozeduren noch optisch zur Seite gehört. Nur dieser Teil der Seite soll für die Berechnung herangezogen werden. Alles was bei der Bindung verloren geht, wirkt schließlich später bei der Betrachtung nicht mehr mit. Dieser Bindeverlust kann bei KOMA-SCRIPT als Bindekorrekturwert `BCOR` angegeben werden. In letzter Konsequenz bedeutet dies jedoch, daß die endgültige Druckvorlage erst dann von LaTeX berechnet/erzeugt werden kann, wenn die Bindeart feststeht. Was KOMA-SCRIPT dabei jedoch nicht berücksichtigt, sind Verschiebeeffekte, wie sie beispielsweise bei Klammerbindung auftreten.

### Implementierung der Klassenoptionen BCOR und DIV

*von hinten durch die Schulter ins Knie*

Die Klassenoptionen `DIV` und `BCOR` sind im Vergleich zu vielen anderen Optionen etwas ungewöhnlich. Sie besitzen sozusagen zusätzlich ein Argument. Bei `DIV` ist dies ein reiner Zahlenwert, bei `BCOR` ein Maß. Die Argumente folgen jeweils unmittelbar auf den Namen der Option.

Mit der Schriftgröße existiert auf den ersten Blick ein ähnliches Konstrukt. Bei genauerer Betrachtung entdeckt man jedoch, daß für jede Schriftgröße eine eigene Option definiert wurde.

Bei `DIV` könnte man unter Umständen noch einen ähnlichen Weg beschreiten und beispielsweise alle Werte von `DIV6` bis `DIV20` als eigene Option definieren. Aber bereits hierfür wären 15 Optionen notwendig. Bei `BCOR` wäre schon der Versuch einer solchen Implementierung unsinnig.

Leider ist bei LaTeX nicht vorgesehen, den Optionen ein zusätzliches Argument mit auf den Weg zu geben. Dennoch erschien es mir sinnvoll, `DIV` und `BCOR` analog zur Schrift- und Papiergröße als Optionen zu implementieren. Ich bin dazu einen etwas umständlichen Weg gegangen, den ich der Übersichtlichkeit wegen beibehalten habe, obwohl auch eine einfachere Lösung denkbar wäre.

Grundlage des Ganzen ist die Möglichkeit, eine Behandlung für *unbekannte* Optionen zu definieren und auch deren Namen zu erfragen. Genau dieser Name wird nun Buchstabe für Buchstabe durchlaufen und geprüft, ob es der jeweils nächste Buchstabe aus der Zeichenfolge „`DIV`“ oder „`BCOR`“ ist. Wurde eine der beiden Zeichenfolgen gefunden, so wird noch auf eine Ziffer getestet und im Erfolgsfall der Zahlwert bzw. die Größe als Argument abgetrennt und verarbeitet.

### Ungewöhnliche Satzspiegel

*Sich beugen ist keine Unehre, aber sich beugen lassen.*

Wie bereits erwähnt, kann es Fälle geben, in denen die Größe des Textbereichs über alles geht. Aber natürlich sollen auch in diesem Fall die Ränder noch so gut wie nur möglich eingestellt werden. Genau dafür gibt es in KOMA-SCRIPT den Präambelbefehl `\areaset`. Diesem Befehl übergibt man die gewünschte Höhe und Breite des Textbereichs, optional noch den Bindekorrekturwert, und erhält zur Belohnung einen mit diesen Werten möglichst gut gewählten Satzspiegel.

Wünscht man jedoch nur eine bestimmte Breite oder eine ungefähre Höhe, so ist es eher angebracht, sich einen dazu passenden DIV-Wert zu errechnen, durch Ausprobieren zu bestimmen oder aus der Tabelle in der Anleitung zu KOMA-SCRIPT [3] zu entnehmen.

### Berücksichtigung von Kopf- und/oder Fußzeile

*Ich weiß nicht, was soll es bedeuten ...*

Manchmal ist es sinnvoll, bei der Konstruktion des Satzspiegels auch die Kopf- und/oder Fußzeile als zum Textbereich dazugehörig zu betrachten. Dies ist beispielsweise der Fall, wenn der Kolumnentitel durch eine horizontale Linie vom eigentlichen Text getrennt ist und dadurch optisch näher an diesen heranrückt.

Speziell zu diesem Zweck gibt es die Option `headinclude` und die gegenteilige Option `headexclude`, sowie für den Fußbereich die entsprechenden Optionen `footinclude` und `footexclude`.

### Satzspiegel nicht nur für KOMA-SCRIPT-Klassen

*Den Seinen gibt's der Herr im Schlaf.*

Die gesamte Satzspiegelberechnung wird nicht von den einzelnen Klassen vorgenommen. Vielmehr verwenden die KOMA-SCRIPT-Klassen hierfür das zum Gesamtpaket gehörende `typearea`-Paket. Dieses Paket funktioniert jedoch nicht nur mit den KOMA-SCRIPT-Klassen, sondern wie jedes ordentliche Paket auch mit den Standardklassen. Die oben beschriebenen Klassenoptionen können dabei direkt bei `\usepackage` als Paketoptionen angegeben werden. Die KOMA-SCRIPT-Klassen machen im Prinzip nichts anderes.

## KOMA-SCRIPT und der Titel

*Darf's ein bißchen mehr sein?*

„Ein bißchen mehr“ ist genau das, was KOMA-SCRIPT an vielen, aber noch lange nicht allen Stellen bietet. Dabei wird versucht, nicht wahllos Befehle anzuhäufen, sondern gezielt dort zu erweitern und zu ergänzen, wo die Grenzen der Standardklassen als Fesseln empfunden werden. Der Titel ist so ein Beispiel.

Bei den Standardklassen besteht die Titelseite aus gerade einmal drei Teilen. Einem fetten Titel (`\title`), dem oder den Autoren (`\author`) und einem Datum (`\date`). Derartige Titel mögen für einen kleinen Aufsatz oder einen Buchumschlag genügen, der Mehrzahl wissenschaftlicher Arbeiten reicht er nicht.

Angenommen Herr Schlau schreibt seine Diplomarbeit. Dann möchte der Professor gerne, daß ganz oben auf der Seite das Logo der Universität, das Institut und natürlich er selbst steht. Wohlgemerkt, das soll nicht mitten auf der Seite, sondern ganz am oberen Rand stehen. Kein Problem, werden Sie sagen, muß man eben auf die Standardtitelseite verzichten und sich eine eigene basteln. „Nein, nein, sagt Herr Schlau, ich nehme KOMA-SCRIPT, da gibt es den Befehl `\titlehead`, damit geht das“. Recht hat Herr Schlau.

Als nächstes will der Professor, daß direkt über dem eigentlichen Titel noch „Diplomarbeit“ steht. Nun, das ist eigentlich ganz einfach, das könnte Herr Schlau in den `\title` mit aufnehmen, vielleicht noch die Schriftgröße verkleinern und durch vertikalen Abstand vom restlichen Titel absetzen. Vielleicht doch etwas umständlich. Richtig, denn Herr Schlau benutzt ja ohnehin schon KOMA-SCRIPT. Also schreibt er einfach `\subject{Diplomarbeit}` und muß sich um nichts mehr Sorgen machen, schließlich gibt es den Befehl genau für solche kleine Zusatzangaben.

Zum Schluß will der Herr Professor dann noch, daß unten auf der Seite die Firma Placeboplus genannt wird, weil deren Leistungspräparate die Denkanstrengungen der schwierigen Arbeit erst ermöglicht haben. Außerdem gibt es dafür eine Großpackung Hirnschmalz gratis. Obwohl es Herrn Schlau langsam zu bunt wird, setzt er mit `\publisher` noch rasch einen Werbeslogan unten auf die Titelseite, auch wenn der Befehl eigentlich nicht dafür gedacht ist.

Jetzt aber packt ihn der Ehrgeiz. Hinten auf die Titelseite könnte man doch ganz oben noch den Hinweis packen, daß Lesen bildet. Ganz unten dann, womit dieses Lesewerk gebildet wurde. Weil die Titelseite immer so rasch schmutzig wird, sollte man vor den eigentlichen Titel vielleicht noch einen Kurztitel setzen. Früher nannte man das Schmutztitel. Eigentlich fehlt dann nur noch die Widmung an seine Frau, die den Nerv hatte, alles zu korrigieren.

Schließlich sieht die Titeldefinition dann so aus:

```
% Schmutztitel
\extratitle{Wie man redet ohne etwas zu sagen}
% Kopf der Titelseite
\titlehead{\UniLogo\hfill
  \parbox{.5\textwidth}
    {\centering Universit"at Kognito\\
     Institut f"ur angewandte Denkologie\\
     Professor Kenntmichnet}}
% Kurzbezeichnung ueber dem Titel
\subject{Diplomarbeit}
% Titel, Autor, Datum
\title{Wie man redet ohne etwas zu sagen}
\author{Schweinchen Schlau}
\date{SS 2004}
% Eigentlich Verlag, Fuss der Titelseite
\publisher{Nimmst Du recht viel Placeboplus,
           verzapfst Du ordentlichen Stu"s}
% Titelrueckseite oben
\uppertitleback{Wer lesen kann, ist klar im Vorteil.}
% Titelrueckseite unten
\lowertitleback{Diese Diplomarbeit wurde mit Hilfe
                von \LaTeX\ erstellt.}
% Widmungsseite
\dedication{Meiner lieben Frau f"ur sehr viel Geduld}
```

Die Titelseiten werden dann ganz normal mit `\maketitle` gesetzt.

Ganz schlau war Herr Schlau natürlich doch nicht, denn er hat etwas Bedeutendes übersehen: KOMA-SCRIPT bietet zwar einige Erweiterungen für die Titelei, aber man muß nicht unbedingt jedesmal alle ausnutzen. Anders gesagt gilt auch hier die Regel: Weniger ist oft mehr.

## KOMA-SCRIPT und das Inhaltsverzeichnis

*„Welches Schweinderl hätten S' gern?“*

Das Inhaltsverzeichnis ist auf den ersten Blick eine klare Sache. Schließlich gibt es einfach nur den Inhalt in Form von Abschnitten und Überschriften wieder. Ganz einfach? Weit gefehlt!

Der Streit beginnt bereits bei der Frage, wie ein Inhaltsverzeichnis formatiert werden soll. KOMA-SCRIPT geht dabei den Weg, der auch bereits von den Standardklassen eingeschlagen wurde und arbeitet mit vertikalen Abständen und horizontalen Einrückungen. Es ergibt sich hier zunächst keine bemerkenswerte Änderung gegenüber den Standardklassen.

Richtig heftig kann die Auseinandersetzung bei der Frage werden, ob eigentlich Tabellen-, Abbildungs- und Literaturverzeichnis und auch der Index im Inhaltsverzeichnis zu erscheinen oder dort keinesfalls etwas zu suchen haben. Teilt man bei Behandlung dieser Streitfrage mit rostigen Nägeln versehene Baseballschläger aus, sollte die Diskussion nur in Anwesenheit eines erfahrenen Notärzteteams geführt werden. KOMA-SCRIPT hält sich bei dieser Frage einerseits raus und verschärft den Streit der Uneinsichtigen andererseits noch ein wenig.

Über die Optionen `liststotoc`, `idxtotoc`, `bibtotoc` können die entsprechenden Verzeichnisse in das Inhaltsverzeichnis aufgenommen werden. Dabei wirkt `liststotoc` gleichzeitig auf Tabellen- und Abbildungsverzeichnis. Da die Verzeichnisse üblicherweise keine Nummer erhalten, erscheinen sie im Inhaltsverzeichnis ebenfalls ohne Nummer. Auf besonderen Wunsch gibt es beim Literaturverzeichnis exklusiv die Möglichkeit dieses zusätzlich mit einer Abschnittsnummer zu versehen. Man erreicht dies mit der Option `bibtotocnumbered` an Stelle von `bibtotoc`. Es darf als fraglich betrachtet werden, ob eine solche Option tatsächlich sinnvoll ist. Vielmehr drängt sich der Eindruck auf, daß hier der Autor dem Druck der Straße nachgegeben hat.

## Überschriften, Kolumnentitel und Paginierung

*Darf's ein bißchen weniger sein?*

Bei Überschriften und Kolumnentiteln heißt es bei den Standardklassen plötzlich nicht kleckern, sondern klotzen. Überschriften werden reichlich groß und fett gesetzt, Kolumnentitel – das sind die Überschriftenwiederholungen in der Kopfzeile – sogar in Großbuchstaben. Nun will ich mich gewiß nicht in den Streit einmischen, ob Großbuchstaben an dieser Stelle angebracht sind. Betrachtet man sich einmal die deutschen Typographie-Gewohnheiten, so wird man diese jedenfalls sehr selten finden. Bemerkenswert erscheint in diesem Zusammenhang noch die Tatsache, daß mit Kapitälchen eine TEX-Standardschriftart zur Verfügung steht, die ebenfalls Großbuchstaben, diese aber wesentlich weniger wuchtig, bietet.

### Kolumnentitel und Paginierung

*Das Schild ist's, das die Kunden lockt.*

KOMA-SCRIPT macht Schluß mit den Großbuchstaben in den Kolumnentiteln. Stattdessen wird die Schriftart über das Makro `\headfont` definiert, das auf eine geneigte Schrift voreingestellt ist.

Bei den Standardklassen wird der Kopfbereich beim Seitenstil `headings` oder `myheadings` sowohl für Kolumnentitel als auch Paginierung – die Angabe der Seitenzahl – verwendet. Beim Seitenstil `plain` wird jedoch der Fußbereich für die Paginierung eingesetzt. Der Seitenstil `plain` wird beispielsweise automatisch bei Seiten mit Kapitelüberschriften verwendet. Dadurch *springt* die Seitenzahl zwischen Kopf- und Fußbereich und teilweise zwischen Seitenrand und Seitenmitte. Im Buchdruck ist derartiges eine mittlere Katastrophe. Das Auge ist ständig auf der Suche nach der Seitennummer und einem Ruhepunkt. Beispielsweise entwickelt sich bei einem Nachschlagewerk die Suche nach einer bestimmten Seite zu einem Slalom.

KOMA-SCRIPT räumt auch hier auf. Paginierung findet immer in der Fußzeile statt. Bei einseitigem Druck (`oneside`) steht die Seitenzahl in der Mitte, bei doppelseitigem Druck (`twoside`) am äußeren Rand. Wie man dies ändern kann, ist in der Anleitung [3] nachzulesen. Die Schriftart der Seitennummer wird mit dem Makro `\pnumfont` festgelegt. Voreingestellt ist hier die Standardschrift (`\normalfont`).

Natürlich gibt es Kritiker, die zwar mit den beschriebenen Änderungen sehr zufrieden sind, denen sie aber nicht ganz genügen. Es fehlt noch eine Kleinigkeit. Die vermutlich häufigste Frage bezüglich der Kopf- und Fußzeilen ist die

Frage nach Linien zwischen Kopf- und Text- bzw. Text- und Fußbereich. Solche Linien rücken die Bereiche optisch dichter aneinander, obwohl sie häufig als *Trennlinien* bezeichnet werden.

KOMA-SCRIPT bietet mit den beiden Klassenoptionen `headsepline` und `footsepline` eine Möglichkeit, um solche Linien einzuschalten bzw. mit `headnosepline` und `footnosepline` auszuschalten. Normalerweise sind sie abgeschaltet.

Wie bereits erwähnt, wird der Kopf- bzw. Fußbereich durch eine entsprechende Trennlinie optisch dichter an den Textbereich herangerückt. Deshalb wird in diesen Fällen der Kopf- bzw. Fußbereich bei der Satzspiegelberechnung zum Textbereich und nicht zu den Rändern gerechnet. Wie man dies wieder abschalten kann, wurde bereits in einem früheren Abschnitt beschrieben.

### Überschriften

*Du sublime au ridicule il n'y a qu'un pas.*[2]

Es wurde auch schon darauf hingewiesen, daß die Überschriften der Standardklassen relativ massiv ausfallen. Ein Grund hierfür liegt in der Schrift. Die Redundanz serifenbehafteter Schriften wird im allgemeinen als Vorteil betrachtet, da dadurch die Lesegeschwindigkeit unter Umständen positiv beeinflußt werden kann. Bei sehr großen und zudem fetten Buchstaben wirken sich die Serifen jedoch häufig negativ auf das Erscheinungsbild aus. Deshalb wurden serifenlose Schriften zunächst auch hauptsächlich in Überschriften eingesetzt. Inzwischen ist ein eher philosophischer als typographischer Streit darüber ausgebrochen, warum nun welche Schrift schneller gelesen werden kann. An diesem Streit will sich KOMA-SCRIPT nicht beteiligen. Dennoch ist bei KOMA-SCRIPT für Überschriften eine serifenlose, fette Schrift voreingestellt. Dies kann man jedoch über Umdefinierung des Makros `\sectfont` beeinflussen. Mit einem einfachen

```
\renewcommand*{\sectfont}{\normalfont\bfseries}
```

erreicht man, daß die Überschriften wieder im Standardfont und fett gesetzt werden. Wer der Meinung ist, serifenlos wäre zwar ganz schön, fett aber überflüssig, der definiere einfach:

```
\renewcommand*{\sectfont}{\normalfont\sffamily}
```

Die Größe der Überschriften ist der nächste Punkt, der nicht jedermanns Geschmack trifft. Spätestens wenn man eine kleinere Papiergröße, beispielsweise `a5paper`, verwendet, nehmen die Überschriften zuviel Platz ein und springen dem Leser wie ein kalter Waschlappen an einem heißen Sommertag ins Gesicht.

[2] *Vom Erhabenen zum Lächerlichen ist nur ein Schritt.*

| Befehl | Schriftgröße bei | | |
|---|---|---|---|
| | `bigheadings` | `normalheadings` | `smallheadings` |
| `\part` | `\Huge` | `\huge` | `\LARGE` |
| `\chapter` | `\huge` | `\LARGE` | `\Large` |
| `\section` | `\Large` | `\Large` | `\large` |
| `\subsection` | `\large` | `\large` | `\normalsize` |
| `\subsubsection` | `\normalsize` | `\normalsize` | `\normalsize` |
| `\paragraph` | `\normalsize` | `\normalsize` | `\normalsize` |
| `\subparagraph` | `\normalsize` | `\normalsize` | `\normalsize` |

Tabelle 3: Klassenoptionen und ihre Auswirkungen auf die Überschriftengrößen bei `scrbook`

Auf der anderen Seite sind große Überschriften mit entsprechendem Leerraum auch nicht in jedem Fall zu verachten.

Über die Klassenoptionen `bigheadings`, `normalheadings` und `smallheadings` existieren in KOMA-SCRIPT wahlweise drei Größenabstufungen für die Überschriften. Tabelle 3 gibt die jeweiligen Schriftgrößen bei `scrbook` in Abhängigkeit der Klassenoption an. Voreingestellt ist `bigheadings`.

Eine weitere Änderung bei den Überschriften betrifft nur `book`/`scrbook` und `report`/`scrreprt`. Normalerweise sieht eine Kapitelüberschrift (`\chapter`) bei den Standardklassen so aus:

# Kapitel 1

# Das Liebesleben der Murmeltiere

KOMA-SCRIPT verzichtet auf das Wörtchen „Kapitel“ und setzt die Kapitelüberschriften wie alle anderen Abschnittsüberschriften *einzeilig*. Bei Verwendung der Option `smallheadings` sieht die Überschrift dann so aus:

## 1 Das Liebesleben der Murmeltiere

Als letztes werden Abschnittsüberschriften über das Makro `\raggedsection` formatiert, das als `\raggedright` vordefiniert ist. Dadurch findet innerhalb von Überschriften keine Trennung mehr statt. Bevorzugt man stattdessen zentrierte Überschriften, so kann man das Makro entsprechend umdefinieren:

```
\let\raggedsection=\centering
```

### Abschnittsnumerierung

*der Punkt nicht nur auf dem „i“*

Laut Duden werden die Nummern in der Abschnittsgliederung ohne abschließenden Punkt geschrieben, wenn alle Abschnittsnummern nur aus arabischen Ziffern bestehen [1, Regel 5]. Ein abschließender Punkt ist hingegen zu setzen, wenn bei der Abschnittsnumerierung auch römische Zahlen oder Großbuchstaben verwendet werden. Dabei spielt es keine Rolle, ob in der konkreten Nummer nun römische Zahlen oder Buchstaben vorkommen [1, Regel 6].

Die Standardklassen sind für den englischen Sprachraum konzipiert und kennen deshalb die Duden-Regeln nicht. KOMA-SCRIPT ist für den deutschen Sprachraum konzipiert. Die Einhaltung der Regeln ist jedoch nicht ganz einfach. In KOMA-SCRIPT ist ein einfaches Verfahren implementiert, das bei Verwendung von `\part` – genauer wird dabei die Definition der Numerierung der Teile betrachtet – oder einem Anhang auf Numerierung mit abschließendem Punkt umschaltet. Damit auch die Abschnitte, die vor der Verwendung eines entsprechenden Befehls liegen, korrekt numeriert werden, muß ein weiterer LaTeX-Lauf durchgeführt werden. Da die meisten LaTeX-Dokumente ohnehin zwei Durchläufe benötigen, sollte dies keine große Rolle spielen.

So schön es ist, wenn die Duden-Regeln automatisch erfüllt werden, so ungünstig ist das, wenn die Automatik einmal versagt oder unerwünscht ist, beispielsweise weil man einen fremdsprachlichen Text schreibt, für den andere Regeln gelten. Deshalb existieren in KOMA-SCRIPT die beiden Klassenoptionen `pointednumber` und `pointlessnumber`, mit denen die Erscheinungsform fest eingestellt werden kann.

### Abschnitte ohne Numerierung

*Weniger ist oft mehr.*

Neben den Standardbefehlen der Abschnittsgliederung bietet KOMA-SCRIPT drei weitere.

Von Zeit zu Zeit benötigt man neben den numerierten Abschnittsüberschriften auch solche ohne Nummern. In allen Standardklassen und somit natürlich

auch in KOMA-SCRIPT sind dafür *Sternvarianten* der Gliederungsbefehle definiert. Diese haben dasselbe Grundaussehen wie die entsprechenden Befehle ohne Stern, setzen jedoch keine Nummer und keinen Eintrag ins Inhaltsverzeichnis. Außerdem wird darauf verzichtet, den Kolumnentitel zu verändern. Dieses Vorgehen hat eine gewisse Berechtigung, wenn man solche Abschnitte zwischen normal numerierten Abschnitten verwendet. Am Anfang oder Ende des Textes wäre aber zumindest die Anpassung des Kolumnentitels dringend erforderlich. Versuchen Sie beispielsweise einmal folgendes Minidokument.

```
\documentclass[twoside]{article}
\newcommand{\FastNichts}{\null\vfill
  \begin{center}
  Seite (fast) ohne Text
  \end{center}
  \vfill\clearpage}
\pagestyle{headings}
\begin{document}
\section{Hauptabschnitt}
\subsection{Unterabschnitt mit Nummer vorher}
\FastNichts
\subsection*{Unterabschnitt ohne Nummer}
\FastNichts
\FastNichts
\subsection{Unterabschnitt mit Nummer nachher}
\FastNichts
\end{document}
```

Sie werden sehen, daß das Ergebnis vermutlich nicht dem entspricht, das Sie erwartet hätten. Im Kopf von Seite drei ist plötzlich und unerwartet wieder „Unterabschnitt mit Nummer vorher“ zu lesen. Eigentlich würde man dort eher „Unterabschnitt ohne Nummer“ erwarten.

In vielen Fällen ist es außerdem erwünscht, daß eine solche Überschrift doch in das Inhaltsverzeichnis aufgenommen wird. Bei den Standardklassen ist dies nur *zu Fuß* möglich.

Bei KOMA-SCRIPT gibt es genau für diese Fälle die beiden Befehle `\addchap` und `\addsec`. Beides sind vollwertige Befehle der Abschnittsgliederung, die sich von `\chapter` bzw. `\section` nur dadurch unterscheiden, daß keine Numerierung erfolgt. Es gibt also ebenfalls die Möglichkeit, ein optionales Argument oder einen Stern zu verwenden. Das optionale Argument wird wie gewohnt verwendet, bei der Sternvariante wird lediglich der Eintrag in das Inhaltsver-

zeichnis unterdrückt. Als Äquivalent zu `\chapter` steht `\addchap` bei `scrartcl` selbstverständlich nicht zur Verfügung.

Am Anfang dieses Abschnittes war von drei Befehlen die Rede. Vorgestellt wurden bisher aber nur zwei. Der dritte Befehl stellt eine Besonderheit dar. Er ist kein Gliederungsbefehl im Sinne von `\chapter`, `\section`, `\paragraph` etc., sondern dient eher als eine Art *Hilfsüberschrift*.

In Lehrbüchern[3] findet man oft Konstrukte wie:

> **Beispiel:**
>
> Legt man die Diskette nicht in der dafür vorgesehenen, korrekten, also sozusagen richtigen, sondern in einer nicht vorgesehenen, inkorrekten, also sozusagen falschen Weise, z. B. seitenverkehrt, also mit dem Schieber nach vorn, aber weiterhin mit der Antriebsmittelrundscheibe nach unten, ein, so erhält man in keinem Fall das gewünschte, nämlich beabsichtigte Ergebnis der Möglichkeit, das Medium, also die Diskette, anschließend formatieren, lesen und bzw. oder beschreiben zu können.

Ein solches Konstrukt mit den Standardmitteln nachzubilden, erscheint zunächst einfach, wenn auch umständlich:

```
\noindent\textbf{Beispiel:}\par\noindent
Legt man die Diskette nicht in der daf"ur vorgesehenen,
korrekten, also ...
```

Erschwerend kommt jedoch noch hinzu, daß hier außerdem ein Seitenumbruch nach der *Hilfsüberschrift* „Beispiel:“ möglichst unterdrückt und stattdessen eher ein Umbruch davor durchgeführt werden sollte. Der Schreib- und Wissensaufwand steigt also noch einmal an. Es lag daher nahe, bereits innerhalb der Klassen einen Befehl zu definieren. Bei KOMA-SCRIPT lautet dieser `\minisec` und erhält als Argument den Text der Hilfsüberschrift. In KOMA-SCRIPT könnte obiges Beispiel also so geschrieben werden:

```
\minisec{Beispiel:}
Wer eine Diskette anders als mit dem Metallschieber
nach vorn und dem Antriebsloch nach unten in den
horizontalen Schlitz eines Diskettenlaufwerks
schiebt, mu"s damit rechnen, da"s sie sich
verklemmt und nur noch mit derselben rohen Gewalt
entfernen l"a"st, mit der sie eingelegt wurde.
```

[3] Zur Erinnerung: SCRIPT wurde ursprünglich für Vorlesungsskripten erstellt.

Für die Darstellung dieser Hilfsüberschrift wird ebenfalls `\sectfont` verwendet.

## Erweiterungen am Rande

*Neben dem Schiff ist gut schwimmen.*

### Randnotizen

*Die „janze“ Richtung paßt uns nicht!*

Bei einigen Autoren sind Randnotizen sehr beliebt. So soll beispielsweise ein „Achtung!“ oder auch nur ein Ausrufezeichen am äußeren Rand die Aufmerksamkeit auf wichtige Stellen lenken. Ein Problem dabei ist, daß man beim Schreiben nie weiß, ob der äußere Rand nun links oder rechts liegt. Nun, diese Überlegung wird einem vom LaTeX-Standardkonstrukt `\marginpar` bereits abgenommen. Auf der linken Seite eines mit `twoside` gesetzten Textes kommt jedoch erschwerend hinzu, daß `\marginpar` die Randnotiz ebenfalls linksbündig setzen würde, kurze *Signaltexte* also vom Text abgerückt werden. Hier wäre es nützlich, wenn die Randnotiz sich grundsätzlich an den Text *anlehnen*, also auf linken Seiten rechtsbündig und auf rechten Seiten linksbündig gesetzt würde. Auch dies läßt sich mit `\marginpar` erreichen, vorausgesetzt man kennt dessen optionales Argument. Nachteilig ist dabei, daß der Signaltext zweimal geschrieben werden muß. Es liegt daher nahe, einen eigenen Befehl zu definieren, der einem diese Arbeit abnimmt. Genau das tut KOMA-SCRIPT mit dem Befehl `\marginline`.

Leider ist die Definition von `\marginpar` im LaTeX-Kernel etwas zu einfach ausgefallen. Die Entscheidung, ob es sich um eine rechte oder eine linke Seite handelt, also das optionale oder das normale Argument zu verwenden ist, wird bereits bei der Abarbeitung des Makros getroffen. Wandert die Notiz nun im Zuge der Absatz- und Seitenformatierung nachträglich auf die nächste Seite, so war die Entscheidung falsch. Es wird dann fälschlicherweise das falsche Argument von `\marginpar` für die Randnotiz verwendet. Da `\marginline` auf dem `\marginpar`-Makro basiert, kann damit im ungünstigsten Fall auch die Formatierung der Randnotiz mit `\marginline` genau falsch sein.

### Noch eine Listenumgebung

*Die Zahl ist das Wesen aller Dinge.*

Man könnte eigentlich meinen, in LaTeX wären bereits genug Listenumgebungen definiert. Im Bereich der Mathematik werden jedoch häufig Gliederungen der Art verwendet:

**Gegeben:** Die Menge aller natürlicher Zahlen, die Menge aller Vielfachen von 2 und die Menge aller Vielfachen von 13.

**Gesucht:** Die kleinste Zahl, die sich in der Schnittmenge der drei Mengen befindet und gleichzeitig größer als 783 ist.

Dabei wäre es schön, wenn die Formatierung etwas gefälliger und damit eher wie folgt wäre:

**Gegeben:** Die Menge aller natürlicher Zahlen, die Menge aller Vielfachen von 2 und die Menge aller Vielfachen von 13.

**Gesucht:** Die kleinste Zahl, die sich in der Schnittmenge der drei Mengen befindet und gleichzeitig größer als 783 ist.

Eine Formatierung in dieser bzw. in ähnlicher Art kann in KOMA-SCRIPT mit Hilfe der `labeling`-Umgebung erreicht werden. Die Syntax dafür ist denkbar einfach:

```
\begin{labeling}[~]{\textbf{Gegeben:}}
\item[\textbf{Gegeben:}]
  Die Menge aller nat"urlicher Zahlen,
  die Menge aller Vielfachen von~\(2\) und
  die Menge aller Vielfachen von~\(13\).
\item[\textbf{Gesucht:}]
  Die kleinste Zahl, die sich in der
  Schnittmenge der drei Mengen befindet
  und gleichzeitig gr"o"ser als~\(783\) ist.
\end{labeling}
```

Natürlich kann man diese Umgebung auch für ganz andere Dinge verwenden. Die Anleitung zu KOMA-SCRIPT nutzt sie beispielsweise für die Befehls- und Variablenübersicht zur Briefklasse `scrlettr`.

## Ausblick

*Heute ist nicht alle Tage, ich komm' wieder, keine Frage.*

KOMA-SCRIPT ist zwar bereits ein sehr stabiles Paket, trotzdem gibt es darin noch die einen oder anderen Punkte zu verbessern. Darüber hinaus entstehen in Zusammenarbeit mit den Anwendern ständig neue Möglichkeiten. Wohin der KOMA-SCRIPT-Zug fährt, ist kaum zu sagen. Die bisherige Entwicklung ist in den Dokumenten `scr_new1` bis `scr_new9` sowie `scrnew10` und folgende, die zum KOMA-SCRIPT-Paket gehören, grob skizziert.

Nicht erläutert wurden in diesem Artikel die kleineren und größeren Beigaben, die KOMA-SCRIPT beiliegen, sowie die KOMA-SCRIPT-Briefklasse. Näheres dazu kann der Anleitung zu KOMA-SCRIPT [3] entnommen werden, deren gründliches Studium überhaupt empfohlen wird. Möglicherweise wird es zum Thema KOMA-SCRIPT auch noch weitere Artikel geben.

Ob und inwiefern KOMA-SCRIPT nützlich oder brauchbar ist, und inwieweit es sinnvoll erscheint, die Klassen von KOMA-SCRIPT an Stelle der Standardklassen einzusetzen, wage ich hier nicht zu beurteilen. Die überwiegend positive Resonanz läßt jedoch hoffen, daß die Arbeit nicht ganz umsonst war. Die eingangs gesteltle Frage nach einer möglichen Alternative zu den Standardklassen möchte ich zum Abschluß mit dem Hinweis beantworten, daß einige der Konzepte und Möglichkeiten von KOMA-SCRIPT auch in den neuen Klassen der *Nederlandse TEX Gebruikersgroep* (NTG) zu finden sind.

## Literatur

[1] Duden: *Die deutsche Rechtschreibung*; Band 1, 20. Auflage, Dudenverlag; 1991.

[2] Jan Tschichold: *Ausgewählte Aufsätze über die Gestalt des Buches und der Typographie*; Zweite Auflage, Birkhäuser Verlag, Basel; 1987.

[3] Frank Neukam, Markus Kohm, Harald Sommerfeldt; *Das* KOMA-SCRIPT *Paket*; 1996. CTAN: `macros/latex/contrib/supported/koma-script`, `scrguide.{tex,dvi,ps}`

# Orale Spielereien mit TEX – Teil III (Addendum)

Bernd Raichle

TEX fügt in einigen Situationen, beispielsweise innerhalb *if*-Abfragen, von sich aus das Token `\relax` ein. Dieses eingefügte Token erscheint bei Makros, die vollständig expandierbar sein müssen, leider in der Ausgabe. Es verhindert Ligaturen oder führt in manchen, wenn auch seltenen Fällen zu Fehlern. Zwar sind vollständig expandierbare Makros nur selten notwendig, jedoch muß man, wenn man sie doch benötigt, einen Weg finden, TEX davon abzuhalten, das Token `\relax` einzufügen.

In der zuletzt erschienenen Folge dieser Serie [1] stellte ich einige Lösungen vor, die jedoch alle ihre Einschränkungen haben. Kurz nach Redaktionsschluß wurde ich von David Kastrup auf eine sehr einfache Möglichkeit hingewiesen, wie man eine Einschränkung für die Lösung mit dem Makro `\CheckForRelax` aufheben kann.

Bevor Sie beginnen, hektisch nach der letzten Folge zu suchen, wiederhole ich hier kurz die Definition des Makros `\CheckForRelax`.

```
\def\CheckForRelax#1{%
  \ifx\relax#1%      % Token = \relax?
    \expandafter\ForgetArgument
    % => \ForgetArgument{#1}
  \else
    \expandafter\DoNotForgetArgument
    % => \DoNotForgetArgument{#1}
  \fi {#1}}
\def\ForgetArgument#1{}
\def\DoNotForgetArgument#1{#1}
```

Wie ausführlich in [1] beschrieben, kann man mit diesem Makro das als Beispiel für alle Vergleiche gewählte Makro `\VergleicheZahlMitNull` mit der Definition

```
\def\VergleicheZahlMitNull#1{%
  \message{-\ifnum 0=#1\fi-}}
```

so umformen, daß TEX zwar in einigen Fällen ein `\relax` vor dem `\fi` einfügt, dieses aber von `\CheckForRelax` wieder entfernt wird.

```
\def\NeuesVergleicheZahlMitNull#1{%
  \message{-\expandafter\CheckForRelax\ifnum 0=#1\fi-}}
```

Ruft man nun die beiden eben gezeigten Makros `\VergleicheZahlMitNull` und `\NeuesVergleicheZahlMitNull` mit Null auf, so erkennt man, daß dieser kleine Trick wirkt und das von TEX automatisch eingefügte `\relax` wieder entfernt wird.

| Eingabe: | Protokollausgabe: |
|---|---|
| `\VergleicheZahlMitNull{0}` | `-\relax -` |
| `\NeuesVergleicheZahlMitNull{0}` | `--` |

Leider hat `\CheckForRelax` den unangenehmen Seiteneffekt, daß das Makro zum einen immer das erste Token des *then*-, *else*- oder *or*-Teils der Bedingung liest und eine eventuell vorhandene Klammerung entfernt. Zum anderen wird auch immer ein vom Makroprogrammierer mit gutem Grund eingefügtes `\relax` entfernt, beispielsweise um eine zu frühe Expansion eines `\ifmmode`-Vergleichs zu Beginn eines Tabelleneintrages zu verhindern [2, S. 240].

David Kastrup wies mich auf eine so einfache Lösung dieses Problems hin, daß ich mich im nachhinein immer noch wundere, nicht selbst darauf gekommen zu sein! Wenn in allen Fällen `\CheckForRelax` immer das erste Token des *then*- oder des *else*-Teils liest und, falls dieses ein `\relax` ist, entfernt, könnte man doch einfach in jeden Zweig einer Bedingung als erstes Token *immer* ein `\relax` einfügen.

Fügt man in dem vorher gezeigten Makro `\NeuesVergleicheZahlMitNull` ein `\relax` als erstes Token des *then*-, *else*- und *or*-Zweiges ein, erhalten wir

```
\def\BesseresVergleicheZahlMitNull#1{%
  \message{-\expandafter\CheckForRelax\ifnum 0=#1\relax\fi-}}
```

Dieses Makro kann endlich ohne die beschriebenen Restriktionen verwendet werden (`\Null` ist im folgenden als `\chardef\Null=0` definiert).

| Eingabe: | Protokollausgabe: |
|---|---|
| `\BesseresVergleicheZahlMitNull{0}` | `--` |
| `\BesseresVergleicheZahlMitNull{\Null}` | `--` |

Folgendes hoffentlich nicht ganz abwegige Beispiel soll diese Lösung verdeutlichen. Darüberhinaus zeigt es, daß man das Einfügen von `\relax` und anschließende Entfernen mit `\CheckForRelax` auch anwenden kann, wenn der *then*- oder *else*-Teil Tokens enthält, die man von der Bedingung trennen muß, damit sie zur Auswertung der *if*-Abfrage nicht mit einbezogen werden.

Das Makro `\PageNumbers` bekommt in zwei Argumenten den Beginn und das Ende eines Zahlenbereichs *x*–*y*. Ist die erste Zahl größer als die zweite, soll ein Fehlertext ausgegeben werden. Sind die beiden Zahlen gleich, soll nur eine einzige Zahl, ansonsten sollen beide Zahlen als Bereich angezeigt werden.

```
\def\PageNumbers#1#2{%
  \ifnum#1=#2%
    #1%
  \else
    \ifnum #1<#2%
```

```
      #1--#2%
    \else
      {\bf Error!}%
  \fi\fi}
```

Leider bekommen wir mit dieser Realisierung in vielen Fällen unerwartete Ergebnisse, da in beiden *if*-Bedingungen die zweite Zahl aus dem Argument `#2` *und* dem im *then*-Teil stehenden Argument `#1` gebildet wird.

| Eingabe: | Expansion: |
|---|---|
| `\PageNumbers{1}{2}` | `--2` |
| `\PageNumbers{1}{1}` | `--1` |
| `\PageNumbers{1}{0}` | `\relax` |

Erst nachdem wir nach den beiden Vergleichen ein `\relax` einfügen, sind die Ergebnisse korrekt. Wollen wir jedoch das Makro zur Ausgabe in eine Datei verwenden, stört das verbliebene, nicht expandierbare Token `\relax`.

```
\def\PageNumbers#1#2{%
  \ifnum#1=#2\relax
    #1%
  \else
    \ifnum #1<#2\relax
      #1--#2%
    \else
      {\bf Error!}%
  \fi\fi}
```

| Eingabe: | Expansion: |
|---|---|
| `\PageNumbers{1}{2}` | `\relax 1--2` |
| `\PageNumbers{1}{1}` | `\relax 1` |
| `\PageNumbers{1}{0}` | `{\bf Error!}` |

Dieses Token `\relax` kann man nun mit `\CheckForRelax` nach der Auswertung der Bedingung entfernen. Hierbei ist es auch wichtig, vor dem letzten *else*-Teil ein `\relax` einzufügen. Andernfalls würde die Gruppe entfernt und die Schriftumschaltung mit `\bf` nicht mehr auf „Error!“ begrenzt.

```
\def\PageNumbers#1#2{%
  \expandafter\CheckForRelax
  \ifnum#1=#2\relax
    #1%
```

```
\else \relax
  \expandafter\CheckForRelax
  \ifnum #1<#2\relax
    #1--#2%
  \else \relax  % <= wichtig!
    {\bf Error!}%
\fi\fi}
```

Damit erhalten wir nun das gewünschte Verhalten ohne das manchmal störende Token `\relax`.

| Eingabe: | Expansion: |
|---|---|
| `\PageNumbers{1}{2}` | `1--2` |
| `\PageNumbers{1}{1}` | `1` |
| `\PageNumbers{1}{0}` | `{\bf Error!}` |

Dies wäre es für diesen Nachtrag zum dritten Teil. Wenn Sie herausfinden können, was die folgenden Zeilen erzeugen, wissen Sie schon das Thema des nächsten Teils dieser Serie.

```
\def\1#1#2\fi#3#4{\fi\ifx.#4#1#2\else\1{#1#3}#2#4\fi}
\if\1S\fi crhslteainfdeenn~~uondde~rR~eakuusrpsrioobni,e~rvte?..
```

### Literatur

[1] Bernd Raichle: *Orale Spielereien mit TEX – Teil III*; in: *Die TEXnische Komödie*, 4/95, S. 15–29, 1995.

[2] Donald E. Knuth: *The TEXbook*; Addison-Wesley Publishing Company; Reading, Mass., 1992.

## Dokumentation von C-Programmen

### Ulrich Breymann

Das Makropaket *c.sty* ermöglicht die Darstellung eines C++-Programms als Teil eines LATEX-Dokuments. Innerhalb der C++-Kommentare ist dabei LATEX-Markup erlaubt, so daß beispielsweise komplexe Formeln zur Dokumentation

eingefügt werden können. Vorteilhaft daran ist, daß ein Programmstück immer noch ohne Änderungen als C++-Quellcode verwendbar ist. Sie können in den Kommentaren Ihres C++-Quellcode LaTeX-Anweisungen einfügen, ohne daß sich der Compiler daran stört.

Im Vergleich zu `cweb` ist *c.sty* sehr einfach, genügt aber für entsprechend einfache Programmdokumentationen.

## Beschreibung

Das Prinzip von *c.sty* ist einfach. Der C++-Code wird in einer `verbatim`-ähnlichen Umgebung gesetzt, die Kommentare des C++-Code werden jedoch als „normaler“ Text gesetzt.

Die Umgebung wird innerhalb eines LaTeX-Dokuments mit `\beginprog` begonnen und endet mit `\endprog`. Soll eine Datei mit C++-Code in ein Dokument eingebunden werden, kann man `\progfile{`*datei.ext*`}` verwenden.

Für die Sonderbehandlung des Kommentars ist es notwendig, daß zu Beginn eines Kommentars die `verbatim`-ähnliche Umgebung verlassen und am Endes des Kommentars diese weiter fortgesetzt wird. In C++ können Kommentare durch zwei Konventionen eingegeben werden:

1. Ein Kommentar wird, wie in C, mit der Zeichenkombination `/*` begonnen. Der Kommentar wird durch das erste Auftreten der Zeichenkombination `*/` beendet und kann auch über mehrere Zeilen gehen.
2. In C++ kann ein Kommentar auch mit der Zeichenkombination `//` begonnen werden. Der Kommentar reicht dann nur bis zum Zeilenende, kann also *nicht* über mehrere Zeilen gehen.

Das Makropaket *c.sty* behandelt daher das Zeichen `/`, mit dem beide Male ein Kommentar begonnen werden muß, gesondert. Ebenso wird abhängig vom Kommentartyp entweder nach `*/` oder nach dem Zeilenende gesucht.

## Einschränkungen

Für den C++-Code, den *c.sty* problemlos formatieren kann, gibt es noch einige Einschränkungen:

- Backslashes (\) im C-Quellcode werden noch nicht unterstützt. So können Backslashes auch nicht in Strings und Zeichenkonstanten verwendet werden. Beispiele:

```
#include "PROJ\myheader.h"
'\n'
```

- Kommentare in der Form `/*...*/` werden zwar unterstützt, das abschließende `*/` und die zweite und folgenden Zeilen bei mehrzeiligen Kommentaren folgt leider nicht der Einrückung des öffnenden `/*`. Eine neue Zeile wird durch `\par` erreicht.

Die Kombination von C-Code und C-Kommentaren mit LaTeX-Markup unterliegt einem grundsätzlichen Konflikt, wenn in C übliche Zeichen in LaTeX eine besondere Bedeutung besitzen. Ein solches Zeichen ist beispielsweise der Unterstrich `_`.

Der Unterstrich im Kommentar des C-Programms

```
int d_neu = 0; // d_neu wird initialisiert
```

muß bei Verarbeitung mit *c.sty* durch die LaTeX-Anweisung `\_` ersetzt werden. Dabei bietet es sich zur Erhöhung der einfachen Lesbarkeit an, die Variable komplett in `\texttt` zu setzen:

```
int d_neu = 0; // \texttt{d\_neu} wird initialisiert
```

Damit ist die Interpretation mit *c.sty* korrekt und wir erhalten das Ergebnis

`int d_neu = 0; //` **d_neu** wird initialisiert

Innerhalb eines C++-Kommentars müssen also LaTeX-Konventionen eingehalten werden. Bei Interesse kann *c.sty* selbstverständlich von Jedem erweitert werden, der Lust dazu hat, die Beschränkungen zu beseitigen.

## Beispiel

Ein Beispiel mit diversen C-Sonderzeichen wie `%<>|#{}[]~` sowie Kommentaren zeigt die Möglichkeiten:

```
//  test.cpp (wie sonst?)
//  C++ Fragmente nur zur Demonstration von c.sty
//  (bitte keinem Compiler vorwerfen)

#include<iostream.h>
int a = 1, b = 3;
cout << a % b;
a /= b;
```

```
char array[9];
a = ~a;   //  Tilde
// Kommentar typewriter font. Ein Bruch: Z/N

int* a_pointer = &a;  //  Unterstrich und &
a < b;
//  Kommentar mit Summenzeichen: ∑ a_i

{
   double dx = 0.001;
   double Integral = 0.0;
   for(int i = 0; i < n; i++)   //  Integral: ∫_0^n f(x)dx
      Integral += f(x) * dx;
}  //  Blockende große Schrift ist kein Problem

/*  Kommentar mit /* und Funktion: f(x) = { x  für x > 0
                                          { -x sonst
neue Zeile
*/

   cout << "ÄÖÜäöü: Umlaut-Beispiel!" << endl;
   cout << "Und ein eszett: ß!" << endl;

   //  math. Formel: ∀_k ∃_{n_0} ∀_{n>n_0} (n! > n^k)
```

Wie sieht nun der dazugehörige Text aus? Hier sei die Benutzung von *c.sty* mit einem PC unter Codepage 850 gezeigt:

```
\documentclass{article}
\usepackage{german}
\usepackage[cp850]{inputenc}
\usepackage{c}               % c.sty aus diesem Artikel

\begin{document}
Laufender Text .... und jetzt das Programm

\beginprog
```

... hier folgt das Programm

```
x = 999; // Dies ist die letze Programmzeile!
\endprog
```

Hier folgt wieder Text usw.

```
\end{document}
```

## Implementierung

Abschließend sei der Source-Code von *c.sty* dargestellt.[1] Leider bräuchte es mehrere Seiten, ihn zu erklären, und da ich mehr Benutzer als Experte bin, verstehe ich selbst nicht alles genau, was ich mir da gelegentlich auch mit der Methode „Versuch und Irrtum“ zusammengebastelt habe. Vielleicht hat Jemand Interesse, die oben genannten Einschränkungen zu beseitigen, *c.sty* zu verbessern und womöglich zu beschreiben. Mir hat dieses Makropaket bisher gute Dienste geleistet.

```
% c.sty

\newif\ifCNeueZeile
\newif\ifCErsteNeueZeile

\def\beginprog{\CErsteNeueZeilefalse \frenchspacing
  \beginprogprog}
\def\endprog{\endgroup}

\def\progfile#1{\beginprogprog\input{#1}\endprog}

\def\beginprogprog{\begingroup
  \parindent=0pt\CIndent
  \CZeichen}

\def\Cdospecials{\do\{\do\}\do\$\do\&%
  \do\#\do\^\do\_\do\%\do\~\do\"\do\'}

\def\Cmakeother#1{\catcode`#1=12\relax}

\begingroup \catcode`\/=\active
```

[1] Der Autor dankt Bernd Raichle für einige Verbesserungen an dem Source-Code.

```
  \gdef\CDefineActiveSlash{%
    \catcode`\/=\active
    \def/##1{\ifx/##1\CCKommentar
      \else \ifx*##1\CSternKommentar
      \else \string/##1\fi\fi}}
\endgroup

\def\CZeichen{\tt
  \let\do=\Cmakeother \Cdospecials
  \def\par{\ifvmode \else
      \ifCErsteNeueZeile \newline\hbox{}\CIndent\fi \fi
    \CErsteNeueZeiletrue}%
  \obeylines
  \obeyspaces
  \CDefineActiveSlash
  \ifCNeueZeile \CNeueZeilefalse \par \fi}

\def\CIndent{{\tt\quad}}

\begingroup \obeylines%
  \gdef\CDefineActiveNewline#1{\let ^^M=#1\relax}%
\endgroup

\def\CCKommentar{// \endgroup
  \begingroup
    \CErsteNeueZeiletrue\CNeueZeiletrue
    \obeylines \CDefineActiveNewline\CZeichen}

\def\CSternKommentar{/* \endgroup
  \CSternKommentarZeile}

\long\def\CSternKommentarZeile#1*/{%
  \begingroup
    \CErsteNeueZeiletrue
    \CDefineActiveNewline\ %
    \def\par{\ifvmode\else \newline\hbox{}\CIndent \fi}%
    #1%
  \endgroup
  {\tt */}%
  \CNeueZeilefalse
```

```
  \beginprogprog
    \newline\hbox{}\quad}

\endinput
```

# Aus dem Fundus

## Nützliches aus CTAN und anderen Quellen

*Auf CTAN (Comprehensive TEX Archive Network) und an anderen Orten liegen Lösungen für viele TEXnische Probleme bereit. Viele davon sind kaum einem größeren Benutzerkreis bekannt. Diese Rubrik soll dazu dienen, Lösungen für diverse Probleme vorzustellen, die man in der Regel schon vorgefertigt findet oder mit wenig Aufwand nutzen kann.*

## Tafel-Fett

Gerd Neugebauer

Manchmal ist es nötig, die mathematischen Symbole für die natürlichen Zahlen, die ganzen Zahlen, die komplexen Zahlen oder ähnliche Symbole zu verwenden. Diese werden typischerweise durch einen zusätzlichen senkrechten Strich am Anfang der Letter dargestellt.

Es gibt verschiedene Lösungen für dieses Problem. Einige dieser Lösungen sollen hier vorgestellt werden.

### Einleitung

Historisch gesehen sind die Symbole für die natürlichen Zahlen usw. als fette Buchstaben entstanden. Um einen fetten Buchstaben mit Kreide auf der Tafel zu schreiben, bietet es sich an, die Linien zu verdoppeln, statt mit der Längsseite der Kreide einen breiteren Strich zu ziehen. Wohl der Schreibgeschwindigkeit wegen hat es sich zusätzlich eingebürgert, nur den ersten senkrechten Strich des Symbols zu verdoppeln, also an diesen eine zweite senkrechte Linie durch einen kleinen Abstand getrennt anzubringen.

Wenn man diese Ausführungen beherzigt, dann ist es typographisch durchaus korrekt, das Symbol für die natürlichen Zahlen einfach durch den fetten Buchstaben $\boldsymbol{N}$ darzustellen. Eine LaTeX-Definition dafür kann dann folgendermaßen aussehen:

```
\newcommand{\Nat}{\mbox{\boldmath\(N\)}}
```

Dieser Befehl kann sowohl innerhalb als auch außerhalb des mathematischen Modus verwendet werden. Dafür ist die `\mbox`-Konstruktion zuständig, die prophylaktisch in den Text-Modus wechselt. Danach werden die fetten mathematischen Schriften aktiviert und der Buchstabe N im mathematischen Modus gesetzt.

Diese Lösung hat mehrere Nachteile. Erstens ist es eine Frage der Ästethik und von Konventionen, ob ein kursives $N$ oder ein aufrechtes N gewünscht wird. Andererseits verschwindet der Unterschied von normalem und fettem Mathematiksatz, wenn man gezwungen ist, global auf die fette Variante umzuschalten. Dies kann durchaus sinnvoll sein, wenn man eine Formel aus einer Arbeit in eine Folie übernimmt und dort global fette Schriften benutzt werden.

Ein aufrechtes N können wir noch einfach erhalten. Dazu müssen wir die Definition einfach folgendermaßen abändern:

```
\newcommand{\Nat}{\mbox{\textbf{N}}}
```

Sind die Konventionen doch zu stark oder die historischen Wurzeln einfach in Vergessenheit geraten, müssen wir auf eine Lösung zurückgreifen, die der Schreibweise mit doppeltem Strich auf der Tafel näher kommt, als die oben gezeigten Lösungen.

Aus den einführenden Erläuterungen sollte jedem der gebräuchliche Name für solche Buchstaben einleuchten. Diese werden als „blackbord bold letters“ bezeichnet. Also als auf eine Tafel geschriebene, fette Buchstaben. Solchem Tafel-Fett wollen wir uns jetzt weiter nähern.

## Lösung mit den Standardschriften

Als nächste Lösung bietet es sich an, die vorhandenen Buchstaben in der gleichen Weise zu behandeln, wie dies auch auf der Tafel passiert. Also müssen wir einen zusätzlichen senkrechten Strich an dem N anbringen. Es kursieren viele Ansätze, die auf dieser Idee beruhen. Den senkrechten Strich bekommen wir von dem Buchstaben I, der den Vorteil hat, daß er auch gleich die passenden Serifen besitzt. Damit können wir unseren Befehl auf die folgende Weise abändern:

```
\newcommand{\Nat}{\ensuremath{\mathrm{I\kern-.18em N}}}
```

Das Ergebnis von `\Nat` ist $\mathrm{I\kern-.18em N}$. Der Befehl `\kern-.18em` dient dazu, die beiden Buchstaben näher aneinander zu rücken. Dabei sind schon verschiedene Größen

berücksichtigt, wie sie durch `\large`, `\small` und andere Schriftgrößenbefehle erreicht werden. Da 1 em ein horizontales, schriftgrößenabhängiges Maß ist, wird das Skalierungsproblem von TEX alleine gelöst.

Als Problem erweist sich allerdings der Wert, um den die beiden Buchstaben zusammengerückt werden müssen. Die verwendeten 0,18 em wurden durch Experimente mit den Computer-Modern-Schriften ermittelt. Für andere Grundschriften, wie bespielsweise Times, können hier durchaus andere Werte sinnvoll sein.

Ein weiteres Problem, das wir mit dieser Lösung umgangen haben, liegt darin, daß das N in Sub- oder Superscripts seine Größe geeignet verändern sollte. Dieses Verhalten wird durch die Verwendung des `\ensuremath` erreicht, anstatt einfach ein `\mbox` zu verwenden. Das Ergebnis mit einer `\mbox`-Konstruktion von `$2^{\Nat}$` ist nämlich $2^{\mbox{\mathbb{N}}}$, während wir mit unserer Lösung $2^{\mathbb{N}}$ erhalten.

Die Lösung, die wir jetzt haben, läßt sich auf alle Buchstaben übertragen, die mit senkrechten Linien anfangen. Die komplexen Zahlen $\mathbb{C}$ oder die rationalen Zahlen $\mathbb{Q}$ bekommen wir auch noch in den Griff – allerdings mit einer anderen Technik. Nachdem wir gesehen haben, daß es geht und wie das Ergebnis aussieht, kann jeder verwegene Leser versuchen, die entsprechenden Definitionen zu bauen. Aber bei den ganzen Zahlen, die mit dem Buchstaben Z bezeichnet werden, müssen wir dann doch langsam passen.

## $\mathcal{AMS}$-Schriften

Sehen wir uns also an, ob wir geeignete Schriften finden, die die gewünschten Symbole enthalten. Die Verwendung von solchen Schriften wäre ja von Anfang an die beste Lösung gewesen. Nur waren solche früher einfach nicht vorhanden.

Die bekanntesten Schriften, die Blackbord-Bold-Zeichen zur Verfügung stellen, sind in den Schriften der American Mathematical Society enthalten. Diese $\mathcal{AMS}$-Schriften sind auf dem TEX-Archiv CTAN im Verzeichnis `tex-archive/fonts/ams/amsfonts` zu finden. Dort finden wir, neben den Schriften selbst, auch gleich das Paket `amssymb`, das uns die Eingabe sehr vereinfacht.[1]

Da die $\mathcal{AMS}$-Schriften zu jeder guten TEX-Installation gehören und deren Verwendung bereits ausführlich beschrieben wurde [1, 2, 3], können wir uns hier kurz fassen. In die Präambel kommt die Deklaration

[1] Vorsicht ist bei der Version dieses Paketes geboten. Es gibt auch andere Pakete, die `amssymb` enthalten. Da ist es ratsam, ein wachsames Auge auf die Protokolldatei zu haben, um sicher zu gehen, daß auch die richtige Version genommen wird.

```
\usepackage{amssymb}
```

Danach steht im mathematischen Modus der Schriftumschaltbefehl `\mathbb` zur Verfügung, mit der wir unsere Definition folgendermaßen abändern können:

```
\newcommand{\Nat}{\ensuremath{\mathbb{N}}}
```

In der folgenden Liste wurden alle Blackboard-Bold-Zeichen, die in den $\mathcal{AMS}$-Schriften vorhanden sind, zusammengestellt.

$\mathbb{A}\ \mathbb{B}\ \mathbb{C}\ \mathbb{D}\ \mathbb{E}\ \mathbb{F}\ \mathbb{G}\ \mathbb{H}\ \mathbb{I}\ \mathbb{J}\ \mathbb{K}\ \mathbb{L}\ \mathbb{M}\ \mathbb{N}\ \mathbb{O}\ \mathbb{P}\ \mathbb{Q}\ \mathbb{R}\ \mathbb{S}\ \mathbb{T}\ \mathbb{U}\ \mathbb{V}\ \mathbb{W}\ \mathbb{X}\ \mathbb{Y}\ \mathbb{Z}\ \Bbbk$

Wie wir sehen, sind alle Großbuchstaben und der Kleinbuchstabe k vorhanden. Dabei ist zu bemerken, daß das $\Bbbk$ nicht etwa durch `\(\mathbb{k}\)` angesprochen werden kann (das ergäbe nämlich ך, den hebräischen Buchstaben Daleth), sondern durch `\(\Bbbk\)`.

Ein kleines Problem habe ich mit dem Aussehen der Buchstaben. Diese sehen eher nach einer ausgehöhlten Fassung der Computer-Modern-Zeichen aus als nach der gewohnten Blackboard-Bold-Schrift. Aber das ist natürlich Geschmackssache. Aus diesem Grund sehen wir uns nach weiteren Alternativen um. Danach können wir immer noch abwägen, welche Lösung uns am besten gefällt.

## Serifenlose Variante

Als nächstes kommen wir zu einer Schrift von Alan Jeffrey. Dieser hat mit `bbold` eine Schrift vorgelegt, die fast alle Buchstaben in einer Blackboard-Bold-artigen Variante enthält. Dazu gehören nicht nur die Groß- und Kleinbuchstaben, sondern auch Ziffern und Sonderzeichen. Dadurch, daß eine Nicht-Standard-Kodierung verwendet wurde, war es sogar möglich, die griechischen Buchstaben in dem Font unterzubringen. Den gesamten Zeichensatz sehen wir in der Tabelle 1.

Wenn wir diese Schrift verwenden wollen, müssen wir erst einmal die METAFONT-Quellen übersetzen und zusammen mit den entstehenden tfm- und pk-Dateien an die geeignete Stelle kopieren. Das Verfahren wurde schon in der letzten Folge dieser Rubrik beschrieben [4].

Nun kommen wir dazu, diese Schrift zu verwenden. Zum Glück hat Alan Jeffrey bereits eine Font-Definitionsdatei `Ubbold.fd` und ein Paket `bbold.sty` für LaTeX bereitgestellt. Leider kollidieren die dann verwendeten Makronamen mit den Namen, die bereits von den $\mathcal{AMS}$-Schriften benutzt werden.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 𝛤 | 𝛥 | 𝛩 | 𝛬 | 𝛯 | 𝛱 | 𝛴 | 𝛶 | 𝛷 | 𝛹 | 𝛺 | 𝛼 | 𝛽 | 𝛾 | 𝛿 | 𝜀 |
| 16 | 𝜁 | 𝜂 | 𝜃 | 𝜄 | 𝜅 | 𝜆 | 𝜇 | 𝜈 | 𝜉 | 𝜋 | 𝜌 | 𝜎 | 𝜏 | 𝜐 | 𝜙 | 𝜒 |
| 32 | 𝜓 | ! | ” | # | $ | % | & | ’ | ( | ) | * | + | , | - | . | / |
| 48 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 64 | @ | 𝔸 | 𝔹 | ℂ | 𝔻 | 𝔼 | 𝔽 | 𝔾 | ℍ | 𝕀 | 𝕁 | 𝕂 | 𝕃 | 𝕄 | ℕ | 𝕆 |
| 80 | ℙ | ℚ | ℝ | 𝕊 | 𝕋 | 𝕌 | 𝕍 | 𝕎 | 𝕏 | 𝕐 | ℤ | [ | \ | ] | ^ | _ |
| 96 | ` | 𝕒 | 𝕓 | 𝕔 | 𝕕 | 𝕖 | 𝕗 | 𝕘 | 𝕙 | 𝕚 | 𝕛 | 𝕜 | 𝕝 | 𝕞 | 𝕟 | 𝕠 |
| 112 | 𝕡 | 𝕢 | 𝕣 | 𝕤 | 𝕥 | 𝕦 | 𝕧 | 𝕨 | 𝕩 | 𝕪 | 𝕫 | { | \| | } | ~ | 𝟊 |

Tabelle 1: Der Font `bbold`

Das ist Absicht, da es damit möglich ist, die $\mathcal{AMS}$-Definitionen durch neue zu ersetzen, die den `bbold`-Zeichensatz benutzen. Das Paket `bbold.sty` ist dazu einfach *nach* den entsprechenden $\mathcal{AMS}$-Paketen zu laden.

Da ich aber hier beide Varianten simultan verwenden wollte, mußte ich mir etwas einfallen lassen. Dazu habe ich eine neue Datei `bbbold.sty` angelegt, in der eigene Schriftumschaltbefehle definiert werden. Hier ist wieder der relevante Ausschnitt zu sehen:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{bbbold}[1996/04/04 Bbbold symbol package]
\newcommand{\bbbfamily}{%
  \fontencoding{U}\fontfamily{bbold}\selectfont}
\newcommand{\textbbb}[1]{{\bbbfamily#1}}
\DeclareMathAlphabet{\mathbbb}{U}{bbold}{m}{n}
```

Diese neue Datei muß an eine Stelle kopiert werden, wo TEX sie finden kann. In einer TDS-konformen Installation [5] ist das etwa das Verzeichnis

*texmf*`/tex/latex/bbold`

Nun können wir in der Präambel unseres Dokumentes das Paket folgendermaßen einbinden:

```
\usepackage{bbbold}
```

Das Paket stellt einen Schriftumschaltbefehl und zwei Befehle zum Setzen eines kurzen Textes bereit. Als erstes ist da der Befehl `\mathbbb`, der im mathematischen Modus dazu verwendet werden muß. Das Argument dieses Befehls wird in der Schrift `bbold` gesetzt. So liefert beispielsweise `$\mathbbb{N}$` das Ergebnis $\mathbbb{N}$.

Analog dazu gibt es den Befehl `\textbbb`, der dasselbe im Text-Modus leistet. Schließlich gibt es noch den Schriftumschaltbefehl `\bbbfamily`, der dazu genutzt werden kann, längere Textpassagen mit der Schrift `bbold` zu setzen.

Es bleibt nur noch zu klären, wie wir an die griechischen Buchstaben herankommen. Um die Erklärungen zum oktalen Zahlensystem (Basis 8), oder dem hexadezimalen (oder sedezimalen) Zahlensystem (Basis 16) zu ersparen, ist die Schrifttabelle mit dezimalen Zahlen beschriftet.

Um ein Zeichen aus der Tabelle verwenden zu können, addieren wir die Zeilen- und die Spaltenwerte und verwenden den Befehl `\symbol`, um das Zeichen anzusprechen. Beispielsweise steht das kleine pi ($\pi$) in der Spalte mit der Nummer 9 und der Zeile mit der Nummer 16. Folglich erhalten wir $\mathbb{\pi}$ im mathematischen Modus durch den Befehl

```
\mbox{\textbbb{\symbol{25}}}
```

Dieser Befehl wechselt in den Textmodus und setzt dort das Zeichen mit der Nummer 25 aus der richtigen Familie. Das funktioniert in dieser Weise für die Buchstaben und Zahlen ganz gut. Aber spätestens bei Operatoren oder Klammern ist das Ergebnis im mathematischen Modus nicht mehr befriedigend. So ergibt die Eingabe `\(\mathbbb{(1+1=2)}\)` das Ergebnis ($\mathbb{(1+1=2)}$). Hierbei werden die Klammern sowie das Plus- und das Gleichheitszeichen nicht aus der `bbold`-Schrift genommen, sondern aus den üblicherweise verwendeten Computer-Modern-Schriften.

Dies liegt daran, daß TEX die Klammern und die mathematischen Operatoren speziell behandelt. Wenn wir solche Zeichen verwenden wollen, bietet es sich an, dafür neue mathematische Symbole zu definieren. Das geschieht mit dem Befehl `\DeclareMathSymbol` in der Präambel des Dokuments. Betrachten wir dazu den folgenden Ausschnitt von Definitionen:

```
\DeclareMathSymbol{\bbOpen}{\mathopen}{bbold}{91}
\DeclareMathSymbol{\bbClose}{\mathclose}{bbold}{93}
\DeclareMathSymbol{\bbPlus}{\mathbin}{bbold}{43}
\DeclareMathSymbol{\bbGT}{\mathrel}{bbold}{62}
```

Dabei wird `\bbOpen` als öffnende Klammer, `\bbClose` als schließende Klammer, `\bbPlus` als binärer Operator und `\bbGT` als mathematische Relation definiert. Die entsprechenden Zeichen werden aus dem mathematischen Font `bbold` genommen, den wir in dem Paket `bbbold` definiert haben. Damit können wir dann

```
\[ A \bbPlus B \bbGT \bbOpen a_1,\ldots, a_n \bbClose \]
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| \bbalpha | 𝛂 | \bbeta | 𝛃 | \bbnu | 𝛎 | \bbupsilon | 𝛖 |
| \bbbeta | 𝛃 | \bbtheta | 𝛉 | \bbxi | 𝛏 | \bbphi | 𝛗 |
| \bbgamma | 𝛄 | \bbiota | 𝛊 | \bbpi | 𝛑 | \bbchi | 𝛘 |
| \bbdelta | 𝛅 | \bbkappa | 𝛋 | \bbrho | 𝛒 | \bbpsi | 𝛙 |
| \bbepsilon | 𝛆 | \bblambda | 𝛌 | \bbsigma | 𝛔 | \bbomega | 𝛚 |
| \bbzeta | 𝛇 | \bbmu | 𝛍 | \bbtau | 𝛕 | | |

Tabelle 2: Die Befehle für die griechischen Buchstaben von `mathbbol`

schreiben und erhalten daraus

$$A \dotplus B \gg [\![a_1, \ldots, a_n]\!]$$

Für die anderen Zeichen aus dieser Schrift können wir nach dem gleichen Muster Definitionen erstellen und diese danach recht einfach verwenden.

Bevor ich vergesse, die Stelle im TEX-Archiv zu nennen, an der die `bbold`-Schrift und die zugehörigen Dateien zu finden sind: `tex-archive/fonts/bbold`.

Es gibt von Jörg Knappen noch ein weiteres Paket namens `mathbbol`, das diese Schriften verwendet. Durch dieses Paket werden die entsprechenden Befehle aus dem $\mathcal{AMS}$-Paket umdefiniert, um danach die `bbold`-Schrift zu verwenden.

Das Paket `mathbbol` kennt unter anderem auch die Option `bbgreekl`, die wie folgt angegeben werden kann:

```
\usepackage[bbgreekl]{mathbbol}
```

Danach stehen die griechischen Buchstaben unter symbolischen Namen zur Verfügung. So kann das $\pi$ im mathematischen Modus mit dem Befehl `\bbpi` angesprochen werden. Die komplette Liste der bereitgestellten Befehle ist in der Tabelle 2 zu sehen.

Wie wir sehen, sind nur die griechischen Kleinbuchstaben auf diese Weise erreichbar. Die entsprechenden Definitionen für die Großbuchstaben und die anderen Sonderzeichen sind nicht vorhanden.

Leider läßt die Dokumentation dieses Paketes zu wünschen übrig. Aber vielleicht braucht man ja keine Dokumentation, wenn man einfach die entsprechenden Befehle aus dem $\mathcal{AMS}$-Paket umdefiniert haben will und ansonsten die obige Tabelle benutzt.

Das Paket `mathbbol` ist im TEX-Archiv im Verzeichnis `tex-archive/macros/latex/contrib/supported/jknappen` zu finden.

## Doppelstreich

Nun wollen wir uns die nächste Schrift ansehen. Sie heißt `doublestroke` und stammt von Olaf Kummer. Im Gegensatz zu `bbold` enthält sie nur die Großbuchstaben, die aber in einer Form ausgeführt sind, die zu Computer-Modern-Roman paßt und meinen Vorstellungen von Blackbord-Bold-Zeichen ziemlich nahekommt:

𝔸 𝔹 ℂ 𝔻 𝔼 𝔽 𝔾 ℍ 𝕀 𝕁 𝕂 𝕃 𝕄 ℕ 𝕆 ℙ ℚ ℝ 𝕊 𝕋 𝕌 𝕍 𝕎 𝕏 𝕐 ℤ

Die Schrift wird von einer Font-Definitionsdatei (`Udsrom.fd`) und einem Paket (`dsfont.sty`) begleitet. Damit ist es recht einfach, die Schrift zu verwenden. Die Installation der Schrift und der fd- und sty-Dateien geschieht nach dem Schema, das im vorangegangenen Abschnitt beschrieben wurde, ohne daß etwas anzupassen wäre.

Jetzt können wir in der Präambel das Paket mit dem folgenden Befehl ansprechen:

```
\usepackage{dsfont}
```

Es stellt einen einzigen Befehl, `\mathds`, bereit, der nur im mathematischen Modus verfügbar ist. Er nimmt ein Argument und setzt es in dem `doublestroke`-Font. Also schreiben wir `$\mathds{N}$` und erhalten $\mathbb{N}$.

Zu finden ist die `doublestroke`-Schrift und die zugehörigen Dateien im TEX-Archiv in dem Verzeichnis `tex-archive/fonts/doublestroke`.

## Computer Modern Blackboard

Als letztes wollen wir uns die Schriftfamilie `bbm` von Gilles F. Robert ansehen. Im Gegensatz zu den vorangegangenen Lösungen handelt es sich hierbei tatsächlich um mehrere zusammenpassende Schriften. Nicht nur, daß eine zu den Computer-Modern-Schriften passende Schrift vorhanden ist, sondern es gibt zusätzlich eine serifenlose und eine Schreibmaschinenschrift. Die beiden erstgenannten Schnitte sind zusätzlich noch in jeweils einer fetten Variante vorhanden.

Die Schriften enthalten jeweils die Groß- und Kleinbuchstaben, runde und eckige Klammern, sowie die Ziffern 1 und 2. Alle vorhandenen Zeichen können wir in der Tabelle 3 bewundern.

Diese Schriften sind im TEX-Archiv in dem Verzeichnis `tex-archive/fonts/cm/bbm` zu finden.

| | |
|---|---|
| `\mathbbm{`$x$`}` | ABCDEFGHIJKLMNOPQRSTUVWXYZ<br>abcdefghijklmnopqrstuvwxyz12()[] |
| `\boldmath\mathbbm{`$x$`}` | **ABCDEFGHIJKLMNOPQRSTUVW**<br>**XYZabcdefghijklmnopqrstuvwxyz12()[]** |
| `\mathbbmss{`$x$`}` | ABCDEFGHIJKLMNOPQRSTUVWXYZ<br>abcdefghijklmnopqrstuvwxyz12()[] |
| `\boldmath\mathbbmss{`$x$`}` | **ABCDEFGHIJKLMNOPQRSTUVWXYZ**<br>**abcdefghijklmnopqrstuvwxyz12()[]** |
| `\mathbbmtt{`$x$`}` | ABCDEFGHIJKLMNOPQRSTUVWXYZ<br>abcdefghijklmnopqrstuvwxyz12()[] |

Tabelle 3: Computer-Modern-Blackboard

Dort befinden sich aber nur die nackten METAFONT-Quellen. Zum Glück hat sich mit Torsten Hilbrich jemand gefunden, der dazu auch die passenden Font-Definitionen und ein Paket namens `bbm` für LaTeX geschrieben hat.

Nach dem Laden des Paketes sind im mathematischen Modus drei neue Befehle vorhanden: `\mathbbm`, `\mathbbmss` und `\mathbbmtt`. Die Befehle haben jeweils ein Argument und setzen dieses mit der entsprechenden Schrift. Die fette Variante kann über die `\boldmath`-Deklaration angesprochen werden.

Das heißt, daß wir normalerweise, wenn `\unboldmath` gesetzt ist, mit der Befehlsfolge `\(\mathbbm{N}\)` ein $\mathbb{N}$ erzeugen können. Wenn jedoch `\boldmath` aktiviert wurde, ergibt sich ein $\boldsymbol{\mathbb{N}}$.

Wie wir bereits weiter oben gesehen haben, müssen wir die Klammern speziell behandeln, indem wir neue Befehle für diese mathematischen Symbole definieren. Das geschieht wieder nach dem gleichen Schema und muß deshalb nicht mehr weiter erläutert werden.

Ansonsten ist über das Paket `bbm` nicht mehr viel zu sagen. Man muß keine Anpassungen vornehmen und kaum etwas beachten. Es funktioniert einfach wie es soll und ist im TeX-Archiv in dem Verzeichnis `tex-archive/macros/latex/contrib/supported/bbm` zu finden.

## Schlußbemerkung

Damit haben wir genügend Varianten gesehen, so daß die Auswahl schon anfängt, schwer zu werden. Ich hoffe jedoch, daß jeder, der es braucht, etwas

Passendes darunter gefunden hat. Wenn nicht, dann findet sich vielleicht bald im TEX-Archiv eine weitere Alternative.

## Nachtrag

Eigentlich hätte ich es gleich am Anfang erwähnen sollen. Es gibt nämlich eine Quelle für Informationen, wie die hier vorgestellten. Diese Quelle ist die sogenannte DE-TEX-/DANTE-FAQ, die von Bernd Raichle und Thomas Hafner geführt wird. Sie ist im CTAN in dem Verzeichnis `tex-archive/usergrps/dante/de-tex-faq/` zu finden. Dort werden auch Hinweise zu immer wieder auftretenden Problemen gegeben.

Im Gegensatz zu dieser Rubrik geschieht das allerdings nur im Telegrammstil. In dieser Rubrik kann von allen Möglichkeiten, die LATEX bietet, Gebrauch gemacht werden. Damit bekommt man nicht nur den Hinweis, daß es eine Schrift oder ein Paket im CTAN gibt, sondern kann auch gleich an Beispielen sehen, wie es wirkt.

Wer also ein Problem hat, der sollte auf jeden Fall die DANTE-FAQ zu Rate ziehen, muß allerdings selbst ausprobieren, wie die Lösungen aussehen.

## Literatur

[1] $\mathcal{AMS}$; *User's Guide to AMSFonts Version 2.2*; Jan. 1995.

[2] Michel Goosens, Frank Mittelbach und Alexander Samarin: *The LATEX Companion*; Addison-Wesley Publishing Company; Reading, Mass.; 1994.

[3] Helmut Kopka: *LATEX-Ergänzungen – mit einer Einführung in METAFONT*; Addison-Wesley Publishing Company; Bonn; 1995.

[4] Gerd Neugebauer: *Von „krakelig" bis „wie gemalt"*; *Die TEXnische Komödie*; 96/1, S. 25–42; 1996.

[5] TUG Working Group on a TEX Directory Structure: *A Directory Structure for TEX Files*; Draft version 0.104; Nov. 1995.

# Was Sie schon immer über TEX wissen wollten . . .

## Roman statt Italic in mathematischen Formeln

Bernd Raichle

Folgende Anfrage aus `de.comp.tex` steht stellvertretend für weitere Fragen, die immer wieder im Zusammenhang mit Schriften im mathematischen Formelsatz unter LaTeX $2_\varepsilon$ gestellt werden:

> Ich versuche die für Buchstaben übliche Math-Italics-Schrift im mathematischen Formelsatz durch die normale Text-Roman-Schrift zu ersetzen, um damit einfacher chemische Summenformeln setzen zu können, ohne jedesmal `\mathrm{...}` eingeben zu müssen.
>
> Bisher habe ich dies nur mit
> `\SetSymbolFont{letters}{normal}{OT1}{cmr}{m}{n}`
>
> erreichen können.

Leider ist diese `\SetSymbolFont`-Anweisung keine wirklich gute Lösung. Man sollte das Encoding von Symbol-Fonts möglichst *nicht* ändern, da dies sehr viele weitere Änderungen nach sich zieht

Der Symbol-Font `letters` ist in LaTeX $2_\varepsilon$ durch

```
\DeclareSymbolFont{letters}     {OML}{cmm} {m}{it}
\SetSymbolFont{letters}   {bold}{OML}{cmm} {b}{it}
```

definiert, ist also `OML`-kodiert. Die obige `\SetSymbolFont`-Anweisung ändert die Kodierung jedoch auf `OT1`. Da aus dem Symbol-Font `letters` neben den Buchstaben unter anderem auch die kleinen griechischen Buchstaben, das Kleiner- und das Größer-Zeichen verwendet werden und diese Zeichen nicht in einer `OT1`-kodierten Schrift enthalten sind, bekommt man somit falsche Zeichen.

Deshalb ist es besser, wenn man einen neuen Symbol-Font definiert, den ich `rletters` für Roman-Letters nenne. Mit

```
\DeclareSymbolFont{rletters}    {OT1}{cmr} {m}{n}
\SetSymbolFont{rletters}  {bold}{OT1}{cmr} {bx}{n}
```

definiert man die normale und die fette Math-Version dieses Symbol-Font.

Anschließend werden einfach nur die mathematischen Zeichen neu deklariert, die man geändert haben will. Für die 26 Klein- und 26 Großbuchstaben sind damit die folgenden 52 Deklarationen notwendig, wobei Sie die ausgelassenen 44 Zeilen sicher ohne Probleme selbst ergänzen können:

```
\DeclareMathSymbol{a}{\mathalpha}{rletters}{`a}
\DeclareMathSymbol{b}{\mathalpha}{rletters}{`b}
\DeclareMathSymbol{c}{\mathalpha}{rletters}{`c}
  ...
\DeclareMathSymbol{z}{\mathalpha}{rletters}{`z}
\DeclareMathSymbol{A}{\mathalpha}{rletters}{`A}
\DeclareMathSymbol{B}{\mathalpha}{rletters}{`B}
\DeclareMathSymbol{C}{\mathalpha}{rletters}{`C}
  ...
\DeclareMathSymbol{Z}{\mathalpha}{rletters}{`Z}
```

Wenn Sie wollen, können Sie auch das Komma, den Punkt und den Schrägstrich, die in `OT1`-kodierten Schriften auch vorhanden sind, aus diesem neuen Symbol-Font nehmen. Dazu sind noch die folgenden drei Zeilen notwendig:

```
\DeclareMathSymbol{,}{\mathpunct}{rletters}{`,}
\DeclareMathSymbol{.}{\mathord}{rletters}{`.}
\DeclareMathSymbol{/}{\mathord}{rletters}{`/}
```

Die Standard-Deklarationen, die LaTeX 2$\varepsilon$ ohne diese Änderungen verwendet, finden Sie in der Datei `fontdef.dtx` bzw. den beiden ausgepackten Dateien `fonttext.ltx` und `fontmath.ltx`. Neben der Dokumentation zum Schriftauswahlschema [1] enthält diese Datei noch weitere Erläuterungen und Hinweise für eigene Änderungen.

## Literatur

[1] LaTeX3 Project Team: *LaTeX 2$\varepsilon$ font selection*; als Datei `fntguide.tex` in jeder LaTeX 2$\varepsilon$-Verteilung enthalten.

# TEX-Beiprogramm

## TEX Live – Die erste TDS-konforme ready-to-run TEX-CD-ROM für Unix-Systeme und andere Plattformen

Ulrik Vieth

*Fight ISO! Go out on the street! Come to a rally!*
*Make a usable TEX CD ... :-)*

— JOACHIM SCHROD, *TWG-TDS* (1996/02/21)

*I have! I used it at home. It all works!*

— SEBASTIAN RAHTZ, *TWG-TDS* (1996/02/22)

### Die Vorgeschichte: Der lange Weg zu einer TDS-CD-ROM

Als vor längerer Zeit (Ende 1993) erste Überlegungen zur Entwicklung einer systemübergreifenden TEX-Verteilung auf CD-ROM für eine Vielzahl verschiedener Betriebssysteme getätigt wurden [7], stellte sich schon bald heraus, daß zunächst einige Probleme zu lösen waren, um den Weg zu einer einheitlichen und vielseitig verwendbaren TEX-Verteilung freizumachen. Obwohl zwar TEX-Implementierungen für fast alle nur denkbaren Systeme existierten, unterschieden sich die einzelnen Verteilungen erheblich in ihrer Organisation, insbesondere hinsichtlich ihrer Verzeichnis-Struktur, was die Zusammenfassung mehrerer ausgepackter Verteilungen für verschiedene Systeme auf einer einzigen CD-ROM zunächst praktisch unmöglich machte.

Auf der einen Seite führten diese Überlegungen Mitte 1994 zur Gründung der „TUG Technical Working Group on a TEX Directory Structure“ (TWG-TDS), über deren Ziele und Ergebnisse bereits an anderer Stelle berichtet wurde [6]. Auf der anderen Seite entstand etwa zur gleichen Zeit in einem Alleingang der niederländischen TEX-Benutzergruppe NTG die recht bekannte und wohl auch erfolgreiche CD-ROM „4allTEX“, welche auf der `emTeX`-Verteilung beruhte und

sich von vornherein praktisch ausschließlich an die Benutzer einer einzelnen Plattform, nämlich von MS-DOS- bzw. OS/2-Systemen richtete. Für Benutzer aller anderen Systeme war diese CD-ROM dagegen fast völlig unbrauchbar und somit noch recht weit vom ursprünglichen Ziel entfernt.

Mit der zunehmenden Verbreitung von billigen und schnellen CD-ROM-Laufwerken, die mittlerweile schon zur Standardausstattung moderner PCs gehören, entstanden in der Folgezeit dann eine Reihe weiterer TEX-CDs verschiedener kommerzieller Anbieter, bei denen es sich häufig einfach um mehr oder weniger vollständige Abzüge der CTAN-Server handelt. Hierzu zählt beispielsweise auch die Anfang 1996 von DANTE e.V. in Zusammenarbeit mit dem Verlag Addison-Wesley herausgegebene CD-ROM „CTAN/3“, die bereits in einer früheren Ausgabe vorgestellt wurde [5].

Natürlich haben diese CTAN-Abzüge auf CD-ROM ihre Berechtigung, sie haben aber einen entscheidenden Nachteil gegenüber sogenannten „ready-to-run“ Verteilungen für bestimmte Betriebssysteme: Da die CTAN-Abzüge in der Regel nur gepackte Archive enthalten, erfordern sie immer noch einen nicht unerheblichen Verwaltungsaufwand, wenn man nach der Erstinstallation einer mehr oder weniger umfassenden Basis-Verteilung noch verschiedene andere Makro- oder Font-Pakete nachinstallieren möchte, die man jeweils erst auspacken, gegebenenfalls bearbeiten und an den richtigen Stellen im jeweiligen System installieren muß. Bei einer „ready-to-run“ Verteilung sind derartige Aufgaben dagegen bereits erledigt, so daß man im günstigsten Fall direkt von der eingelegten CD-ROM aus arbeiten oder einfach die vorbereiteten Pakete auf seine Festplatte kopieren kann.

Mit der kürzlich erschienenen CD-ROM „TEX Live“ wird nun erstmals auch eine „ready-to-run“ TEX-CD-ROM speziell für Unix-Systeme angeboten, die auf der `teTeX`-Verteilung beruht. Diese CD-ROM verfolgt gewissermaßen ein doppeltes Ziel: einerseits als Unix-TEX-CD-ROM mit lauffähigen Programmen für eine größere Zahl gängiger Unix-Plattformen, andererseits als TDS-CD-ROM mit einem reichhaltigen Angebot von Makro- und Font-Paketen, die einschließlich der dazugehörigen Dokumentation vollständig TDS-konform, d. h. entsprechend den Empfehlungen der TWG-TDS organisiert sind, und somit nicht nur unter Unix, sondern auf praktisch allen gängigen Systemen verwendbar sein sollten.

Mit der CD-ROM „TEX Live“ wurde somit nach fast $2\,{}^1\!/_2$ Jahren endlich das Ziel einer einheitlichen, systemübergreifenden TEX-Verteilung erreicht, das am Anfang der Entwicklung gestanden hatte. Zugleich wurde mit dieser ersten TDS-CD-ROM auch eine schon häufiger kritisierte, noch bestehende Marktlücke im bisherigen Sortiment von TEX-CD-ROMs geschlossen.

## Die Entstehung der CD-ROM „TEX Live“

Bereits zu Beginn der Arbeit der TWG-TDS war klar, daß zu den Zielen der Gruppe neben der Ausarbeitung von Empfehlungen für die Organisation von TEX-Installationen auch die Konzeption von TEX-Verteilungen auf CD-ROM gehörte. Nachdem dieses Ziel über längere Zeit bis zur Verabschiedung der vorläufig endgültigen Fassung des „TDS-Standard“ (Version 0.999) [1, 2] in den Hintergrund gerückt war, wurde schließlich im Februar 1996 auf Initiative von Sebastian Rahtz ein neuer Anlauf zu einer „TDS/Unix-TEX-CD-ROM“ ins Leben gerufen [8], wobei von vornherein als Fertigstellungstermin die Tagung der französischsprachigen Anwendervereinigung GUTenberg über das Thema TEX-Verteilungen Ende Mai 1996 angepeilt wurde.

Als Ausgangspunkt für dieses Projekt wurde die bewährte `teTeX`-Verteilung von Thomas Esser gewählt, da diese bereits für eine Vielzahl gängiger Unix-Systeme als Binary-Verteilung vorlag und darin auch bereits die aktuellsten TDS-Empfehlungen berücksichtigt waren. Auf die Aufnahme zusätzlicher Komponenten wie beispielsweise METAPOST oder Neuentwicklungen wie ε-TEX und Ω wurde nach anfänglicher Diskussion jedoch schließlich verzichtet, da die dafür erforderliche Neu-Übersetzung aller Programme für alle unterstützten Plattformen innerhalb des vorgegebenen Zeitrahmens zu große organisatorische Probleme bereitet hätte. Derartige Erweiterungen wurden deshalb zunächst zurückgestellt, werden aber für eine zukünftige zweite Auflage weiterhin im Auge behalten.[1]

Nachdem über die Frage des Umfangs der CD-ROM entschieden war und dabei weitgehend auf Experimente verzichtet wurde, war nun zur Realisierung des Projekts vor allem Fleißarbeit gefragt, um sämtliche vorgesehenen Makro- und Font-Pakete des `texmf`-Verzeichnis-Baums (darunter beispielsweise auch alle auf CTAN verfügbaren `latex/contrib`-Pakete) TDS-konform zusammenzustellen. Im wesentlichen wurde diese Arbeit von Sebastian Rahtz als Koordinator sowie einigen Helfern aus den Reihen der TWG-TDS übernommen. Des weiteren wurden von Thomas Esser neue Installationsskripten entwickelt, um einerseits eine bequeme menügesteuerte Installation ausgewählter Pakete auf Festplatte und andererseits den direkten Betrieb von der eingelegten CD-ROM zu ermöglichen.

[1] Es ist beabsichtigt, in Zukunft in regelmäßigen Abständen neue aktualisierte Auflagen der CD-ROM „TEX Live“ herauszubringen, wobei die nächste Auflage nach Möglichkeit bereits auf Version 7.0 der `web2c`-Verteilung von Karl Berry beruhen soll, die sich schon seit längerem in einem Pre-Test befindet. Darin werden dann bereits METAPOST sowie ML-TEX und voraussichtlich auch ε-TEX als neue Komponenten enthalten sein.

Im Hinblick auf die beabsichtigte Ausrichtung der CD-ROM „TEX Live“ auf Unix-Systeme sollte sie unter Verwendung von „Rock Ridge Extensions“ gefertigt werden, welche es ermöglichen, trotz der technischen Beschränkungen des ISO-9660-CD-ROM-Dateisystems auch längere Dateinamen mit gemischter Groß- und Kleinschreibung zu verwenden, wie man dies von Unix-Systemen gewohnt ist. Dies bietet zwar einerseits den Vorteil, die Namen aller Dateien unverfälscht übernehmen zu können, erfordert aber andererseits auch gewisse Vorsichtsmaßnahmen, wenn man sicherstellen möchte, daß die TDS-Kompatibilität des `texmf`-Baums nicht nur unter Unix, sondern auch auf anderen Systemen bei Verkürzung der Dateinamen auf 8+3 Zeichen gewahrt bleibt, was natürlich insbesondere die Eindeutigkeit der verkürzten Dateinamen voraussetzt.

Bei den allermeisten Makro- und Font-Paketen ergaben sich in dieser Hinsicht keine Probleme, jedenfalls wenn man voraussetzt, daß TEX-Implementierungen auf ISO-9660-Systemen in der Lage sein sollten, gegebenenfalls selbständig nach verkürzten Dateinamen zu suchen, wenn der Name einer gesuchten Datei über $8 + 3$ Zeichen hinausgeht. Lediglich bei der Konvertierung von Texinfo-Dokumentation ins HTML- bzw. GNU-Info-Format ergaben sich hier einige Komplikationen, die zunächst zu kontroversen Diskussionen führten, inwieweit eine Unix-TEX-CD-ROM auf diese Problematik überhaupt Rücksicht nehmen sollte. Schließlich konnte jedoch auch hierfür eine Lösung zur allseitigen Zufriedenheit gefunden werden, so daß der Produktion einer TDS-konformen CD-ROM mit „Rock Ridge Extensions“ (für die Pakete mit nicht TDS-konformen Dateinamen) letztendlich nichts mehr im Wege stand.

Nachdem alle wichtigen Fragen bereits in den ersten Wochen des Projekts in Abstimmung mit der TWG-TDS geklärt wurden, konnten die Arbeiten wie geplant ihren Lauf nehmen und die CD-ROM schließlich Mitte Mai 1996 in Produktion gegeben und rechtzeitig zur GUTenberg-Tagung über TEX-Verteilungen Ende Mai 1996 fertiggestellt werden, wo sie erstmals der Öffentlichkeit präsentiert wurde [4].

## Erfahrungen mit der CD-ROM „TEX Live“

Das Ergebnis dieses Projekts kann sich wohl sehen lassen. Die CD-ROM „TEX Live“ ist vollgepackt mit fast 630 MB TEX-Software, darunter fast 50 MB für die eigentliche `teTeX`-Verteilung, 50 MB für die GUTenberg-Verteilungen für DOS-, Windows- und Macintosh-Systeme, etwa 130 MB für die direkt von der CD-ROM aus lauffähigen `teTeX`-Binaries für insgesamt 23 Unix-Plattformen[2]

[2] Bei dieser Zahl von Plattformen ist zu berücksichtigen, daß verschiedene Betriebssystem-Versionen oder Binary-Formate für dieselbe Hardware-Architektur separat gezählt werden,

sowie fast 385 MB für den systemunabhängigen `texmf`-Verzeichnis-Baum, welcher sich in insgesamt 310 separat installierbare Makro-, Font- und Dokumentations-Pakete untergliedert, auf die im folgenden noch genauer eingegangen wird.

Von diesen 385 MB des `texmf`-Verzeichnis-Baums entfallen wiederum nahezu 125 MB auf Fonts, darunter allein fast 65 MB für PK-Dateien in verschiedenen Druckerauflösungen. Ferner entfallen fast 185 MB auf Dokumentation, darunter auch 130 MB für Dokumentation im PDF-Format (Acrobat-Format), welche derzeit mangels geeigneter PDF-Viewer (Acrobat-Reader) leider noch nicht auf allen Plattformen verwendbar ist. Die übrige Dokumentation liegt zum überwiegenden Teil im DVI- oder PS-Format vor, ergänzt durch einen derzeit noch relativ geringen Anteil an HTML-Dokumentation sowie GNU-Info-Dateien und Unix-Manual-Pages.

Die Erschließung der vorhandenen Dokumentation durch HTML-Indexseiten, welche es ermöglichen, aus einem WWW-Browser heraus auf Dokumente gleich welchen Formats bequem zuzugreifen, wurde zwar begonnen, ist aber leider noch etwas unvollständig geblieben. Es ist jedoch zu hoffen, daß sich dies in einer künftigen Neuauflage verbessern läßt.

Insgesamt kann man festhalten, daß versucht wurde, einen beachtlichen Teil des derzeitigen Angebots der auf CTAN-Servern verfügbaren TEX-Software im Rahmen der TDS-Struktur zu integrieren. Hierzu gehören natürlich selbstverständlich die zum Zeitpunkt der Erstellung aktuelle LATEX-Version (vom Dezember 1995) mit allen verfügbaren LATEX-Contrib-Paketen, daneben aber auch eine recht großzügige Auswahl an METAFONT-Zeichensätzen und PostScript-Font-Metriken, wie auch einige etwas exotische Makro-Pakete, die den meisten Lesern vielleicht gar nicht bekannt sein dürften.

Mit dieser sehr reichhaltigen Auswahl ist die CD-ROM „TEX Live“ zwar für alle Eventualitäten gerüstet, andererseits wirkt das Angebot schon recht unübersichtlich und für den Neuling vielleicht sogar abschreckend. Es wurde deshalb versucht, das Angebot in sogenannte „Collections“ zu etwa einem Dutzend Themenkreisen (beispielsweise AMS, Plain TEX, LATEX, Grafik, Fonts, Dokumentation, usw.) zu gliedern und die darin enthaltenen Pakete jeweils in drei Gruppen („Basic“, „Recommended“, „Other“) zu klassifizieren. Das Installationsskript ermöglicht es nun, entweder für alle oder für jeden Themenkreis einzeln den gewünschten Umfang der Installation aus diesen drei Gruppen auszuwählen, und somit seinen jeweiligen Bedürfnissen anzupassen. Ferner besteht

also beispielsweise auch `sparc-sunos4.1.3`, `sparc-solaris2.4` und `sparc-solaris2.5` oder auch `i486-linux` und `i486-linuxaout`.

auch die Möglichkeit, mit einem anderen Skript gezielt einzelne Pakete zu installieren, wenn man genau weiß, was man möchte. Eine Liste aller Pakete ist dazu praktischerweise in der Dokumentation zur CD-ROM „TEX Live“ [3] enthalten.

Eine „Basic“-Installation aus allen Themenkreisen benötigt beispielsweise nur knapp 10 MB und enthält nur die allerwichtigsten Pakete, was es auch einem Anfänger ermöglichen sollte, problemlos zu einem relativ überschaubaren, aber dennoch sinnvoll zusammengestellten TEX-System zu gelangen. Eine „Recommended“-Installation belegt dagegen schon etwas über 90 MB und enthält damit wohl schon einiges mehr, als die meisten TEX-Anwender überhaupt haben möchten, so daß sich hierbei schon eine sorgfältigere Auswahl empfiehlt. Eine komplette „Other“-Installation dürfte letztlich wohl auf eine fast vollständige Kopie des CD-Inhalts hinauslaufen und nur in den seltensten Fällen sinnvoll sein.

Unabhängig von der Auswahl der zu installierenden Makro- und Font-Pakete besorgt das Installationsskript natürlich auch die Installation der Unix-Binaries. Die jeweils verwendete Unix-Plattform wird dabei automatisch erkannt und dem Anwender vorgeschlagen. Es ist ebenfalls möglich, Binaries auch gleich für mehrere Plattformen zu installieren, was sich wohl insbesondere für zentrale Server-Installationen anbietet. Letzteres dürfte die CD-ROM „TEX Live“ vielleicht auch speziell für die mitunter gestreßten Verwalter heterogener Institutsnetze an Universitäten interessant machen, die nicht immer die nötige Zeit haben, sich selbst um die Pflege ihrer TEX-Installation zu kümmern, selbst wenn sie über eine direkte Internet-Anbindung und damit über Zugang zu den CTAN-Servern verfügen. Ebensogut ist die CD-ROM „TEX Live“ natürlich auch für private Anwender geeignet, die eine TDS-konforme TEX-Verteilung suchen oder ihre bisherige TEX-Installation auf den neuesten Stand bringen möchten, wenn auch in diesem Fall wohl kaum Bedarf für die Unterstützung von 23 verschiedenen Unix-Plattformen bestehen dürfte.

## Fazit und Ausblick

Zusammenfassend kann man sagen, daß es sich bei der CD-ROM „TEX Live“ um ein durchaus gelungenes Erstlingswerk handelt, das man guten Gewissens weiter empfehlen kann. Sicherlich gibt es das eine oder andere, was noch verbessert werden könnte. Beispielsweise könnte man sich noch flexiblere Auswahlmöglichkeiten der zu installierenden Pakete oder eine bessere Erschließung der auf der CD-ROM angebotenen Dokumentation wünschen, jedoch müssen hierzu erst einmal Erfahrungen gesammelt werden. Abgesehen davon mangelt es

derzeit sicherlich noch an der Unterstützung von DOS-, Windows-, Macintosh- und anderen Nicht-Unix-Plattformen, was für zukünftige Neuauflagen unbedingt anzustreben wäre, damit die CD-ROM „TEX Live“ wirklich zu einer für alle Systeme gleichermaßen geeigneten TEX-Verteilung werden kann. Allerdings wird bei Einbeziehung zusätzlicher Plattformen wohl an anderer Stelle gespart werden müssen, da auch die Kapazität einer CD-ROM leider begrenzt ist.

Nicht verschwiegen werden soll hier auch, daß sich leider auch einige Fehler eingeschlichen haben, angefangen von einem fehlerhaften Installationsskript und anderen kleinen Pannen bis hin zu der einen oder anderen falsch eingeordneten Datei im `texmf`-Baum, was die Funktion allerdings nicht beeinträchtigt und lediglich einige TDS-Puristen stören dürfte.

Leider unterliegt die CD-ROM „TEX Live“ wie auch alle anderen TEX-Verteilungen auf CD-ROM dem grundsätzlichen Problem, daß sie schon sehr bald nach dem Erscheinen etliches von ihrer Aktualität verloren hat. Erwähnt seien hier nur das LATEX $2_\varepsilon$-Release und die Version 1.3 der DC-Fonts vom Juni 1996 oder die inzwischen erschienene neue Version der `teTeX`-Verteilung (`teTeX-0.4`), in der nun auch METAPOST enthalten ist, was bisher noch fehlte. Letztendlich spricht dies aber nicht gegen die CD-ROM „TEX Live“, ist sie doch immer noch die aktuellste aller derzeit angebotenen TEX-CDs. Vielmehr verdeutlicht dies nur den Bedarf an regelmäßig erscheinenden, aktualisierten TEX-CDs, die vielleicht in Zukunft besser nicht an den Terminen von TEX-Tagungen sondern an den Erscheinungsdaten der wichtigsten Makro- oder Font-Pakete wie beispielsweise LATEX $2_\varepsilon$ oder den DC/EC-Fonts orientiert werden sollten.

Zum Abschluß bleibt eigentlich nur noch die Frage, wo man die CD-ROM „TEX Live“ bekommen kann. Entstanden ist sie in einem Gemeinschaftsprojekt der TEX Users Group (TUG) mit den britischen und französischen TEX-Benutzergruppen UK TUG und GUTenberg und wurde demzufolge zunächst auch nur von diesen Gruppen sowie der niederländischen Benutzergruppe NTG vertrieben, die jedoch Mitgliedern von DANTE e.V. dieselben Ermäßigungen wie den eigenen Mitgliedern einräumen. Mit etwas Verspätung ist inzwischen nun auch DANTE e.V. in den Kreis der Anbieter eingetreten und bietet die CD-ROM „TEX Live“ ab sofort für Mitglieder zum Preis von 25,– DM an.

Zum Lieferumfang gehört neben der CD-ROM eine 36-seitige Dokumentation [3] mit Installationsanleitung, Hinweisen und Tips zur `teTeX`-Verteilung, einer vollständigen Liste aller Pakete, sowie Erläuterungen zur TDS-Struktur. Weitere Informationen, insbesondere zu Bug-Fixes und Updates gibt es ferner

auf dem WWW-Server der TUG unter `http://www.tug.org/tex-live.html` sowie auf den CTAN-Servern (`ftp.dante.de`) im Verzeichnis `info/texlive/`.

Abschließend bleibt eigentlich nur noch folgende Empfehlung zu zitieren:

> *If you are on any flavour of Unix, BUY this CD!*
> *There is no complicated compilation or moving*
> *of installed files around, it will just WORK!*
>
> — SEBASTIAN RAHTZ, *TWG-TDS* (1996/05/30)

## Literatur


[1] TUG Working Group on a TEX Directory Structure: *A Directory Structure for TEX Files (Version 0.999)*; erhältlich via FTP von allen CTAN-Servern, z. B. von `ftp://ftp.dante.de/tex-archive/tds/draft-standard/`.

[2] TUG Working Group on a TEX Directory Structure: *A Directory Structure for TEX Files (Version 0.999)*; *TUGboat*; 16#4, S. 401–413; 1995.

[3] Sebastian Rahtz (Editor): *The TEX Live Guide*; Mai 1996.

[4] Thomas Esser: *The teTeX distribution and the TEX Live CD*; Keynote address, Journée GUTenberg; Mai 1996; Folien erhältlich via WWW von `http://www.informatik.uni-hannover.de/~te/talks/`.

[5] Joachim Lammarsch: *Beschreibung der CD-ROM von DANTE e.V.*; *Die TEXnische Komödie*; 3/95, S. 48–50; 1995.

[6] Joachim Schrod: *TDS – Die vorgeschlagene TEX Directory Structure*; *Die TEXnische Komödie*; 3/95, S. 44–47; 1995.

[7] *Archive der Mailing-Liste* `TEX-CD@SHSU.EDU`; abrufbar via E-Mail von `FILESERV@SHSU.EDU` mit `"SENDME TEX-CD.yyyy-mm"`; ab `1993-12`.

[8] *Archive der Mailing-Liste* `TWG-TDS@SHSU.EDU`; abrufbar via E-Mail von `FILESERV@SHSU.EDU` mit `"SENDME TWG-TDS.yyyy-mm"`; ab `1994-05`.

# Beschleunigung eines LaTeX-Durchlaufs

Peter Nüchter

Wohl jeder LaTeX-Benutzer hat sich schon über die langwierigen Meldungen beim Laden irgendwelcher *Packages* geärgert, die bei jedem TeX-Durchlauf erscheinen. Natürlich sind es nicht die Meldungen selbst, die stören, aber gerade beim Bearbeiten kurzer Teildokumente kommt der *Overhead* des Ladens zeitlich doch deutlich zum Tragen.

Wenn man sich als LaTeX-Benutzer langsam vom „Kopka“ [2] zum „Begleiter“ [1] hocharbeitet, so verliert man doch den Bezug zum eigentlichen TeX. Denn *natürlich* bietet TeX schon von Hause aus genau das, was man braucht: IniTeX!

Die Idee schien so einfach, daß ich überrascht war, als es auf Anhieb klappte: Man nehme die komplette Präambel seines LaTeX-Dokuments, also alles, was vor `\begin{document}` steht und nicht mehr geändert werden soll, und schreibt diese Präambel in eine separate Datei, der ich im folgenden den Namen `footex.tex` gebe. Dinge, wie beispielsweise Definitionen für eigene Makros, die man noch nachträglich ändern möchte, sollten in dieser Datei nicht enthalten sein oder per `\input` eingebunden werden. Mit Hilfe von IniTeX erzeugt man dann aus `footex.tex` eine Format-Datei namens `footex.fmt`. Nun muß man nur noch sein eigenes Shell-Kommando, beispielsweise unter dem Namen `footex`, bereitstellen und schon kann man dieses sehr spezielle „LaTeX“ verwenden!

## Kochrezept

1. Bereitstellen des Kommandos `footex`:
   - Unix-Benutzer haben es ganz einfach:

     Stellen Sie fest, wo das Programm `virtex` liegt und machen Sie einen sogenannten Soft-Link in Ihr `~/bin`-Verzeichnis unter dem Namen `footex`:

     ```
     $ which virtex
     virtex is /usr/local/bin/virtex
     $ ln -s /usr/local/bin/virtex ~/bin/footex
     ```

- TeX-Benutzer unter anderen Betriebssystemen müssen das Kommando entsprechend definieren. Beispielsweise erstellen MS-DOS-Benutzer eine Batchdatei `latex.bat` bzw. `latex.cmd`, die TeX mit dem Format `footex.fmt` explizit aufruft.

2. Erstellen des Formats `footex.fmt`:

   Als Unix-Benutzer rufen Sie `initex '&latex footex \dump'` auf. Damit wird im Arbeitsverzeichnis die Format-Datei `footex.fmt` erzeugt, die dann von unserem im Schritt zuvor erzeugten Kommando `footex` verwendet wird. Der Suchpfad für Format-Dateien, den man z. B. über die Umgebungsvariable `TEXFORMATS` angeben kann, muß dazu das aktuelle Verzeichnis enthalten.

3. Bearbeiten des eigentlichen LaTeX-Dokuments, wobei dieses Dokument keine Präambel besitzt, mit dem Kommando `footex`.

Vorteil des Verfahrens: Man kann beliebig viele verschiedene `footex.tex`-Dateien in beliebigen Arbeitsverzeichnissen verwenden. Das Kommando `footex` muß nur einmal bereitgestellt werden.

Nachteil dieses Verfahrens: Nach jeder Änderung der Datei `footex.tex` ist ein IniTeX-Durchlauf erforderlich. Das könnte durch ein *Makefile* automatisch erledigt werden.

## Beispiel

Der Text eines Artikels steht in einer Datei namens `texkom.tex`. Die erste Zeile dieser Datei lautet `\begin{document}`. Die zugehörige LaTeX-Präambel steht in der Datei `footex.tex`:

```
\documentclass[a4paper]{article}
\usepackage{german,xspace}
\newcommand{\initex}{Ini\TeX\xspace}
\setcounter{secnumdepth}{0}
\endinput
```

Nach dem IniTeX-Durchlauf, mit dem die Format-Datei `footex.fmt` erstellt wird, muß dann jeweils nur noch `footex texkom` aufgerufen werden, um das Dokument `texkom.tex` zu übersetzen. Der Zeitgewinn ist beträchtlich!

## Literatur

[1] Michel Goossens, Frank Mittelbach und Alexander Samarin: *Der LaTeX-Begleiter*; Addison-Wesley, Bonn; 1994.

[2] Helmut Kopka: *LaTeX: Einführung*; Addison-Wesley, Bonn; 2., überarbeitete Aufl., 1996.

*Anmerkung der Redaktion:* David Carlisle, ein Mitglied des LaTeX 2ε-Teams, hat für das von Peter Nüchter beschriebene Verfahren ein Makropaket geschrieben, mit dem zusätzliche Informationen über die so vorgeladenen LaTeX 2ε-Klassen und -Pakete in der Format-Datei abgelegt werden. Das Makropaket `mylatex.ltx` kann man auf dem CTAN-Server von DANTE e.V. (`ftp.dante.de`) im Verzeichnis

```
macros/latex/contrib/supported/carlisle/
```

finden. Im Kommentar dieses Makropakets findet man weitere Informationen.

Man sollte bei dieser Vorgehensweise bedenken, daß es in einigen Fällen zu unerwarteten Problemen bei der Erzeugung und der anschließenden Verwendung einer speziellen Format-Datei kommen kann. Deshalb enthält `mylatex.ltx` zusätzlichen Code, damit beispielsweise die Deklaration `\makeindex` korrekt bearbeitet wird. Ebenso gibt es noch weitere Deklarationen, die normalerweise ohne Probleme in der Präambel eines Dokuments verwendet werden können, die jedoch in einer so erzeugten Format-Datei nicht mehr korrekt funktionieren.

Bernd Raichle

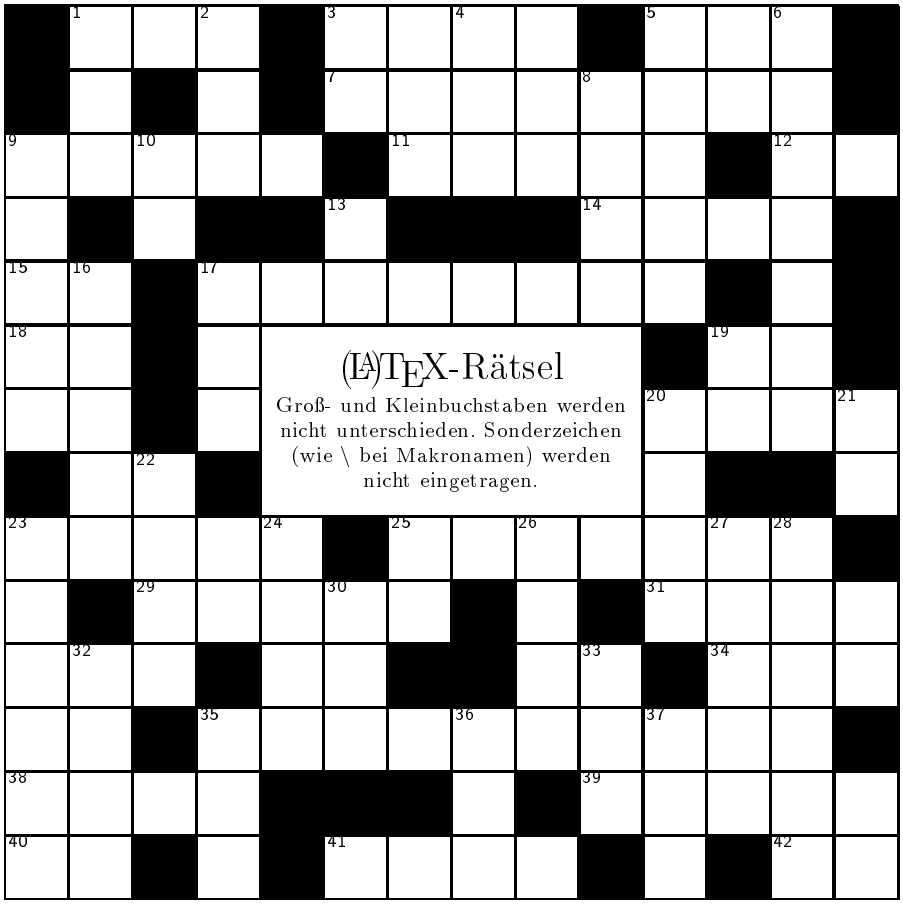
# Das alternative (LA)TEX-Glossar – Addendum

Prof. Dr. Klaus Lagally

Im folgenden sind einige Begriffe zu finden, die das Glossar aus Heft 4/95 ergänzen.

| | |
|---|---|
| `\accent` | MlTEX |
| `\afterassignment` | völlige Erschöpfung |
| `\allowbreak` | Mensaessen |
| `\arg` | schlimm |
| `\ast` | großer Zweig |
| `\badness` | *siehe auch* `\arg`, *siehe auch* `\break` |
| `\body` | Reizwäsche; |
| | Leiche |
| `\break` | *siehe auch* `\allowbreak` |
| `\brokenpenalty` | Strafe fürs Kaputtmachen |
| `\bull` | BSE-Träger |
| `\closein` | Knast |
| `\closeout` | Aussperrung |
| `\colon` | Querdarm (med.) |
| `\cong` | *siehe auch* `\hong` |
| `\count` | Graf |
| `\dag` | `\day`, *siehe auch* Babel (dänisch) |
| `\ddots` | Punkte, gestottert |
| `\dim` | düster |
| `\dosupereject` | Umgang mit Kritikern |
| `\downbracefill` | BH Größe 115F |
| `\dummy` | Blödmann |
| `\eightpoint` | Abart des Marienkäfers |

| | |
|---|---|
| `\empty` | |
| `\endgraf` | Aberkennung des Adelstitels |
| `\errmessage` | falsche Nachricht, Ente |
| `\escapechar` | entlaufener Buchstabe |
| `\expandafter` | (zensiert, zu vulgär) |
| `\fi` | *siehe auch* `\phi` (nach Rechtschreibreform) |
| `\firstmark` | erste Punkte in Flensburg; erster Eintrag ins Klassenbuch |
| `\floatingpenalty` | Baden verboten! |
| `\futurelet` | Kredit in Aussicht |
| `glue set` | Flickzeug |
| `\grave` | Grab |
| `\hoffset` | Reizwäsche, *siehe auch* `\body` |
| `\ifcat` | haben Sie eine Katze? |
| `\ifdim` | im Dämmerlicht |
| `\ifinner` | laßt mich raus hier! |
| `\iftrue` | ich glaub es nicht |
| `\ifvoid` | noch ein Bier bitte |
| `\immediate` | *siehe auch* `\nu` |
| `\lastmark` | Exmatrikulation |
| `\topmark` | Klassenprimus |
| `undefined reference` | *siehe auch* `\hong` |
| `\span` | SpannerIn |
| `\xleaders` | Politiker im Ruhestand |




**Waagrecht:** 1 Font mit geschwungenen Großbuchstaben 3 Zeichen ə (wasysym) 5 intern verwendetes LaTeX 2ε-Paket aus dem Tools-Bundle 7 senkrechtes Gespenst 9 $\delta$ 11 War das alles? 12 å 14 Tabulatorpositionen (plain-TeX) 15 angelsächsisches Längenmaß 17 Unteraufzählungspunkt (plain-TeX) 18 PostScript 20 Divisionsrest 23 $\supsetneqq$ ($\mathcal{AMS}$) 25 LaTeX-Schleife 29 $\oplus$ 31 Literaturreferenz 34 | 35 Noch 'n mathematischer Satz 38 einzelne Zeile über volle Textbreite (plain-TeX) 39 Math-Extension-Font (plain-TeX) 40 $\gg$ 41 Ansonsten 42 Tiefstellen (Alternative zu _)

**Senkrecht:** **1** BibTeX-Stil für das Format des „Council of Biology Editors“ **2** Weise aktuelle Bedeutung zu! (TeX-Primitiv) **4** $\approx$ (wasysym) **5** Familie des Nichtproportionalfonts **6** Font mit Horoskopzeichen **8** BibTeX-Notizfeld **9** DVI-Gerätetreiber **10** Logarithmus **13** $\neq$ **16** $\not\succ$ ($\mathcal{AMS}$) **17** $\int$ **19** $\pm$ **20** $\prec$ **21** Maßeinheit nach franz. Buchdrucker **22** BibTeX-Typ **23** LaTeX-Paket für Algorithmen **24** Kleister **26** Zeichen ə (wasysym) **27** TeX-Längenregister **28** $\otimes$ **30** Schriftherstellen **32** Zeichenprogramm **33** Winkelfunktion **35** $\neq$ **36** weiter, schmaler oder weniger als das (TeX-Primitiv) **37** Querverweis

# Spielplan

## Termine

**24.9.–26.9.1996** EP96 – International Conference on Electronic Documents, Document Manipulation and Document Dissemination
Xerox Research Center, Palo Alto, Kalifornien, USA
Kontakt: Xerox Corporation

**10.10.–11.10.1996** 15. Mitgliederversammlung von DANTE e.V.
Universität Hamburg
Kontakt: Reinhard Zierke

**24.10.1996** 18. NTG Meeting
University of Utrecht, Niederlande
Am Tag vor oder nach dem Treffen findet ein Ein-Tages-Kurs „(La)TEX and Graphics“ statt, der auf METAPOST, PostScript, mftoeps, etc. eingehen wird. Rechtzeitige Anmeldung ist erforderlich!
Kontakt: Kees van der Laan

**8.12.–12.12.1996** SGML '96
Toronto, Kanada
Kontakt: Marion Elledge

**26.2.–28.2.1997** DANTE '97 und
16. Mitgliederversammlung von DANTE e.V.
Fachhochschule München
Kontakt: Thomas Hafner

**6.4.–11.4.1997** Hypertext '97 – 8th ACM Conference on Hypertext
Southampton, UK
Kontakt: Wendy Hall

# Stammtische

*In verschiedenen Städten im Einzugsbereich von DANTE e.V. finden regelmäßig Treffen von TEX-Anwendern statt, die für Jeden offen sind. Wer gerne auch einen solchen Termin anbieten möchte, um sich mit anderen TEXies auszutauschen, schickt einfach die Adresse der Ansprechperson, die Adresse des Treffpunktes und den Zeitpunkt des Treffens zur Veröffentlichung an die Redaktion.*

**10587** Berlin
Rolf Niepraschk
Physikalisch-Technische Bundesanstalt
Abbestr. 2–12
Tel.: 030/34 81 316
`niepraschk@ptb.de`
*Gaststätte „Bärenschenke“*
*Friedrichstr. 124*
*Letzter Donnerstag im Monat, 19.00 Uhr*

**22527** Hamburg
Volker Huettenrauch
`volker_huettenrauch@hh.maus.de`
*Themi/Xenios, Osterstr. 46*
*Letzter Mittwoch im Monat, 18.00 Uhr*

**28359** Bremen
Martin Schröder
Tel.: 0421/22 39 425
`ms@dream.hb.north.de`
*Universität Bremen, MZH 4. Stock*
*gegenüber den Fahrstühlen*
*Erster Donnerstag im Monat, 18.30 Uhr*

**35392** Gießen
Günter Partosch
HRZ der Justus-Liebig-Universität
Heinrich-Buff-Ring 44
`guenter.partosch@`
`hrz.uni-giessen.de`
*„Licher Bierstuben“, Licher Straße*
*Letzter Montag im Monat, 19.30 Uhr*

**42283** Wuppertal
Andreas Schrell
Erlenstr. 1
Tel.: 0202/50 63 81
`Andreas_Schrell@w2.maus.de`
*Gasthaus „Yol“, Ernststr. 45*
*Zweiter Donnerstag im Monat, 19.30 Uhr*

**47226** Duisburg
Friedhelm Sowa
Rheinstr. 14
*„Gatz an der Kö“, Königstraße 67*
*Dritter Dienstag im Monat, 19.30 Uhr*

**50931** Köln
Uwe Münch
Schmittgasse 92
51143 Köln
Tel.: 02203/8 71 11
`muench@ph-cip.uni-koeln.de`
*Zentrum für Paralleles Rechnen,*
*Weyertal 80*
*Vierter Dienstag im Monat, 20.00 Uhr*

**65195** Wiesbaden
Christian Kayssner
Elsässer Platz 9
Tel.: 0611/48 11 7
*Andreas Klause, Elsässer Platz 3*
*Erster Montag im Monat, 20.00 Uhr*

**69008** Heidelberg
Luzia Dietsche
Tel.: 06221/2 97 66
`dante@dante.de`
*China-Restaurant Palast*
*Lessingstr. 36*
*Letzter Mittwoch im Monat, 20.00 Uhr*

**76128** Karlsruhe
Klaus Braune
Tel.: 0721/6 08 40 31
`braune@rz.uni-karlsruhe.de`
*Universität Karlsruhe, Rechenzentrum*
*3. OG Raum 316*
*Zirkel 2*
*Erster Donnerstag im Monat, 19.30 Uhr*

# Adressen

DANTE, Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 10 18 40
69008 Heidelberg

Tel.: 0 62 21 / 2 97 66
Fax: 0 62 21 / 16 79 06
e-mail: dante@dante.de

Konten: Postgiroamt Karlsruhe
BLZ 660 100 75

| | |
|---|---|
| 2134 00-757 | für Beiträge |
| 2946 01-750 | für Bücher und Disketten |
| 1990 66-752 | für Tagungen |

## Präsidium

| | | |
|---|---|---|
| Präsident: | Joachim Lammarsch | (president@dante.de) |
| Vizepräsident: | Uwe Untermarzoner | (vice-president@dante.de) |
| Schatzmeister: | Friedhelm Sowa | (treasurer@dante.de) |
| Schriftführerin: | Luzia Dietsche | (secretary@dante.de) |

## Server

| | |
|---|---|
| ftp: | ftp.dante.de [129.206.100.192] |
| e-mail: | ftpmail@dante.de |
| gopher: | gopher.dante.de |
| WWW: | http://www.dante.de/ |
| Mailbox: | 0 62 21/16 84 26 (nur für Mitglieder) |

# Autoren/Organisatoren

**Ulrich Breymann** [37]
Hochschule Bremen
Neustadtswall 30
28199 Bremen
breymann@alf.zfn.uni-bremen.de

**Luzia Dietsche** [3]
siehe Seite 72

**Marion Elledge** [69]
Information Technologies
100 Daingerfield Road, 4th Floor
Alexandria, VA 22314-2888, USA
mern@well.sf.ca.us

**Thomas Hafner** [69]
FH München, FB Elektrotechnik
Dachauerstr. 98 b
80335 München
dante97@fitug.de

**Wendy Hall** [69]
Dept. of Electr. and Comp. Science
University of Southampton
Southampton SO17 1BJ, UK
ht97-info@ecs.soton.ac.uk

**Donald E. Knuth** [10]
Computer Science Dept.
Stanford University
Stanford, CA 94305, USA

**Markus Kohm** [14]
Augartenstraße 27
76137 Karlsruhe

**Prof. Dr. Klaus Lagally** [66]
Institut für Informatik
Breitwiesenstrasse 20-22
70565 Stuttgart
lagally@informatik.uni-stuttgart.de

**Joachim Lammarsch** [4]
siehe Seite 72

**Gerd Neugebauer** [43]
Mainzer Str. 8
56321 Rhens
gerd@informatik.uni-koblenz.de

**Peter Nüchter** [62]
Univ. Ulm, Abt. Mikrowellentechnik
Albert-Einstein-Allee 41
89069 Ulm
peter@mwt.e-technik.uni-ulm.de

**Bernd Raichle** [33, 53]
siehe Seite 74

**Friedhelm Sowa** [6]
siehe Seite 72

**Kees van der Laan** [69]
Hunzeweg 57
9893 PB Garnwerd, NL
Tel.: +31/594/621525
cgl@rc.service.rug.nl

**Ulrik Vieth** [55]
Heinrich-Heine-Universität Düsseldorf
Institut für Theoretische Physik II
Universitätsstraße 1
40225 Düsseldorf

**Xerox Corporation** [69]
3400 Hillview Avenue
Palo Alto, California 94304, USA
ep96@xsoft.xerox.com

**Reinhard Zierke** [69]
Univ. Hamburg, FB Informatik
Vogt-Kölln-Straße 30
22527 Hamburg
Tel.: 040/5494-2295
dante-mv15@informatik.uni-hamburg.de

# Technischer Beirat

*Zuschriften an die Koordinatoren werden in der Regel nur beantwortet, wenn ein ausreichend frankierter und adressierter Rückumschlag mitgeschickt wird. Die Koordinatoren sind nicht verpflichtet, auf jede Frage einzugehen.*

*Amiga*
Markus Erlmeier
Postfach 415
84001 Landshut
Tel.: 0871/77939
Fax: 0871/75381
Btx: 087177939-0001
MAUS: `Markus Erlmeier@LA`
FIDO: `2:2494/106.21`
`amiga@dante.de`

*Atari*
Stefan Lindner
Karolinenstr. 52b
90763 Fürth
`atari@dante.de`
*oder*
Lutz Birkhahn
Darfelder Str. 38
48727 Billerbeck
Tel.: 02543/4666
`atari@dante.de`

*German-Style*
Bernd Raichle
Stettener Str. 73
73732 Esslingen
`german@dante.de`

*Graphik*
Friedhelm Sowa
Heinr.-Heine Universität
Rechenzentrum
Universitätsstr. 1
40225 Düsseldorf
Tel.: 0211/3113913
`graphik@dante.de`

*Macintosh*
Lothar Meyer-Lerbs
Am Rüten 100
28357 Bremen
Tel.: 0421/252624
`macintosh@dante.de`

*Mailbox von DANTE e.V.*
Jürgen Unger
Ringstr. 24
64668 Rimbach
`mailbox@dante.de`

*METAFONT*
Jörg Knappen
Barbarossaring 43
55118 Mainz
`metafont@dante.de`

*OS/2*
Thomas Koch
Hauptstr. 367
53639 Königswinter
`os2@dante.de`

*PostScript*
Jürgen Glöckner
Ph.-Schmitt-Str. 8b
69207 Sandhausen
Tel.: 06224/3750
`postscript@dante.de`

*PubliCTEX*
Dr. Peter Breitenlohner
Max-Planck-Institut für Physik
Postfach 401212
80805 München
`pc@dante.de`

*Server-Koordination*
Dr. Rainer Schöpf
Zentrum für Datenverarbeitung
der Universität Mainz
Anselm-Franz-von-Bentzel-
Weg 12
55099 Mainz
`server@dante.de`

*Treiberentwicklung und SGML*
Joachim Schrod
Kranichweg 1
63322 Rödermark-Urberach
`treiber@dante.de`

*UNIX*
Dr. Klaus Braune
Universität Karlsruhe
Rechenzentrum
Zirkel 2
76128 Karlsruhe
Tel.: 0721/608-4031
`unix@dante.de`

*Verlag und Buchhandel*
Christa Loeser
Trübnerstr. 38
69121 Heidelberg
Tel.: 06221/400177
Fax: 06221/472909
`verlage@dante.de`

*VMS*
Ralf Gärtner
Rüschhausweg 14
48161 Münster
`vms@dante.de`

# Die TEXnische Komödie

8. Jahrgang Heft 2/1996 August 1996