MAPS

NUMMER 30 • VOORJAAR 2004

NEDERLANDSTALIGE TEX GEBRUIKERSGROEP

Voorzitter
Hans Hagen
pragma@wxs.nl

Secretaris
Willi Egger
w.egger@boede.nl

Penningmeester
Wybo Dekker
wybo@servalys.nl

Bestuursleden
Maarten Wisse
Maarten.Wisse@urz.uni-hd.de

Frans Goddijn
frans@goddijn.com

Karel Wesseling
k.h.wesseling@planet.nl

Postadres
Nederlandstalige TEX
Gebruikersgroep
Maasstraat 2
5836 BB Sambeek

Postgiro
1306238
t.n.v. NTG, Deil
BIC-code: PSTBNL21
IBAN-code: NL05PSTB0001306238

E-mail bestuur
ntg@ntg.nl

E-mail MAPS redactie
maps@ntg.nl

WWW
www.ntg.nl

De **Nederlandstalige TEX Gebruikersgroep (NTG)** is een vereniging die tot doel heeft de kennis en het gebruik van TEX te bevorderen. De NTG fungeert als een forum voor nieuwe ontwikkelingen met betrekking tot computergebaseerde documentopmaak in het algemeen en de ontwikkeling van 'TEX and friends' in het bijzonder. De doelstellingen probeert de NTG te realiseren door onder meer het uitwisselen van informatie, het organiseren van conferenties en symposia met betrekking tot TEX en daarmee verwante programmatuur.
De NTG biedt haar leden ondermeer:

- Tweemaal per jaar een NTG-bijeenkomst.
- Het NTG-tijdschrift MAPS.
- De 'TEX Live'-distributie op DVD/CDROM inclusief de complete CTAN softwarearchieven.
- Verschillende discussielijsten (mailing lists) over TEX-gerelateerde onderwerpen, zowel voor beginners als gevorderden, algemeen en specialistisch.
- De FTP server `ftp.ntg.nl` waarop vele honderden megabytes aan algemeen te gebruiken 'TEX-producten' staan.
- De WWW server `www.ntg.nl` waarop algemene informatie staat over de NTG, bijeenkomsten, publicaties en links naar andere TEX sites.
- Korting op (buitenlandse) TEX-conferenties en -cursussen en op het lidmaatschap van andere TEX-gebruikersgroepen.

Lid worden kan door overmaking van de verschuldigde contributie naar de NTG-giro (zie links); vermeld IBAN- zowel als SWIFT/BIC-code en selecteer shared cost. Daarnaast dient via `www.ntg.nl` een informatieformulier te worden ingevuld. Zonodig kan ook een papieren formulier bij het secretariaat worden opgevraagd.
De contributie bedraagt € 40; voor studenten geldt een tarief van € 20. Dit geeft alle lidmaatschapsvoordelen maar *geen stemrecht*. Een bewijs van inschrijving is vereist. Een gecombineerd NTG/TUG-lidmaatschap levert een korting van 10% op beide contributies op. De prijs in euro's wordt bepaald door de dollarkoers aan het begin van het jaar. De ongekorte TUG-contributie is momenteel $65.

MAPS bijdragen kunt u opsturen naar `maps@ntg.nl`, bij voorkeur in LaTEX- of ConTEXt formaat. Bijdragen op alle niveaus van expertise zijn welkom.

Productie. De Maps wordt gezet met behulp van een LaTEX class file en een ConTEXt module. Het pdf bestand voor de drukker wordt aangemaakt met behulp van pdfetex 1.11b Web2C 7.5.2 draaiend onder Linux 2.4 en Mac OS X Panther. De gebruikte fonts zijn Bitstream Charter, schreefloze en niet-proportionele fonts uit de Latin Modern collectie, en de Euler wiskunde fonts, alle vrij beschikbaar.

TEX is een door professor Donald E. Knuth ontwikkelde 'opmaaktaal' voor het letterzetten van documenten, een documentopmaaksysteem. Met TEX is het mogelijk om kwalitatief hoogstaand drukwerk te vervaardigen. Het is eveneens zeer geschikt voor formules in mathematische teksten.
Er is een aantal op TEX gebaseerde producten, waarmee ook de logische structuur van een document beschreven kan worden, met behoud van de letterzetmogelijkheden van TEX. Voorbeelden zijn LaTEX van Leslie Lamport, $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX van Michael Spivak, en ConTEXt van Hans Hagen.

Inhoudsopgave

Redactioneel

Voor u ligt het dertigste nummer van de MAPS. Hoewel we als redactie zoals gewoonlijk veel moeite hebben moeten doen om u tot schrijven te bewegen, zijn we van mening dat we toch nog een zeer interessante MAPS hebben weten te produceren.

Dat neemt niet weg, dat we ons over de volgende MAPS wel zorgen maken. Daarom is op de laatste ledenvergadering besloten om u een extra stimulans te geven door de volgende MAPS in kleur uit te voeren. Want natuurlijk voert u al uw TEX-werkstukken in kleur uit en hebt u steeds gedacht: dat is niets voor de MAPS, daarin komen mijn verhalen niet tot hun recht.

Dat excuus gaat voor de volgende MAPS dus niet meer op! Eindelijk kunt u die kleurige menukaart die u voor uw dochters' trouwdag maakte in bredere kring verspreiden, net als het programma voor het liefdadigheidsconcert waar men u op de Rotary Club zo mee complimenteerde! Aan uw collega-leraren kunt u laten zien hoe men vragen-en-oplossingen met kleurige kopjes en plaatjes in elkaar zet en aan uw collega-astronomen hoeveel mooier TEX kan zijn dan Power Point.

Maar laat u dat niet weerhouden ook uw zwart-wit verhalen in te sturen — de volgende MAPS zal niet alléén maar kleur bevatten.

Dan wil ik u nu graag meenemen langs de bonte verzameling van artikelen in deze MAPS:

□ Siep Kroonenberg (*The Maps style*) heeft de Maps class file drastisch aangepast — vandaar dat u nu een MAPS voor u ziet die met vrije fonts is gezet. Siep legt uit waarom en hoe, zowel voor LaTEX als voor ConTEXt.

□ Piet van Oostrum (*Een uittreksel uit de recente bijdragen in het* CTAN*-archief*) heeft voor ons bijgehouden wat er de laatste tijd zoal aan interessants in het CTAN-archief is opgenomen.

□ Siep Kroonenberg (*Schatgraven op TEXlive*) inspireert u door u langs minder bekende kleinoden op de TEXLive te leiden.

□ Hans Hagen(*TEXLive collection — past and future*) vertelt over de aanzienlijke ontwikkelingen die plaats vinden in de TEXLive distributie.

□ Taco Hoekwater (*De CXTEX distributie*) is druk bezig een minimalistische ConTEXt-distributie samen te stellen, bestemd voor gebruik door leken. Hij geeft een overzicht van wat daar zoal bij komt kijken en wat ermee te winnen valt.

□ Hans Hagen (*The Scite–TEX integration*) heeft de flexible SciTE-editor voor ConTEXt ingericht en vertelt over de mogelijkheden van die combinatie.

□ Wybo Dekker (*Introducing oldstyle figures in existing virtual fonts*) wist tevoren helemaal niets van dit onderwerp, vroeg hulp aan de goeroes, en werd tenslotte in meer details gedirigeerd dan hij ooit gedacht had te kunnen bevatten. Dit is een verslag van zijn queeste.

□ Adam Lindsay (*Apple symbols* en *Unicode Symbols*) gaat in op MacOSX-specifieke fonts en hun installatie. Deze twee artikelen zijn licht bewerkte versies van Adam's *My Way* publicaties.

□ Wybo Dekker (*Woordafbreking op ë en ï*) liep aan tegen vreemd afbreekgedrag en zocht eens uit hoe het nu precies zit.

□ Roland Smith (*LaTEX genereren vanuit C-programma's*) laat zien hoe men LaTEX-uitvoer kan genereren vanuit C-programma's. Zijn C-programma betreft de eigenschappen van vezelversterkte kunststoflaminaten — vandaar dan ook dat hij C met Perl lamineert om zijn doel te bereiken.

□ Willi Egger (*Help! The typesetting area*) neemt ons mee door de historie van de boekdrukkunst, met nadruk op de paginaindeling. Een aantal van ons komt het bekend voor, omdat Willi hierover op de voorlaatste NTG-dag een boeiend verhaal heeft gehouden.

□ Siep Kroonenberg (*TEX and prepress*) verhaalt van de valkuilen en -strikken waarin u kunt raken wanneer u na het schrijven van uw boek opgelucht met uw PDF naar de drukker tijgt. Zorgvuldige lezing van dit verhaal kan u voor veel teleurstellingen behoeden.

□ Piet van Oostrum (*Een tutorial over het gebruik van BibTEX*) vertelde op de voorlaatste NTG-dag een boeiend verhaal over BibTEX. Dit artikel legt dat verhaal niet alleen vast, maar omvat nog een aanzienlijke hoeveelheid extra informatie.

De redactie verwacht dat hier veel inspiratie voor u klaar ligt!

Wybo Dekker
wybo@servalys.nl

The Maps style*

Abstract
This paper introduces the renewed Maps classfile and includes some usage notes.

Keywords
Maps Latex classfile Context module fonts

You may have noticed some changes to the Maps style: a simpler article title, different headers, no footers, a different choice of fonts.

The main motivation, from my part, was simply a desire for something new.

Fonts

The revised Maps makes use of free fonts exclusively: Bitstream Charter as main text font, Euler[1] for math and Latin Modern[2] for everything else.

There are some nice correspondences in letterforms between Charter and Latin Modern Sans. However, the large x-height of Charter and the small x-height of Latin Modern Sans makes them tricky to combine. Charter is scaled to 95% to compensate for this.

Unfortunately, smallcaps for Charter don't belong to the free set. Therefore, the Maps style does not use smallcaps.

Getting the Maps style

If you are writing for the Maps, you may want to format your paper yourself in the Maps style. There is no real need to do this; the standard Latex article style or simple Context markup serve just as well.

There should be a download link on `http://www.ntg.nl/maps.html`. The package includes Latex- and Context sample files. The Maps font setup has not been activated, in order to avoid unnecessary complexities and incompatibilities for authors.

Using the Latex version

Document class options

Layout options. The default Latex Maps layout is two-column, with the title and the optional subtitle over both columns. There are two single-column layouts, a symmetric one (document option `onecolumn`) and an asymmetric one (document option `asym`).

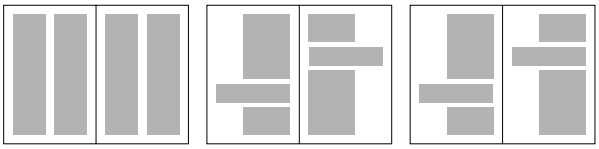


Figure 1. The three Maps layouts: two-column, one-column and asymmetric. The latter two layouts each have 'wide' sections, sticking out into the wide margins.

The default two-column layout option is implemented with the built-in Latex two-column option rather than with the `multicol` package.

Other document options. There are two additional options to tune the look of your paper:

□ `nosubsub` defines only two different levels of sectioning. If you use the two-column layout and don't need more, then this option will probably improve the look of your paper.

□ `deftables` suppresses extra vertical space around vertical rules in tabulars; see the section on tabulars.

Article start

An article may start like this:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\documentclass {maps}
\usepackage{graphicx}
\begin{document}
\author[Anton Ulrich Thor]{%
  Anton Ulrich Thor\\
  Institute of Indefinite Studies\\
  Unseen University\\
  Ankh Morpork\\
    \texttt{a.u.thor@uu.am.dw}}
\title[Short title]{%
  Long title,\\
  which may not fit one one line%
    \thanks{Whoever...}}
%\subtitle{An optional subtitle}

\maketitle

\begin{abstract}
This is an abstract.
```

*We thank the TEX Live people for providing us with a rich ready-to-run TEX environment.

```
\end{abstract}

\begin{keywords}
Example, Maps, Latex
\end{keywords}

\section{A section}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The optional short author parameter will be used for the first and all odd page headers. The long author parameter, which has no particular structure, will be typeset at the end. You should include the optional parameter if the `author` command is long or complex or includes newlines.

Because the full author parameter is typeset at the very end, a `thanks` command is only supported for the title.

It is strongly recommended to include abstract- and keywords environments.

Section numbering

Use section numbers only if you actually refer to them. To encourage this, the classfile turns off section numbering even for sections. You can turn it back on with

```
\setcounter{secnumdepth}{..}
```

Lists

If you have lists with long entries and use the default two-column layout, give the `itemouter`, `enumouter` and `descript` environments a try. These avoid protracted indentation:

```
\begin{itemouter}
\item This is the first item of a
  non-indented itemized list,
  produced with the \texttt{itemouter}
  environment.
\item This is the second item.
\end{itemouter}
```

produces

□ This is the first item of a non-indented itemized list, produced with the `itemouter` environment.

□ This is the second item.

Tabulars

The Maps classfile adds some vertical space around horizontal rules in tables. This makes vertical rules look funny, but most of the time you are better off without vertical rules anyway; see table 1. If you really insist on vertical rules, use the `deftables` document option.

var	value
$Q_{s,max}$	0.18
K_s	1.0
$Y_{x/s}$	0.5
$Y_{p/s}$	0.854
$Q_{p,max}$	0.0045
μ_{crit}	0.01
k_h	0.002
m_s	0.025

var	value
$Q_{s,max}$	0.18
K_s	1.0
$Y_{x/s}$	0.5
$Y_{p/s}$	0.854
$Q_{p,max}$	0.0045
μ_{crit}	0.01
k_h	0.002
m_s	0.025

Table 1. Tabulars with and without vertical rules

Footnotes

Footnotes have been turned into endnotes. You need to manually add a command `\theendnotes` to get them to print. The title of the notes section is defined by the command `\notesname`.

Wide typesetting in single-column layout

For both single-column layouts, there are environments `fullwidth` and `verboutdent` which typeset their content across the full page, including most of the wide margin; see figure 1.

```
\begin{fullwidth}
x x x x x x x x x x x x x x x x x x x x
x x x x x x x x x x x x x x x x x x x x
x x x x x x x x x x x x x x x x x x x x
x x x x x x x x x x x x x x x x x x x x
\end{fullwidth}

\begin{verboutdent}
{}\/$xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
\end{verboutdent}
```

The implementation of `fullwidth` is rather simplistic and may easily break, in which case more sophisticated hackery will be needed.

The Context version

The Context version supports the same general layouts but not all the features of the Latex version. Consult the sample file in the Maps package.

Article start

A Context Maps article might start as follows:

```
% non-default layouts:
%\enablemode[onecolumn]
%\enablemode[asym]
% subsection lowest level of sectioning
%\enablemode[nosubsub]
\usemodule[map-se]
```

```
\starttext
\startArticle [%
   Page=13,
   Title={An example document},
   SubTitle={with an optional subtitle},
   Author={%
     Anton Ulrich Thor\\
     Institute of Indefinite Studies\\
     Unseen University\\
     Ankh Morpork\\
     \type{a.u.thor@uu.am.dw}},
   RunningAuthor={Anton Ulrich Thor}%
]

\startAbstract
...
\stopAbstract

\startKeywords
...
\stopKeywords

\section{...}
```

Section numbering

Use section numbers only if you actually refer to them. To encourage this, the Maps module turns off section numbering even for sections. You can turn them back on with

```
\setupheads [sectionnumber=yes]
```

Lists

A Context counterpart of the `itemouter` Latex environment is `outeritemize`:

```
\startouteritemize
\item ...
\item ...
\stopouteritemize
```

Full-width verbatims are also available:

```
\startwidetyping
...
\stopwidetyping
```

The Context version, like the Latex version, turns footnotes into endnotes. You may want to put a command such as

```
\def\notesname{Footnotes}
```

in your preamble. You also need to explicitly place the footnotes:

```
\placefootnotes
```

More advice

As mentioned above, we certainly don't require authors to use the Maps style. We basically publish the Maps style because people ask for it. The Maps style is a work in progress anyway.

If you want to facilitate production, the following will make more of a difference to us:

□ Use clean, minimalist markup.

□ Don't try to fix typographic or layout problems yourself; your hacks are likely to get in our way.

□ Minimize your use of packages. Don't use packages that merely serve to improve the appearance of your paper.

□ Make sure you send a complete submission, including all the less-standard packages that you use.

□ Use eps, pdf, png or jpg formats for graphics; jpg only for photographs.

□ Don't convert screenshots to jpg; both compression rate and quality will be disappointing.

□ Include a pdf of your document, as a check for us.

Notes

1. The eulervm package by Walter Schmidt makes the Euler math font family a suitable drop-in replacement for Computer Modern, and combines well with many commercial fonts.
2. Latin Modern is a Type 1 reimplementation of Computer Modern with a large character set, which supports the T1 encoding.

Siep Kroonenberg
siepo@cybercomm.nl

Een uittreksel uit de recente bijdragen in het CTAN archief

Abstract
Dit artikel beschrijft een aantal recente bijdragen uit het CTAN archief. De selectie is gebaseerd op wat ik zelf interessant vind en wat ik denk dat voor veel anderen interessant is. Het is dus wel een persoonlijke keuze. Het heeft niet de bedoeling om een volledig overzicht te geven. De uitgebreidere bijdragen zijn ook geen handleidingen. Beschouw het maar als een soort menukaart die de bedoeling heeft om de lezer nieuwsgierig te laten worden.

Keywords
TEX, LaTEX, packages, CTAN, classes, beamer, slides, fonts,

Presentaties met LaTEX

N.B. Het onderstaande is niet een vergelijkend warenonderzoek. Daarvoor zou meer onderzoek nodig zijn. Misschien schrijft iemand nog eens een artikel dat dit doet.

Al vanaf het begin van LaTEX had Leslie Lamport voorzien in de mogelijkheid om met behulp van LaTEX overhead presentaties te maken. Hiervoor was de klasse (toen nog *documentstyle* genoemd) slides. Deze klasse had als bijzondere kenmerken:

□ Er werden speciale fonts gebruikt. Deze fonts waren schreefloos omdat algemeen aangenomen wordt dat dit voor projectie beter leest. Hij gebruikte echter niet het normale schreefloze font omdat hierbij het onderscheid tussen bijvoorbeeld hoofdletter 'I' en kleine letter 'l' te klein is. De 'I' heeft in dat font daarom wel schreven. Verder werden er natuurlijk grotere afmetingen van fonts gebruikt dan voor gedrukte tekst. En tenslotte was er voor elk font ook een onzichtbare tegenhanger, waarbij de tekens dezelfde afmetingen hadden als de gewone fonts maar geen pixels. Deze werden gebruikt om 'overlays' te maken, waarbij je een stukje tekst weglaat en dat in een volgende slide toevoegt.

□ Zoals hierboven al genoemd de mogelijkheid om 'overlays' te maken.

□ Een aantal opties uit de standaard klassen, zoals het gebruik van zwevende figuren en tabellen, was weggelaten.

Ik heb dit in de verleden tijd geschreven, hoewel de klasse slides ook in moderne LaTEX distributies aanwezig is en zelfs met LaTEX 2_ε gemoderniseerd. Echter deze klasse biedt voor de moderne presentator slechts minimale faciliteiten. In die tijd werd presenteren voornamelijk met een overhead projector gedaan en de klasse is daar eigenlijk ook voor bedoeld[1]. Voor gebruik met een beamer is het te primitief.

Het gebruik van de speciale fonts maakte het moeilijk om je eigen fonts met slides te gebruiken. Je moest dan zorgen dat je de fontgroottes aanpaste, en als je overlays wilde gebruiken moest je ook nog de onzichtbare tegenhangers hebben. Je kon dus niet zomaar `\usepackage{times}` of zo gebruiken zoals bij andere klasses.

Latere ontwikkelingen. Voor veel mensen was de klasse slides toch niet goed genoeg. Vandaar dat er vele pogingen zijn ondernomen om klasses te maken die meer faciliteiten boden. Een bekend pakket was seminar van Timothy van Zandt. Ook dit was bedoeld voor overhead transparanten, maar het had meer mogelijkheden, zoals het aanbrengen van frames om een slide. Seminar was vooral bedoeld om samen te werken met pstricks van dezelfde auteur waardoor prachtige slides gemaakt konden worden.

Het fontprobleem loste seminar op door in de LaTEX source gewone afmetingen te gebruiken, maar een TEX magnification te gebruiken waardoor in de DVI driver alles met een factor opgeblazen werd. Dit heeft echter tot gevolg dat alle afmetingen (o.a. figuren) ook opgeblazen worden, dus hier moet je bij het specificeren van afmetingen hiermee rekening houden. Mijn persoonlijke ervaring is dat ik dit erg hinderlijk vond; ik heb seminar wel gebruikt maar toen er zich betere alternatieven aandienden ben ik vooral om deze reden er snel weer van afgestapt.

Ook seminar stamt uit de tijd voor de beamer en is dus gericht op het produceren van overhead transparanten.

Beamer presentaties. Toen beamers populair werden kwamen er speciale klassen voor presentaties op beamers. Vooral het gebruik van PDFTEX (en dus PDFLaTEX) heeft deze ontwikkeling gestimuleerd. Ik denk dat conTEXt hierbij voorliep op LaTEX. Er is mo-

menteel echter een breed scala aan klasses en stijlen voor het produceren van beamerpresentaties met LaTeX. Enkele zal ik hierbij noemen. Ze zijn ofwel al aanwezig in moderne TeX-distributies of ze zijn te downloaden van CTAN.

Prosper Prosper is een uitbreiding van seminar geschreven door Frédéric Goulard. Het voorziet in gekleurde achtergronden, effecten bij de overgang van een slide naar de volgende (met Acrobat Reader), overlays (waarbij dus delen van de slide na elkaar verschijnen) en verschillende stijlen.
Enkele uitbreidingen van prosper zijn HA-prosper van Hendri Adriaens en ppr-prv (Prosper Preview), een klasse om afdrukken van prosper slides te maken (2 per pagina).
Hoewel prosper van CTAN te downloaden is, is de originele site ervoor op Sourceforge: `http://prosper.sourceforge.net/`. Er is ook een Wiki site: `http://wikiprosper.bbclone.de/`

Pdfslide Pdfslide is een slides pakket van C. V. Radhakrishnan, die ook al het iets simpeler pdfscreen had geschreven. Pdfslide werkt alleen met pdflatex, en kan dus geen DVI produceren. Het kenmerkende van pdfslide is de navigatiebalk die normaliter gebruikt wordt. Het komt ook met verschillende stijlen en ondersteunt de pagina-overgangen van Acrobat Reader.
Pdfslide is geen klasse maar een package (`pdfslide.sty`) en wordt bijvoorbeeld in samenhang met de article class gebruikt.

TeXpower `texpower` is een package van Stephan Lehmke dat ook gebruikt kan worden om slides te maken maar vaak ook gebruikt wordt in samenhang met de andere pakketten (prosper, pdfslide, beamer) om dynamiek in de slides te brengen: delen die later verschijnen of weer verdwijnen. Hiervoor heeft texpower het commando `\pause` waarmee een pauze in de slide ingelast wordt. Bij het voortgaan (bijvoorbeeld met een muisklik) wordt dan het volgende deel van de slide vertoont. Dit commando kan alleen in verticale mode (zeg maar tussen twee alinea's) gebruikt worden.
Er is ook een krachtiger mechanisme waarmee delen van de slide in willekeurige volgorde vertoond kunnen worden, bijvoorbeeld om een deel van een formule of zin in te vullen of te vervangen.
Ook texpower is op Sourceforge te vinden (`http://texpower.sourceforge.net/`). Hier zijn ook verschillende voorbeelden te vinden.

Beamer Beamer is de meest recente (voor zover ik weet) loot aan de stam van de slides pakketten in LaTeX. Beamer heeft weer alle genoemde karakteristieken zoals animaties, verschillende stijlen e.d. Wat mij verder opviel was de mogelijkheid om zowel een artikel als een presentatie op eenvoudige wijze uit dezelfde LaTeX file te genereren.
Ook beamer is op Sourceforge te vinden met diverse voorbeelden (`http://latex-beamer.sourceforge.net/`).

Tenslotte is een overzicht van verschillende presentatiepakketten (niet alleen TeX) te vinden op `http://www.miwie.org/presentations/presentations.html`.

Framed

Een pakket wat ik graag wil aankondigen is framed van Donald Arseneau. Een bekende TeXer die er niet voor terugdeinst om moeilijke TeX problemen aan te pakken. Het probleem dat framed aanpakt is het voorzien van een achtergrondkleur of een kader van een tekst die toch over meerdere pagina's afgebroken moet worden. Als ik het goed heb is deze faciliteit standaard in conTeXt aanwezig. In LaTeX niet. In de moderne vormgeving van bijvoorbeeld leerboeken wordt vooral de faciliteit van gekleurde achtergronden veel gebruikt. De standaard manier van LaTeX is de `\fbox` of iets met minipages maar deze breken niet af bij een paginaovergang.

Framed is een klein package (256 regels inclusief commentaar), en zit vol met TeX trucks. Ik weet niet of het in alle situaties goed werkt (Donald noemt het een 'pre-production' versie), want TeX is hier eigenlijk niet goed voor ontworpen. Het package heeft omgevingen framed, shaded en leftbar voor resp. kaders, achtergrondkleuren en een streep links van de tekst. Het is ook mogelijk zelf soortgelijke omgevingen te maken. De tekst in de omgeving wordt bij een paginaovergang gesplitst en de delen worden dan aan een macro aangeboden die de afwerking (bijv. het kader) doet.

Fonts

Latin Modern is een font familie in Type 1 formaat die afgeleid is van de Computer Modern serie. Het is niet zomaar een vervanger van Computer Modern en aanverwante fonts, in de zin dat je TeX document hetzelfde blijft en alleen de DVI processor of pdftex de andere fonts gebruikt zoals dit bij CM-Super gebeurt. Het gebruik van Latin Modern moet specifiek aangegeven worden bijvoorbeeld in LaTeX met `\usepackage{lmodern}` en in conTeXt met `\usetypescript`. De serie bevat behalve Computer Modern ook tekensets in de Cork encoding (dus vergelijkbaar met de EC fonts) en andere tekens. De reacties zijn in het algemeen enthousiast, vanwege de goede kwaliteit ervan. Deze is beter dan die van CM-Super, terwijl de fonts veel minder schijfruimte innemen.

Het interessante van de Latin Modern fonts is dat ze

ook gebruikt kunnen worden in andere programma's zoals tekstverwerkers. Met de originele TEX fonts kan dit niet, voornamelijk omdat darin de spatie niet als teken aanwezig is.

Latin Modern fonts worden in de nieuwe MAPS stijl gebruikt voor sommige elementen, bijvoorbeeld de sectiekopjes. Zie ook het artikel van Siep Kroonenberg 'The Maps style' in dit nummer.

Fourier-GUTenberg is een collectie fonts van de Franse TEX gebruikersgroep GUTenberg, die juist niet op Computer Modern gebaseerd is. Het gebruikt ook niet de OT1 encoding zoals Computer Modern maar de T1 encoding.

Een probleem wanneer je eens iets anders wilt is altijd om een font te vinden dat de wiskundige tekens van TEX ondersteunt en goed bij het broodfont past. Deze combinaties zijn nauwelijks aanwezig of ze zijn duur. Fourier-GUTenberg gaat uit van de Adobe Utopia fonts en voegt daar wiskundige en diverse andere symbolen aan toe. Zie figuur 1 voor een voorbeeld. In het artikel van Siep Kroonenberg 'Schatgraven op TEX Live' in dit nummer staat nog een voorbeeld.

The usual notation for binomials is similar to the fraction concept, so it has a similar command `\binom` with two arguments. Example:

$$\sum_{\gamma\in\Gamma_C} I_\gamma = 2^k - \binom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2} + \cdots + (-1)^l\binom{k}{l}2^{k-l} + \cdots + (-1)^k = (2-1)^k = 1$$

Figuur 1. Voorbeeld Fourier-GUTenberg

Pict2e

Het lang verwachte package pict2e is een vervanger van de standaard LaTEX picture omgeving die gebruik maakt van de mogelijkheden die Postscript en pdflatex geven. De standaard picture omgeving is nogal beperkt: zo kunnen lijnen alleen richtingen hebben waarvan de richtingscoeficiënt uit te drukken is als een breuk van gehele getallen tussen -6 en $+6$, cirkels zijn alleen in beperkte groottes te verkrijgen enz. Dit alles omdat LaTEX zelf geen grafische primitieven heeft maar de pictures op moet bouwen uit tekens uit een speciaal font.

Pict2e daarentegen maakt wel gebruik van de postprocessor. Het gebruikt hierbij hetzelfde systeem als het graphics package, nl aparte drivers voor de verschillende uitvoermogelijkheden (bijvoorbeeld dvips en pdftex).

Andere bijdragen

Onderstaande bijdragen zijn LaTEX packages of classes, tenzij anders vermeld. De locaties zijn op CTAN. Meestal betreft het updates van bestaande pakketten.

ledmac een pakket voor het zetten van critische edities met LaTEX (meestal zijn dit antieke documenten waaraan commentaar van onderzoekers is toegevoegd). In `macros/latex/contrib/ledmac`

fancyhdr van ondergetekende. De nieuwe editie heeft de mogelijkheid om de plaats van de headers en footers in horizontale richting te specificeren, en is beschreven in de tweede editie van de LaTEX Companion. In `macros/latex/contrib/fancyhdr`.

ctable Typesetten van gecentreerde tabellen, en (multiple-)figure floats met footnotes. Van ons aller Wybo Dekker. In `macros/latex/contrib/ctable`.

vpp (View and Print Postscript/PDF); een script om postscript of PDF te previewen of te printen o.a. n-up en als A5 boekjes. Ook van Wybo. In `support/view_print_ps_pdf`.

WinShell een grafische shell voor TEX onder MS Windows. In `systems/win32/winshell`.

tpic2pdftex.awk een awk script om figuren in de pic taal om te zetten in PDF. In `graphics/tpic2pdftex`.

empheq een uitbreiding van amsmath voor formulers, o.a. over- en onderaccolades, verschillende pijlsoorten. In `macros/latex/contrib/empheq`.

aurical (AuriocusKallgraphicus) een calligrafisch font. In `fonts/aurical`.

wasy fonts, een font met allerlei symbolen. In `fonts/wasy2`.

footmisc een package om allerlei lastige dingen met voetnoten te doen, bijvoorbeeld per pagina nummeren. In `macros/latex/contrib/footmisc`.

Noten

1. Een aantal jaren geleden kreeg ons instituut een kleurenprinter van het inktjet type. Toen gingen we natuurlijk kleurentransparanten maken. Echter het afdrukken van de transparanten kostte mij evenveel tijd als het lesgeven ermee.

Piet van Oostrum

Schatgraven op TEX Live

Abstract
Dit stuk brengt de rijke inhoud van de TEX Live cd onder de aandacht van de lezer.

Keywords
TEX Live, fonts, documentatie, TEX distributie

TEX installeren is sinds enkele jaren een fluitje van een cent. Leden van TEX gebruikersgroepen krijgen regelmatig TEX Live toegestuurd, een zeer complete vrije TEX distributie op cd. Deze kun je naar keus installeren of rechtstreeks zonder installatie vanaf cd draaien, althans onder Windows en de belangrijkste Unix varianten, inclusief Mac OS X. Of, als je al langs andere weg een aktuele TEX hebt, dan is TEX Live mooi om achter de hand te hebben voor extra spullen die je onverwacht toch nodig blijkt te hebben.

Maar als je je op een zondagmiddag verveelt dan kun je ook eens rondkijken wat de TEX Live cd allemaal te bieden heeft. Bekijk, bij wijze van wegwijzer, eerst de documentatie van TEX Live zelf (`texmf/doc/tldoc/`[1]).

Extra hulpprogramma's voor Windows

Linux- en Unix gebruikers beschikken over een scala aan hulp-programma's die meestal ontbreken op een Windows systeem maar vrijwel onmisbaar zijn voor wie met TEX wil werken. De TEX Live cd vult veel van deze tekorten aan: er zijn bijvoorbeeld Windows versies van Perl, Ghostscript, ImageMagick met convert (voor conversie tussen bitmapped grafische formaten), bzip2 (compressie), g[un]zip (compressie) en xemacs.

Documentatie

De 'The UK TEX FAQ' (`FAQ/english/newfaq`) is een document van bijna honderd pagina's met 286 vragen over vooral LaTEX maar ook over TEX in het algemeen.

Wie geen LaTEX boek heeft vindt een redelijke inleiding in 'De niet zo korte inleiding tot LaTEX 2ε' (`texmf/doc/guides/lshort-dutch/`). Dit boek is in verschillende talen beschikbaar.

Meer diepgaande informatie over LaTEX is beschikbaar in een aantal gespecialiseerde handleidingen van de LaTEX mensen: 'Packages in the 'graphics' bundle' (`texmf/doc/latex/graphics/grfguide.*`) en diverse handleidingen in `texmf/doc/latex/base/`, met name 'LaTEX 2ε font selection' (`fntguide`).

'The Font Installation Guide' (`texmf/doc/guides/Type1fonts/fontinstallationguide.pdf`) is een prachtig vormgegeven uitgebreide behandeling van Fontinst.

'The Comprehensive LaTEX Symbol List' (`texmf/doc/guides/symbols/`) kan ook handig zijn, evenals een tweetal LaTEX reference manuals in html formaat: `texmf/doc/html/latex2e-html/index.html` en `texmf/doc/html/latex/latex_toc.html`.

Voor veel LaTEX packages is documentatie te vinden onder `texmf/doc/latex/`.

De ConTEXt manuals zijn te vinden onder `texmf/doc/context/manual/`.

Kortom: als je vragen hebt over TEX dan zijn je eigen TEX installatie en de TEX Live cd de eerst aangewezen plaatsen om te zoeken naar een antwoord.

Fonts

Er is onder `texmf/fonts/` een schier onafzienbare collectie van vrije fonts.

Sommige zijn alleen beschikbaar in MetaFont formaat. Voor het genereren van mooie pdf-bestanden willen we die vermijden. Ze zijn wel geschikt voor gedrukte uitvoer, en knutselaars onder ons zijn wellicht bereid en in staat ze te converteren naar Type 1 formaat met b.v. textrace (niet op TEX Live).

Latin Modern: Computer Modern in Type1 formaat. pdf[e]tex is onder TEX Live en teTeX al geconfigureerd om zoveel mogelijk gebruik te maken van Type1 fonts; daar hoef je in veel gevallen geen moeite voor te doen.

Speciale aandacht verdient Latin Modern, dat geen conversie is maar een op MetaPost gebaseerde herimplementatie van de Computer Modern tekstfonts. De Latin Modern fonts hebben een grote characterset en kunnen daarom ook met T1 encoding worden gebruikt:

```
\usepackage[T1]{fontenc}
\usepackage{textcomp,lmodern}
```

De Maps gebruikt schreefloze varianten van Latin Modern voor kopjes en dergelijke.

De 'PostScript 35' en bijpassende math fonts. De meesten van ons zijn bekend met de vrije URW klonen van de standaard 35 fonts die in Adobe PostScript printers zijn ingebouwd. Wie bijvoorbeeld de Bookman familie uit deze set wil aktiveren kan volstaan met:

```
\usepackage{bookman}
```

in de preamble van zijn dokument. Formules worden dan nog steeds uit Computer Modern gezet, wat weinig fraai oogt.

Er is maar een beperkte keus in lettertypen voor wiskundige symbolen.

$$\sum_{i=1}^{qT-1} u_i^2 + \sum_{i=0}^{T-1} \mu_i \left\{ \left[\sum_{j=0}^{q-1} x_j u_{iq+j} \right] - v_i \right\}$$

Dit is een voorbeeld van Bookman met Computer Modern voor wiskunde.

Txfonts en pxfonts. Dit zijn vrije fontsets/packages die bijpassende fonts voor formules toevoegen aan respektievelijk Times en Palatino. Het kan niet eenvoudiger:

```
\usepackage{pxfonts}
```

Er is maar een beperkte keus in lettertypen voor wiskundige symbolen.

$$\sum_{i=1}^{qT-1} u_i^2 + \sum_{i=0}^{T-1} \mu_i \left\{ \left[\sum_{j=0}^{q-1} x_j u_{iq+j} \right] - v_i \right\}$$

Dit is een voorbeeld van pxfonts: Palatino met bijbehorende symbolen en SmallCaps en Old-Style figures 013.

Charter, Utopia en Fourier. Sinds jaar en dag zijn Bitstream Charter en Adobe Utopia vrij beschikbaar. De Maps gebruikt Charter voor broodtekst. Fourier, of voluit Fourier-Gutenberg, is, net als txfonts en pxfonts, een complete package dat Utopia gebruikt voor tekst en daar bijpassende wiskunde-fonts aan toevoegt. De texmf-extra directory op de TEX Live dvd bevat alle bestanden die hiervoor nodig zijn. Om licentie-redenen is Fourier niet in texmf zelf opgenomen.

Er is maar een beperkte keus in lettertypen voor wiskundige symbolen.

$$\sum_{i=1}^{qT-1} u_i^2 + \sum_{i=0}^{T-1} \mu_i \left\{ \left[\sum_{j=0}^{q-1} x_j u_{iq+j} \right] - v_i \right\}$$

Dit is een voorbeeld van Fourier: Adobe Utopia met bijbehorende symbolen.

Het eulervm alternatief. Walter Schmidt's Eulervm package maakt het eenvoudig om de calligrafisch getinte wiskundige Euler fontset te combineren met allerlei gangbare PostScript fonts.

Er is maar een beperkte keus in lettertypen voor wiskundige symbolen.

$$\sum_{i=1}^{qT-1} u_i^2 + \sum_{i=0}^{T-1} \mu_i \left\{ \left[\sum_{j=0}^{q-1} x_j u_{iq+j} \right] - v_i \right\}$$

Dit is een voorbeeld van Bookman met Eulervm.

De Poolse Antykwa's. Meer dekoratief zijn Antykwa Poltawskiego (LaTEX: package `antpolt`) en Antykwa Torunska (LaTEX: package `antyktor`). Deze laatste is gebruikt voor het nieuwe Maps logo.

Antykwa Torunska

Macro-pakketten

Gezien de hoeveelheid en verscheidenheid van het materiaal kan ik hier nauwelijks iets zinnigs over zeggen; kijk zelf maar onder `texmf` in het algemeen en `texmf/tex` in het bijzonder.

Downloaden

Mocht je de TEX Live cd niet in de bus gekregen hebben, dan kun je de TEX Live cd downloaden van CTAN sites, bijvoorbeeld:
`ftp://ftp.cs.uu.nl/mirror/tex-archive/` of
`ftp://ftp.dante.de/pub/tex/`,
beide onder `systems/texlive/Images/`, of
`http://tug.org/ftp/texlive/Images/`.
Voor de versie van 2003 zijn er drie cd images: een wat minder uitgebreide 'demo' cd image die je rechtstreeks kunt draaien; een dvd image die je rechtstreeks kunt draaien en een cd image die je wel kunt installeren op je computer maar niet rechtstreeks draaien. De TUG site heeft ze alle drie; de eerste twee alleen de laatste.

CTAN

De dvd bevat ook een snapshot van het CTAN archief. Kijk vooral ook naar Piet van Oostrum's artikel over CTAN in dit nummer.

Noot

1. De bestanden en folders hebben betrekking op de TEX Live van 2003 maar zullen in de meeste gevallen ook van toepassing zijn op andere edities.

Siep Kroonenberg
`siepo@cybercomm.nl`

TeXLive Collection
past and future

Abstract
Past and future of the TeXLive Collection is described.

Keywords
TeXLive Collection

Introduction

It must have been in the second half of the eighties that I obtained a copy of the TEXbook. It contained what appeared to me as fascinating magic. Then our company purchased microTEX, the software program ready to run on a personal computer. It came with a DVI viewer and a printer driver for a matrix printer. From there we moved on to a big PCTEX, Y&Y's DVIPSONE, BLUESKY's outline fonts, now all history.

A few years later we learned of the Dutch speaking TEX User Group NTG and, because we had run into some limitations of TEX —too small a hash— we tried EMTEX, which later became part of 4TEX. 4TEX was one of the first TEX distributions on CDROM, an integrated set of the most popular programs available in the TEX world. We depended on the yearly updates of 4TEX and later TEXLIVE, of which version 8 was released in 2003, until today.

Beginning with version 8 TEXLIVE has become the TEX Collection. It combines an out–of–the–box TEX system and the complete CTAN repository (Comprehensive TEX Archive Network: a snapshot of almost all that is available for TEX users). TEX systems started on floppy disks but soon filled CDROM's and now DVD's. An archive of a couple of hundred files grew into tens of thousands.

tree	directories	files	bytes
texmf	3.750	45.000	626 M
texmf-extra	115	1.500	66 M
bin	16	2.500	250 M
source	380	6.900	104 M

If the CTAN archive is included we have a grand total of 138.000 (unzipped even 420.000) files, organized in 10.000 directories, totaling 5.906.870.829 byte, or 6 GB.

With version 8 the organizers realized that comprehensive began to become incomprehensible. Even though the TDS, the TEX Directory Structure, had brought some order in grouping files they were still faced with the fact that old TEX systems had been replaced with new systems in a continuous process to adapt to changing operating systems, improved text editors and more sophisticated and generally available viewers and printers. Fundamental changes appeared necessary and are implemented in the TEX Collection 2004. This paper will focus on some of the most important of these changes.

The engine

Donald Knuth's TEX was the ground breaking program that could typeset and be a programming language at the same time. TEX as a typesetting engine has been adapted to handle larger size memory, extended with features, translated into other programming languages, like C, and with the coming of PDF, the Portable Document Format, is now capable of producing PDF output directly with PDF-ε-TEX. The most important change in the 2004 release is that PDF-ε-TEX has become the main TEX engine. PDF-ε-TEX incorporates all 'accepted' extensions with proven reliability, produces DVI output by default, PDF when commanded, and ε-TEX is in there once explicitly enabled. To trigger PDF output Context users just add as the first line in their text files:

```
% output=pdftex
```

Context is a monolithic and coherent package of macro definitions that use the programming abilities of almost any TEX to accomplish a large variety of easy to use special typesetting functions.

Other macro packages have often been associated with a specific TEX binary. In practice this lead to several combinations of so called format files holding the macro definitions and binaries.

For plain TEX the system call (on the command line) and the engine are the same.

system call	format	engine
tex	plain.fmt	tex
etex	etex.efmt	etex
pdftex	pdftex.fmt	pdftex
pdfetex	pdfetex.efmt	pdfetex

For Latex the system call matches not the engine but

the format name. Here the command that starts TEX and loads a format is just a shortcut to calling the engine with a specific format.

system call	format	engine
latex	latex.fmt	tex
pdflatex	pdflatex.efmt	pdftex

For Context each format is named after the user interface language, the language of commands, messages, keywords, and so forth. This must not be confused with the language of the document text to be typeset. Each interface can handle all document languages.

system call	format	engine interface
cont-cz	cont-cz.efmt	pdfetexczech
cont-de	cont-de.efmt	pdfetexgerman
cont-en	cont-en.efmt	pdfetexenglish
cont-it	cont-it.efmt	pdfetexitalian
cont-nl	cont-nl.efmt	pdfetexdutch
cont-ro	cont-ro.efmt	pdfetexromanian

Normally, however, Context is launched by TEXEXEC, a PERL script that automates many annoying user tasks.

So, what is the importance of the change to PDF-ε-TEX in the 2004 Collection? Very little for the user, the system calls are unchanged! For TEXLIVE system maintenance, however, the change means that the various different TEX binaries can be removed and replaced by a single TEX engine that combines them all: PDF-ε-TEX. Extensions like ε-TEX, pdfTEX, MLTEX and ENCTEX are no longer needed as separate entities.

system call	format	engine
tex	plain.fmt	pdfetex
etex	etex.efmt	pdfetex
pdftex	pdftex.fmt	pdfetex
pdfetex	pdfetex.efmt	pdfetex
latex	latex.fmt	pdfetex
pdflatex	pdflatex.efmt	pdfetex

Because of the growing dependency on this engine PDF-ε-TEX has rigourous quality assurance and DANTE, NTG, and TUG have decided to financially support its primary author Hàn Thế Thành to extend and improve the program.

A change such as this is not trivial since it must be certain that existing documents can be processed without change, and macro packages must still believe that the correct binary is available. Macro packages may use undocumented features and nasty tricks to determine what engine is present. Currently PDFTEX is extended to take care of this problem. The configuration file has gone, more extensive map file handling has been implemented, and extensions are being separated to allow for experimental versions (XPDFETEX).

PDF-ε-TEX, although quite universally useful, still lacks some features such as Unicode awareness. TEX engine development, therefore, must continue. Those on the Context mailing list may know Giuseppe Bilotta as an enthousiastic user and advocate of TEX. In 2003 Giuseppe published EOMEGA, an extended version of TEX that uses Unicode natively. His initiative evolved into the ALEPH project which aims at merging ε-TEX with OMEGA. This is because some Context users wanted to use OMEGA features. Latex is also moving towards ε-TEX, enhancing the importance of the ALEPH initiative.

Those who have become dependent on OMEGA may get attracted by ALEPH's image: stable realware thus giving it a good chance to become the default engine under the OMEGA based formats on TEXLIVE. Producing PDF output directly is not a feature but the DVIPDFMX converter can produce the same rich PDF output as PDF-ε-TEX does for Context users.

Latin Modern

What more is new on the TEXLIVE 2004? First of all, the Latin Modern fonts. This project was funded by user groups. The fonts are extended versions of Computer Modern, with additional characters covering all western languages. Latin Modern will replace the textual part of Computer Modern Roman. The fonts are already on TEXLIVE 2003, so you can play with them. For instance, `cmr10`, `aer10`, `plr10`, `csr10` as well as in the near future `vnr10` will be replaced by `lmr10`. This change is downwards compatible. It removes a lot of nearly duplicate files from TEXLIVE. If all works out well, users will not notice the font change. Of course, the original `cmr10` will still be present.

Currently extra instances are made with a few more glyphs, more kerning pairs. Visual improvements are made based on suggestions by Donald Knuth in his errata documents.

Font files

A more drastic change is that some files change place in the TDS tree. Until now the encoding (`enc`) and the fontmap (`map`) files were located under the DVIPS and PDFTEX paths:

```
texmf/dvips
texmf/dvips/config
texmf/dvips/config/whatever
texmf/pdftex
texmf/pdfte
texmf/pdftex/config/whatever
```

The configuration file `texmf.cnf` informs applications about where to find these encoding and fontmap files. A changed `texmf.cnf` assures that most applications and users will not encounter problems. The new locations are:

```
texmf/fonts/enc/whatever
texmf/fonts/map/whatever
texmf/fonts/lig/whatever
```

Note the new ligature path. It is used by for instance AFMTOPL. Some changes are already reflected in the current TEXLIVE version but probably go unnoticed because both old and new locations are supported.

If you install your own fonts you need to relocate your map files. Font metrics remain in their usual place and encoding files are seldomly made by users. Instead of relocating another option is to adapt the `texmf.cnf` file, but this would complicate future updating. It is better to not touch this file.

Scripts

Context includes some PERL scripts taking care of sorting the index, managing multiple runs and other chores. Initially, the number of scripts was small and they ended up in a dedicated Context directory.

Since then other macro packages also come with PERL scripts and Context added RUBY scripts leading to these paths:

```
texmf/context/perltk
texmf/context/ruby
```

TEXLIVE uses stubs in the binary path to launch such scripts. The stubs use KPSEWHICH to locate the main script file. For reasons of consistency, maintainance and robust locating, scripts now have their own root path, for Context:

```
texmf/scripts/context/perl
texmf/scripts/context/ruby
```

Companion files that do not fit in this directory structure remain where they are located presently. In practice users will not notice the changes because the stubs take care of things. Future versions of KPSEWHICH will provide more robust and convenient ways to locate such script files.

Beware: if you write your own scripts you should realize that call to KPSEWHICH have to be adapted, for instance:

```
kpsewhich -progname=context
  -format="other text files" texexec.pl
```

is now:

```
kpsewhich -progname=context
  -format="scripts" texexec.pl
```

More

AFM files will no longer be distributed in their compressed form (gzip). Engine dependent TEX source files end up in specific paths. Most common users will not notice because users of engine dependent sources have their own way of structuring the directory tree.

The KPSE file searching library and tools get a few more features. Next year's TEXLIVE will have a completely rewritten version of this library, one that opens some windows to the future such as automatic updating, remote processing, and fetching resources from zip archives.

Production

Getting TEXLIVE ready requires an enormous effort. Only a few macro collections are submitted in the right structure. Consequently, much scripting takes place to get the files where they belong in the tree. Interdependencies are not always made clear and maintainers of packages come and go. When the structure changes files need to be relocated. Bugs in binaries need to be solved. New features have to be tested first. Documentation needs to be updated. Frequently new CDROM images are constructed and tested, on all platforms. Thus the TEXLIVE mailing list is a busy one. Last year we even had a show–stopper. At press time it was discovered that 8 bit file output no longer worked.

Finally, the Collection has to be produced. The 2003 Collection was the first to be distributed on DVD. Even after TEXLIVE and CTAN were put on the DVDplenty of space was available, so extra's were added (in the `texmf-extra` path) and the next release will provide even more. The DVD is one of the first dual layer data DVD's. This meant producing special split ISO–images and proofing of the first DVD: the presses were actually stopped after the first copy for testing!

In 2003 and 2004 DANTE invited those involved in this monster performance to their main anual meeting, altogether some 15 contributors from all over the world. They discussed the present and the future of such distributions. I leave the reporting of that discussion to the chairman. Happy users of TEXLIVE, however, should recognize with gratitude that getting this job done is far from trivial and effortless. We all should treasure those who are making TEXLIVE happen year after year. You can find their names on the cover of the DVD and in the documentation.

Summary

When the next TEXLIVE shows up in your postbox, update and things will work as usual. If you have your own fonts installed, however, you need to relocate your personal mapfiles to `.../fonts/map`, and run `mktexlsr` to update your files database. Also, if your scripts use KPSEWHICH, check them.

Hans Hagen
pragma@wxs.nl

De CXTEX distributie

Abstract
Het einddoel van het CXTEX project is om een hele texexec aanroep van begin tot eind te kunnen uitvoeren binnen één enkel, zo efficiënt mogelijk, systeemproces.
De eerste onderdelen van deze distributie worden in dit artikel gepresenteerd: nog op zichzelf staande, maar al wel naar C vertaalde versies van texexec, texutil en pdfetex.

Intro

In het dagelijks taalgebruik hebben we het over 'TEX draaien' of 'texexec runnen'. Dat wekt de indruk dat er sprake is van één enkele executable die wordt uitgevoerd om een PDF-bestand te genereren. De werkelijkheid is echter aanzienlijk complexer, omdat een bestand vaak herhaaldelijk gecompileerd moet worden. Dat kan zijn vanwege kruisverwijzingen, maar het kan ook voorkomen dat er externe programma's aangeroepen moeten worden voor bijvoorbeeld het aanmaken van indexen en de bibliografie.

Een concreet voorbeeld: ConTEXt maakt gebruik van twee verschillende hulpprogramma's:

Texutil

Tijdens het verwerken van een TEX-bestand bewaart ConTEXt allerlei informatie (onder andere voor de inhoudsopgave, lijsten van tabellen en figuren, registergegevens en kruisreferenties) in een bestand met als extensie `.tui`.

Dit bestand wordt na de TEX aanroep verwerkt door het programma texutil, om gebruikt te worden als invoer voor een eventuele volgende TEX aanroep.

Texexec

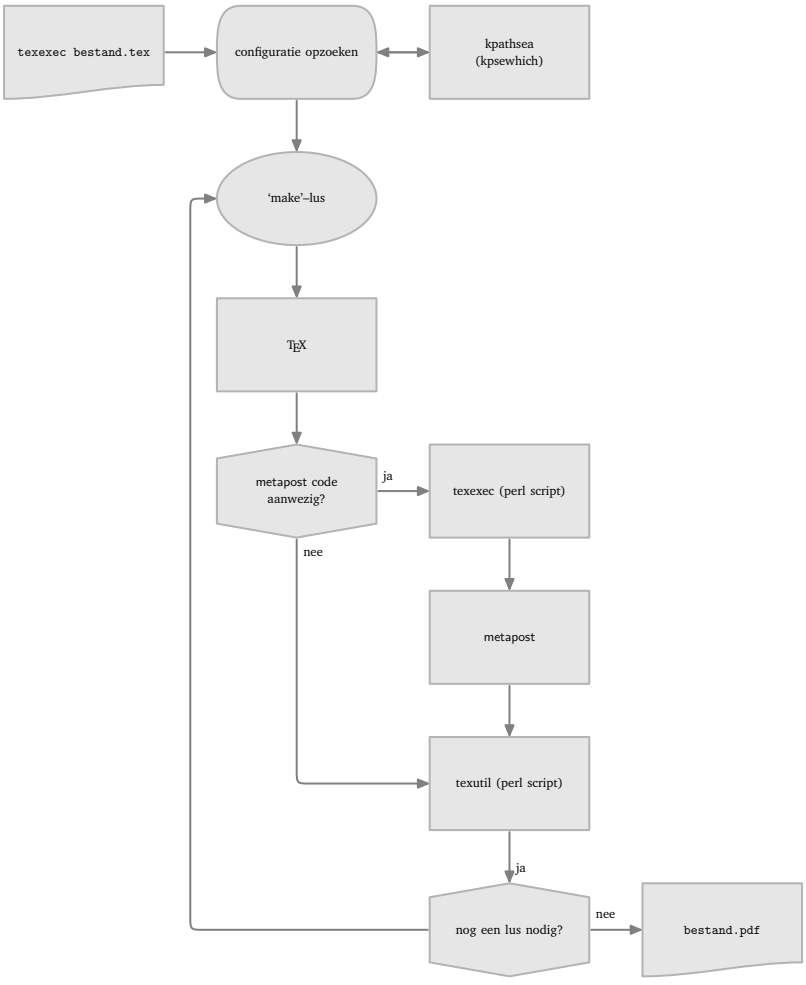
texutil en TEX zelf worden aangestuurd door het programma texexec. De belangrijkste taak van dit programma is om 'make' te emuleren. Dat wil zeggen: ervoor te zorgen dat TEX en texutil vaak genoeg worden aangeroepen om alle referenties op te lossen.

Bovendien verzorgt texexec het (eventueel) uitvoeren van metapost voor het aanmaken van gewenste plaatjes.

Het stroomdiagram van texexec is te zien boven aan de volgende bladzijde.

Omwille van het overzicht is ook nog het een en ander verwijderd uit het diagram:

- De twee stappen 'texexec (perl script)' en texutil (perl script)' voeren eenzelfde 'find configuration <=> kpathsea (kpsewhich)' actie uit als die je ziet weergegeven op de eerste regel van het diagram.
- En 'texutil' en 'texexec' bevatten zelfs nog een subprocedure om de nodige perl scripts *zelf* te vinden: de executables zijn namelijk kleine programma's die eerst `texutil.pl` dan wel `texexec.pl` opzoeken op de harde schijf, en daarna perl aanroepen.
- En dan zijn er nog diverse impliciete en expliciete programma-aanroepen in de scripts: commando's die uitgevoerd worden om bijvoorbeeld tijdelijke bestanden te maken, kopiëren of verwijderen.



Het gevolg van dit alles is dat tijdens het uitvoeren van het commando

```
$ texexec paper
```

er pakweg 60 verschillende processen worden opgestart voordat texexec klaar is met het aanmaken van het PDF-bestand voor het artikel dat je nu zit te lezen.

Het opstarten en afsluiten van al die processen (die grotendeels met elkaar communiceren via bestanden op de harde schijf) beslaat een flink gedeelte van de tijd die gemoeid is met het 'texexec runnen'.

Doelstelling

Ons bedrijf (Elvenkind) is een product aan het ontwikkelen (op basis van TEX uiteraard) dat XML Formatting Objects converteert naar PDF-bestanden. Omdat we dit programma willen en moeten leveren aan klanten die nog nooit van TEX, metapost, ConTEXt of perl gehoord hebben, willen we uitdrukkelijk een zo simpel mogelijke distributie. Ons doel is daarom dan ook:

- □ Eén enkel uitvoerbaar programma voor het produceren van de PDF.
- □ Met alle benodigde 'make' functionaliteit ingebakken, zodat een éénmalige aanroep volstaat.

- Configuratie door middel van een `.ini` bestand. Het voordeel van het `.ini` bestandsformaat is dat het veel makkelijker te begrijpen is dan de huidige (van de Unix shell afgeleide) syntax van `texmf.cnf`.
- We hebben graag een minimaal aantal instellingen in dit bestand. Instellingen die alleen voor experts begrijpelijk zijn willen we zoveel mogelijk verwijderen.
- We willen slechts een handjevol data-archieven verspreiden in plaats van een uitgebreide TDS boomstructuur met (tien)duizenden losse bestandjes.
- Het eindresultaat moet zo snel mogelijk zijn, voor gebruik in batch- en CGI-achtige omgevingen.

Uitvoering in stappen

Het belangrijkste eerste doel is het geschikt maken van de verschillende onderdelen van het toekomstige programma voor gebruik buiten de Web2C/TEX-live omgeving.

Losweken uit Web2c

Met name `kpathsea` is nauw verbonden met de Web2C compilatie-omgeving. Er waren wat trucs nodig om te zorgen dat de kpathsea-bibliotheek wilde compileren buiten de TEX-live boom. Het is gelukt, maar de huidige situatie is verre van elegant.

Daarnaast waren Siep Kroonenberg's patches voor de MinGW GNU C-Compiler nodig om een versie voor Microsoft Windows te kunnen compileren.

Conversie

Het aanmaken van één enkele executable is uiteraard een stuk eenvoudiger als de diverse programma's gebruik maken van dezelfde programmeertaal.

Gezien onze expertise en onze wensen was er feitelijk maar één optie beschikbaar: de programmeertaal C. Alle andere mogelijkheden zouden ons te veel moeite kosten om te leren óf zouden een veel te inefficiënt eindresultaat opleveren.

Een groot gedeelte van de gewenste 'make' en commando-regel functionaliteit was reeds beschikbaar in het perl script texexec, en we wisten al dat we ConTEXt als format zouden gaan gebruiken dus texutil was ook nodig. pdfetex was verreweg de beste TEX voor ons doel, en metapost zal worden gebruikt voor het aanmaken van plaatjes.

Omwille van het vereenvoudigen van ons productie-proces hebben we besloten eerst losstaande versies (in de programmeertaal C) te maken van de gewenste programma's. Een bonus daarvan is dat er een tussenresultaat beschikbaar is.

Texutil

Texutil wordt een vrij op zichzelf staand onderdeel van het uiteindelijke programma, en daarom is er vrij veel tijd besteed aan de conversie. De functionaliteit is identiek aan TeXUtil 8.2, maar ten opzichte van de perl implementatie zijn er een aantal flinke interne wijzigingen doorgevoerd.

- Er is een 'functie-bibliotheek' afgesplitst. Die is er vooral voor Win32/Unix unificatie (bewerkingen op bestanden en mappen, geheugen-allocatie).
- Er is langzamerhand ook een 'perl emulatie-bibliotheek' ontstaan. Het was voor een aantal perl functies vrij eenvoudig om generieke C code te schrijven die de feitelijke conversie simpeler maakte (zoals voor stringoperaties en het werken met hashes).
- Verwijdering van globale variabelen. Alle subroutines hebben als eerste argument een object dat een instance is van '`struct texutilstruct*`'. Die enkele struct bevat alle oorspronkelijke globale variabelen.
- Afsplitsing van een `main()`-functie die slechts zorg draagt voor de commandoregel-opties, zodat de hoofdmoot van het programma op triviale wijze beschikbaar is als een 'texutil-bibliotheek'.

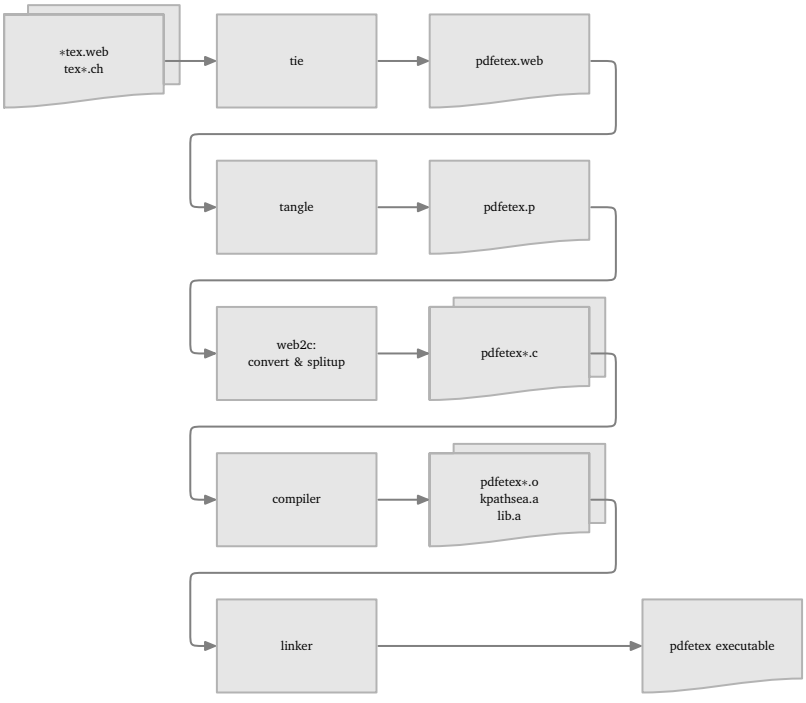
Texexec

Conversie van texexec naar C was redelijk eenvoudig. Omdat er tijdens deze conversie de geconverteerde versie van texutil al beschikbaar was, is er voor gekozen om rechtstreeks te linken tegen de 'texutil-bibliotheek', maar verder is het resultaat een getrouwe kopie van de functionaliteit van TeXExec 4.0.

Een interessant verschijnsel was dat juist de conversie van het help-systeem vrij ingewikkeld bleek. Dat is opmerkelijk omdat de help juist gebruik maakt van een perl module (Struct.pm) die uitdrukkelijk een C-constructie emuleert.

Pdfetex

Nu komen we bij de hoofdmoot van de werkzaamheden tot-nu-toe: de conversie van pdfetex. Hieronder zie je ter introductie het stroomdiagram van de compilatie van de Web2C-versie van TEX.



Voor ons doel was de handigste plaats om in te grijpen net na de 'tie' stap. Om uit te leggen waarom dit makkelijker is dan na de 'web2c' stap, moeten we een stukje voorbeeldcode laten zien. Als voorbeeld volgt nu de definitie van een van TEX's "Basic printing procedures".

De originele web programmacode voor de functie `print_cs`, die commando's en macro-namen toont bevat een paar web-commando's voor het aanmaken van de index (`@.XXX@>`) en de indentatie is niet echt fantastisch, maar verder is de pascal code redelijk makkelijk te volgen:

```
procedure print_cs(@!p:integer); {prints a purported control sequence}
begin if p<hash_base then {single character}
  if p>=single_base then
    if p=null_cs then
```

```
      begin print_esc("csname"); print_esc("endcsname");
      end
    else  begin print_esc(p-single_base);
      if cat_code(p-single_base)=letter then print_char(" ");
      end
  else if p<active_base then print_esc("IMPOSSIBLE.")
@.IMPOSSIBLE@>
  else print(p-active_base)
else if p>=undefined_control_sequence then print_esc("IMPOSSIBLE.")
else if (text(p)<0)or(text(p)>=str_ptr) then print_esc("NONEXISTENT.")
@.NONEXISTENT@>
else  begin print_esc(text(p));
  print_char(" ");
  end;
end;
```

De pascal code die wordt gegenereerd door tangle ziet er als volgt uit:

```
{:65}{262:}procedure printcs(p:integer);
begin if p<514 then if p>=257 then if p=513 then begin printesc(537);
printesc(538);end else begin printesc(p-257);
if eqtb[13636+p-257].hh.rh=11 then printchar(32);
end else if p<1 then printesc(539)else print(p-1)else if((p>=12526)and(p
<=16050))or(p>eqtbtop)then printesc(539)else if(hash[p].rh>=strptr)then
printesc(540)else begin printesc(hash[p].rh);printchar(32);end;end;
```

En dat is helemaal niet meer makkelijk te lezen, laat staan makkelijk te editen. Er zijn een aantal problemen met de leesbaarheid van deze pascal code.

1. Alle strings zijn vervangen door nummers (die nummers verwijzen naar het pool-bestand).
2. Alle 'symbolische constanten' (zoals `hash_base`) zijn vervangen door hun numerieke waarde.
3. Tangle's eindresultaat bevat net genoeg witruimte om de code eenduidig te maken, niets extra's.
4. De underscores uit het originele programmafragment zijn allemaal verdwenen.

Gelukkig voegt het Web2C programma convert weer wat witruimte toe, maar dat is niet meer goed genoeg om de situatie te redden. Het eindresultaat in C code van de 'web2c' stap ziet er zo uit:

```
void
#ifdef HAVE_PROTOTYPES
zprintcs ( integer p )
#else
zprintcs ( p )
  integer p ;
#endif
{
  printcs_regmem
  if ( p < 514 )
  if ( p >= 257 )
  if ( p == 513 )
  {
    printesc ( 537 ) ;
    printesc ( 538 ) ;
  }
```

```
  else {
    printesc ( p - 257 ) ;
    if ( eqtb [13636 + p - 257 ].hh .v.RH == 11 )
    printchar ( 32 ) ;
  }
  else if ( p < 1 )
  printesc ( 539 ) ;
  else print ( p - 1 ) ;
  else if ( ( ( p >= 12526 ) && ( p <= 16050 ) ) - ( p > eqtbtop ) )
  printesc ( 539 ) ;
  else if ( ( hash [p ].v.RH >= strptr ) )
  printesc ( 540 ) ;
  else {
    printesc ( hash [p ].v.RH ) ;
    printchar ( 32 ) ;
  }
}
```

Vanwege de slechte leesbaarheid van de door tangle/web2c gegenereerde code is er voor gekozen om de hele conversie van Web code naar C-code handmatig te doen. Dat was een heel karwei, maar het eindresultaat ziet er nu zo uit:

```
void print_cs (int p) { /* prints a purported control sequence */
  if (p < hash_base) { /* single character */
    if (p >= single_base) {
      if (p == null_cs) {
        print_esc_string ("csname");
        print_esc_string ("endcsname");
      } else {
        print_esc (p - single_base);
        if (cat_code (p - single_base) == letter)
          print_char (' ');
      }
    } else {
      if (p < active_base) { print_esc_string ("IMPOSSIBLE."); }
      else                 { zprint (p - active_base);         }
    }
  } else {
    if ((p >= undefined_control_sequence) && (p <= eqtb_size)) {
      print_esc_string ("IMPOSSIBLE.");
    } else {
      if (text (p) >= str_ptr){
        print_esc_string ("NONEXISTENT.");
      } else {
        print_esc (text (p)); print_char (' ');
      }
    }
  }
}
```

En dat vond ik zelf er een stuk prettiger uitzien. De ‘nieuwe’ procedures die op `_string` eindigen zijn ontstaan omdat dankzij die functies het `pool` bestand opgeheven kon worden. Een extra voordeel van de conversie ‘met de hand’ is dat de Web-commentaren zijn blijven bestaan.

De huidige C code bestaat uit ongeveer 50.000 regels geconverteerde C en commentaren, in zo’n 70 bronbestanden met elk een eigen headerbestand. De pdfetex

Web-code waar we mee begonnen zijn is gebaseerd op pdfetex versie 1.11b, die gebruik maakte van web2c 7.5.2.

Het grootste deel van de extra functionaliteit die werd aangeboden door het Web2C-systeem is niet beschikbaar in de CXTEX-versie. Een paar onderdelen daarvan zijn heel doelbewust niet geconverteerd, zoals de volgende opties:

- `-translate-file`
  CXTEX gebruikt momenteel altijd 8-bit invoer. Uiteindelijk zal encTEX beschikbaar komen ter vervanging van deze optie.
- `-src-specials`
  Het gebruik van source specials is specifiek verbonden met DVI-uitvoer. De CXTEX zal echter geen DVI previewer bevatten waardoor deze optie zinloos wordt.
- `-ipc`
  Deze optie was alleen beschikbaar als hij expliciet was aangezet tijdens het compileren van de Web2c-versie, en werkte slechts in combinatie met DVI-uitvoer.
- `[-no]-mktex=FMT`
  De mktexXX-scripts worden nooit aangeroepen door CXTEX, dit is alvast in voorbereiding op het verdwijnen van de kpathsea-bibliotheek.

En een paar andere Web2C-opties zijn gewoonweg niet vertaald, maar zonder een uitdrukkelijke reden anders dan gebrek aan (werk)tijd. De onderstaande opties komen daarom wellicht later weer terug:

`-parse-first-line`
`-file-line-error-style`
`-recorder`
`-shell-escape` (staat momenteel altijd aan)
`-output-comment`

Er zijn nog wat andere verschillen met Web2C:

- De hash is bevroren op $(2^{20} - 2^{12} = 1.044.480)$ mogelijke commando's.
- Er wordt dynamische her-allocatie gebruikt voor een aantal van TEX's interne structuren. Op dit moment zijn dat:
  - `buf_size`
  - `nest_size`
  - `save_size`
  - `pool_size`
  - `max_strings`

  en vrijwel alle andere structuren zullen nog gedaan worden.
- Natuurlijk is de banner aangepast (Een CXTEX versienummer is toegevoegd aan de toch al lange rij versienummers).
- De TFM-lees-code is geoptimaliseerd.
- Er is geen pool file.
- Format files zijn niet uitwisselbaar met de Web2c-versies. Om toch toe te staan dat beide executables bestaan binnen één `texmf`-bestandsstructuur, hebben de CXTEX formats als extensie `efm`. Dit is een tijdelijke oplossing, het probleem zal uiteindelijk vanzelf verdwijnen dankzij een toekomstige aanpassing in de TEX Directory Structuur (TDS).

Voor de toekomst

- □ Conversie van metapost naar C en het geschikt maken van de metapost broncode om gebruikt te kunnen worden als bibliotheek-subroutine. Werk hieraan is inmiddels begonnen.
- □ Vervangen van de huidige kpathsea door een privé-bibliotheek of door de nieuwe generatie kpathsea waar Olaf Weber momenteel aan werkt.
- □ Generalisatie van de commando-regel.
- □ Creëren van de feitelijke CXTEX-executable.
- □ Maken van een grafische font installatie- en configuratietool

Wie wat waar?

De Homepage is http://tex.aanhet.net/cxtex.

Er staan op de website archieven van de bronbestanden en de Win32 executables van de huidige versie (0.5.1). Zelf compileren onder diverse unices hoort vlekkeloos te verlopen, de enige vereiste is een relatief nieuwe versie van gcc en g++ (versie 2.96 of hoger). Aan ondersteuning voor MacOS X wordt gewerkt.

Taco Hoekwater

The Scite – TEX integration

Abstract
Editors are a sensitive, often emotional subject. Some editors have exactly the properties a software designer or a writer desires and one gets attached to it. Still, most computer experts such as TEX users often are use three or more different editors each day. Scite is a modern programmers editor which is very flexible, very configurable, and easily extended. We integrated Scite with TEX, CONTEXT, LATEX, METAPOST and viewer and succeeded in that it is now possible to design and write your texts, manuscripts, reports, manuals and books with the Scite editor without having to leave the editor to compile and view your work. The article describes what is available and what you need with special emphasis on highlighting commands with lexers.

About Scite

Scite is a source code editor written by Neil Hodgson. After playing with several editors we found that this editor is quite configurable and extendible. At PRAGMA ADE we use TEXEDIT, an editor written long ago in Niklaus Wirth's MODULA as well as a platform independent reimplementation of it called TEXWORK written in PERL/TK. Although our editors possess some functionality that is not (yet) present in Scite, we decided to use Scite because it frees us from the editor maintenance chore.

Installing Scite

Installing Scite is straightforward. We assume below that you use MS WINDOWS but for other operating systems installation is not much different.

First you need to fetch the archive from:

```
www.scintilla.org
```

The MS WINDOWS binaries are in `wscite.zip`, and you can unzip this in any directory as long as the binary executable ends up in your `PATH` or as shortcut icon on your desktop.

The TEX lexer

Scite provides several so called 'lexers'. A lexer does the syntax highlighting of your document. The TEX commands that are used in instructing the compiler on how to typeset your document can be given a color. And the various brackets that need to be balanced, such as '`{ } [ ] ( )`' also get a color. The way a TEX file is treated is configurable and can be found in the file:

```
tex.properties
```

You can edit this file to your needs using the menu entry under `options` in the top bar of Scite. In this file, the following settings apply to the TEX lexer:

```
lexer.tex.interface.default=0
lexer.tex.use.keywords=1
lexer.tex.comment.process=0
lexer.tex.auto.if=1
```

The option `lexer.tex.interface.default` for example determines the way TEX keywords are highlighted. You can control the interface from your document as well by placing the line below as the first line in it. This often makes more sense than editing the configuration file to your incidental needs.

```
% interface=all|tex|nl|en|de|cz|it|ro|latex
```

The values in the properties file and the keywords in the preamble line above have the following meaning:

0	`all`	all commands (preceded by a backslash)
1	`tex`	TEX, ε-TEX, PDFTEX, OMEGA primitives (and macros)
2	`nl`	the dutch CONTEXT interface
3	`en`	the english CONTEXT interface
4	`de`	the german CONTEXT interface
5	`cz`	the czech CONTEXT interface
6	`it`	the italian CONTEXT interface
7	`ro`	the romanian CONTEXT interface
8	`latex`	LATEX (apart from packages)

The configuration file is set up in such a way that you can easily add more keywords to the lists. The keywords for the second and higher CONTEXT command language and LATEX interfaces are defined in their own properties files:

```
cont-nl-scite.properties
...
cont-en-scite.properties
latex-scite.properties
```

The CONTEXT distribution comes with a file:

```
context.properties
```

as well as the interface specific files. There are two ways to make sure that the extra CONTEXT and LATEX keywords are loaded.

Under the menu entry 'options', you will find a long list of property filenames. By opening one of them, you can determine the location of these files by looking at your windows banner (at the top). You should copy the following files to that path, perhaps named `c:\scite\wscite`.

```
cont-*-scite.properties
latex-scite.properties
```

The first line `*-scite.properties` files define the CONTEXT keywords, and the second line configures Scite for LATEX. If you need another brand of TEX file to be processed, you can tweak the properties, or safer: redefine some of them in your `SciTEUser.properties`. For example plain TEX users may want:

```
file.patterns.tex=*.tex;*.sty;
filter.tex=TeX|$(file.patterns.tex)|
lexer.$(file.patterns.tex)=tex
command.compile.$(file.patterns.tex)=
command.build.$(file.patterns.tex)=tex $(FileNameExt)
command.go.$(file.patterns.tex)=gv $(FileName).pdf
```

LATEX users may want:

```
file.patterns.latex=*.tex;*.sty;*.aux;*.toc;*.idx;
filter.latex=LaTeX|$(file.patterns.latex)|
lexer.$(file.patterns.latex)=tex
command.compile.$(file.patterns.latex)=
command.build.$(file.patterns.latex)=pdflatex $(FileNameExt)
command.go.$(file.patterns.latex)=gv $(FileName).pdf
```

A CONTEXT user needs:

```
file.patterns.context=*.tex;*.tui;*.tuo;*.sty;
filter.context=ConTeXt|$(file.patterns.context)|
```

```
lexer.$(file.patterns.context)=tex
command.compile.$(file.patterns.context)=
command.build.$(file.patterns.context)=texexec --pdf $(FileNameExt)
command.go.$(file.patterns.context)=gv $(FileName).pdf
```

For CONTEXT users these definitions are already assembled in a special properties file:

```
context.properties
```

Put it in the same location as your `SciTEUser.properties`, and in this user file add the following line:

```
import context
```

Now you have many more commands available. Familiarize yourself with these new options and take a look into this file. Beware: this setup assumes that you have the Latin Modern Typewriter font on your system and that your operating system is aware of that. If Windows is unaware then locate the Latin Modern Fonts in the `texmf-extra` tree on the latest TEX Live distribution of the TEX user groups. Install the Latin Modern files on Windows by copying the `pfb` and `pfm` files to the fonts path on your system, in most cases this is:

```
c:\windows\fonts
```

You can add or locally change options after the line that loads `context.properties`. If you did not copy the `cont-*.properties` files to the Scite properties path, you can put them in the same path as `SciTEUser.properties`, in which case you have to add:

```
import latex-scite
import context
```

Try it and find out that Scite can easily be tuned to your preferences.

The instructions for context users are:

fetch `wscite.zip` from `www.scintilla.org`
create a path `c:\scite`
unzip `wscite.zip` in `c:\scite`
copy `context.properties` to `c:\wscite`
open `c:\scite\wscite\SciTEUser.properties` (using Scite)
at the end, add the line `import context.properties`
if needed, add `c:\scite\wscite` to your path
if needed, create shortcut to `c:\scite\wscite\scite.exe`

The CONTEXT related properties files are not included in the Scite distribution but instead are part of the CONTEXT distribution which can be found in one of the following places:

```
../tex/texmf/context/data
../tex/texmf-local/context/data
```

We generate the interface specific property files automatically from the CONTEXT interface definition files, while the `xx` file (in the CONTEXT zip file) is hand–crafted and contains missing or very special bits and pieces.

Let us return to the powerful properties options in `tex.properties`. For testing purposes you can disable keyword coloring with:

```
lexer.tex.use.keywords=0
```

You can also influence the way comment is treated with:

```
lexer.tex.comment.process=0
```

When set to zero, comment is not interpreted as TEX code and it will come out in a uniform color. But, when set to one, you will get as much color as a TEX source. The lexer tries to cope with the TEX syntax as well as possible and takes care of the `^^` notation. A special treatment gets `\if`:

```
lexer.tex.auto.if=1
```

This is the default setting. When set to one, all `\ifwhatever`'s will be seen as a command. When set to zero, only the primitive `\if` will be treatedas such. When this property is set to one, the lexer will not color an `\ifwhatever` that follows an `\newif`.

The METAPOST lexer

The METAPOST lexer is set up slightly differently from its TEX counterpart, because METAPOST is more a true language than TEX is although, as with the TEX lexer, we can control the interpretation of identifiers. The METAPOST specific configuration file is:

```
metapost.properties
```

Here are located properties like:

```
lexer.metapost.interface.default=1
```

Instead of editing the configuration file you can control the lexer with the first line in your document:

```
% interface=none|metapost|mp|metafun
```

The numbers and keywords have the following meaning:

0	`none`	no highlighting of identifiers
1	`metapost or mp`	METAPOST primitives and macros
2	`metafun`	MetaFun macros

Similar to the TEX lexer, you can influence the way comments are handled:

```
lexer.metapost.comment.process=1
```

This will interpret comment as METAPOST code, which is not that useful (opposite to TEX, where documentation is often coded in TEX).

The lexer will color the METAPOST keywords and — when enabled — additional keywords (like those of MetaFun) also, and shown in a slanted font. These MetaFun keywords are defined in yet another separate file:

```
metafun-scite.properties
```

You can either copy this file to the path where your global properties files is located, or put a copy in the path of your user properties file. In that case you again need to add an entry to in file `SciTEUser.properties`:

```
import metafun-scite
```

The lexer recognizes `btex ... etex` pairs and will treat anything in between as just text. The same happens with strings (placed between `"`). Both act on a per line basis.

Epilogue. This completes our special TEX installation instructions for the Scite integrating editor. We believe Scite to be a well designed, flexible editor allowing the perfect integration of TEX and METAPOST systems. An hour or less suffices to discover the convenience of this system. The next hour shows some minor weak points on which we are working to reinforce them.

Hans Hagen
pragma@wxs.nl

Introducing oldstyle figures in existing virtual fonts*

Abstract
This paper describes a *Ruby* script *osf* that can be used to make a copy of a virtual font with its figures replaced with old style figures.

Keywords
ruby script txfonts pxfonts oldstyle figures

Introduction

When I was typesetting a biography I chose palatino for its font because I like palatino, especially for texts about history. I could have used the `palatino` package, but I don't like its math, nor its typewriter face, so I normally use the `pxfonts` package instead. However, both packages use table figures while, especially in palatino, old style would fit a lot better.

Reading the `pxfonts` documentation, I was happy to see that it contains tables with Text Companion Fonts, with old style figures. However, although the tables are there, the documentation doesn't say anything about how to get such characters in your document. Inspection of the style file `pxfonts.sty`, also, shows that it ignores these fonts.

So I asked the gurus: how do I get old style figures with `pxfonts`? And I was showered with font jargon, most of which I had heard about, but never really understood. But what I distilled from it was: have a look at virtual fonts–you can export them from the binary `.vf` files with `vftovp`, edit the resulting `.vpl` file and then convert that back to `.vf` with `vptovf`.

Virtual font files

So, for a start, I had a look at the virtual font files for the *Roman Upright* fonts, `pxr.vf`, with table figures, and the *Text Companion Roman Upright* fonts, `pcxr.vf`, with old style figures. I did so by running:

```
vftovp pxr pxr pxr
vftovp pcxr pcxr pcxr
```

*Without the help of Siep Kroonenberg on font matters this article would never have seen daylight

As you can read in the `vftovp` documentation, this converts the information in pxr.vf and pxr.tfm into the readable and editable *Virtual Property List*: `pxr.vpl` in your current directory; similarly for `pcxr`. Here are aligned excerpts of the two files:

```
 1  (VTITLE PXR)                  (VTITLE PCXR)
 2  (FAMILY PXR)                  (FAMILY UNSPECIFIED)
 3  (FACE F MRR)                  (FACE F MRR)
 4  (CODINGSCHEME TEX TEXT)      (CODINGSCHEME TEX TEXT CO...
 5  (DESIGNSIZE R 10.0)           (DESIGNSIZE R 10.0)
 6  (COMMENT DESIGNSIZE IS IN ... (COMMENT DESIGNSIZE IS IN...
 7  (COMMENT OTHER SIZES ARE M... (COMMENT OTHER SIZES ARE
 8  (CHECKSUM O 22721014137)      (CHECKSUM O 32012256317)
 9                                (SEVENBITSAFEFLAG TRUE)
10  (FONTDIMEN                    (FONTDIMEN
11     (SLANT R 0.0)                 (SLANT R 0.0)
12     (SPACE R 0.25)                (SPACE R 0.25)
13     (STRETCH R 0.2)               (STRETCH R 0.2)
14     (SHRINK R 0.1)                (SHRINK R 0.1)
15     (XHEIGHT R 0.469)             (XHEIGHT R 0.469)
16     (QUAD R 1.0)                  (QUAD R 1.0)
17     (EXTRASPACE R 0.1)            (EXTRASPACE R 0.1)
18     )                             )
19  (MAPFONT D 0                  (MAPFONT D 0
20     (FONTNAME RPXPPLR)            (FONTNAME RPCXR)
21    (FONTCHECKSUM O 3657114...    (FONTCHECKSUM O 450707...
22     (FONTAT R 1.0)                (FONTAT R 1.0)
23     (FONTDSIZE R 10.0)            (FONTDSIZE R 10.0)
24     )                             )
25  (MAPFONT D 1                  (MAPFONT D 1
26     (FONTNAME RPXR)               (FONTNAME RPXPPLR)
27    (FONTCHECKSUM O 2703202...    (FONTCHECKSUM O 365711...
28     (FONTAT R 1.0)                (FONTAT R 1.0)
29     (FONTDSIZE R 10.0)            (FONTDSIZE R 10.0)
30     )                             )
31  (LIGTABLE                     [... no ligtable ...]
32
33  [... lots of lines ...]       [... lots of lines ...]
34
35     )
36  (CHARACTER O 0                (CHARACTER O 0
37     (CHARWD R 0.556)              (CHARWD R 0.332996)
38     (CHARHT R 0.686247)           (CHARHT R 0.685245)
39     (CHARDP R 0.00475)
40     (MAP                          (MAP
41        (SELECTFONT D 1)              (SELECTFONT D 1)
42        (SETCHAR O 0)                 (SETCHAR O 36)
43        )                             )
44     )                             )
45
46  [... octal 1..57 ...]         [... octal 1..57 ...]
47
48  (CHARACTER C 0                (CHARACTER C 0
```

```
49     (CHARWD R 0.5)                (CHARWD R 0.5)
50     (CHARHT R 0.686247)           (CHARHT R 0.476)
51     (CHARDP R 0.017999)           (CHARDP R 0.011)
52     (MAP                          (MAP
53        (SETCHAR C 0)                 (SETCHAR C 0)
54        )                             )
55     )                             )
56   (CHARACTER C 1                (CHARACTER C 1
57     (CHARWD R 0.5)                (CHARWD R 0.5)
58     (CHARHT R 0.702999)           (CHARHT R 0.476)
59     (CHARDP R 0.00475)            (CHARDP R 0.011)
60     (MAP                          (MAP
61        (SETCHAR C 1)                 (SETCHAR C 1)
62        )                             )
63     )                             )
64
65   [... digits 2..8 ...]         [... digits 2..8 ...]
66
67   (CHARACTER C 9                (CHARACTER C 9
68     (CHARWD R 0.5)                (CHARWD R 0.5)
69     (CHARHT R 0.686247)           (CHARHT R 0.476)
70     (CHARDP R 0.017999)           (CHARDP R 0.237247)
71     (MAP                          (MAP
72        (SETCHAR C 9)                 (SETCHAR C 9)
73        )                             )
74     )                             )
75
76   [... more characters ...]     [... more characters ...]
```

These virtual property listings look like lisp code. Their structure is described in the source of Donald Knuth's `vptovf` program; you can generate the documentation with:

```
weave vptovf.web
pdftex vptovf
```

You can find `vptovf.web` on the TEXLive Collection's DVD in the directory `ctan/systems/knuth/etc`.

The virtual property listings start with some general comments and dimensions (lines 1–18), followed by one or more font mapping sections (lines 19–30), a ligature table (lines 31–35, not in `pcxr.vpl`) and finally one section for each defined character, starting with octal 0 (line 36–44) and potentially ending with octal 377. Characters are identified with their octal number (like `CHARACTER O 0` on line 36), except for the digits and letters, which are identified by themselves (like `CHARACTER C 0` for the digit 0 on line 48).

A character section defines width, height and depth for the corresponding glyph (lines 37–39 for octal 0 for example) and it tells from which font it is taken (line 41) and from which position in that font (line 42). Thus the glyph for character octal 0 in `pxr` is taken from position 0 of font 1, which points to the font `rpxr` on line 26 in the second font mapping section. Similarly, the glyph for octal 0 of `pcxr` is taken from font `rpxpplr`, position 0.

Now what we are interested in is the glyphs for the digits. In both virtual fonts, `pxr` and `pcxr`, these are taken from the default font, because there is no `SELECTFONT` statement in their character sections. This default is the first font, which means `rpxpplr` for `pxr` and `rpcxr` for `pcxr`. This made me think that what I probably had to do was to take the first `MAPFONT` section (pointing to `rpcxr`) from the old style virtual font `pcxr`, and add it to the `pxr` virtual font to give it a third `MAPFONT` section. I simply appended it to the end of the `.vpl` file. Of course, it needed a new unique identifier, so I changed `MAPFONT D 0` into `MAPFONT D 2`. Furthermore, I removed the character sections of the digits, and appended the (old style) digit character sections taken from `pcxr.vpl`. These had of course to point to the new `MAPFONT` section, so I added `SELECTFONT D 2` to each of them.

Finally, I converted the new `pxr.vpl` back into `pxr.vf` and `pxr.tfm`:

```
vptovf pxr pxr pxr
```

and I made a little test file:

```
\documentclass{article}
\usepackage{pxfonts}
\begin{document}
   Hello number 0123456789!
\end{document}
```

And it worked!

Editing `.vf` files with a script

Editing a `.vf` file as described takes quite some time. Moreover, we need to edit more than a single file if we want old style figures to appear not only in roman upright text, but also in bold, sans serif, slanted, italic and so on. In the case of `pxfonts`, this means editing 16 files, in the case of `txfonts` even 42 files!

Therefore I made a *Ruby* script to take over the work. The script is listed at the end of this article.[1] It's heart is a subroutine *convert_font*, which takes two arguments: the name of the font to be converted (say `pxr`) and the name of the font containing old style digits (say `pcxr`). In essence, it does te following:

- convert the table font to `.vpl`, saving it in a new `.vpl` file, `pxr.vpl` removing the digits on the fly and remembering the highest mapfont identifier in it.
- convert the old style font to `.vpl`, isolating it's map font and digit sections.
- append these to the new `pxr.vpl` file after fixing the map font identifier.
- convert the new `pxr.vpl` file to `pxr.vf` and `pxr.tfm` with `vptovf`.

The rest of the script handles the command line argument to give you some options:

with two arguments, the first should be a font to be converted, the second a font from which the old style digits should come from. So you can say:

```
osf pxr pcxr
```

which would create two files in the current directory: `pxr.vf` and `pxr.tfm`.

with one argument, it should be a font to be converted. The old style font from which the digits are to be taken will be searched in the directory where the converted font occurs. You will be presented a list of those fonts from which you can make your choice. There will be a default which the script thinks is most likely on the basis of its name. So the following would be a possible dialog:

```
osf pxr
I found 10 fonts with old style digits:
 1  3 pcxb
 2  4 pcxbi
 3  5 pcxbsl
 4  3 pcxi
 5  1 pcxr
 6  4 pcxsl
 7  4 pxbmi
 8  5 pxbmi1
 9  3 pxmi
10  4 pxmi1
My guess is 5 (pcxr)
Your guess [5]:
```

with no arguments you can make use of predefined combinations of fonts and their old style companions. Currently, two combinations are defined in the DATA section of the script: `pxfonts` and `txfonts`. The following would be a possible dialog:

```
osf
2 font conversions have been predefined
1 pxfonts
2 txfonts
Please make your choice [1]:
Converting pxfonts
p1xb       pcxb
p1xbi      pcxbi
p1xbsc     pcxb
p1xbsl     pcxbsl
p1xi       pcxi
p1xr       pcxr
p1xsc      pcxr
p1xsl      pcxsl
pxb        pcxb
pxbi       pcxbi
pxbsc      pcxb
pxbsl      pcxbsl
pxi        pcxi
pxr        pcxr
pxsc       pcxr
pxsl       pcxsl
```

As a result, you would find `.vf` and `.tfm` files for all faces of `pxfonts` that may need conversion to old style.

Of course, storing all these files together with your LaTeX document is not very elegant, although you might go for this option occasionally. Another option would be to store these files in your user tree, or even in the local tree. But that would mean that you would get old style figures in *all* your documents, which is not necessarily what you want. A better alternative is to give your font a new name.

Renaming the font

This section will be dedicated to `pxfonts` and will describe how to create a new font, `osf-pxfonts`, from it, including a style file, `osf-pxfonts.sty` which gives you access to the same font families, series and shapes as does `pxfonts`, but with oldstyle figures for the roman and sans serif families. The encoding will be T1. It's easy to translate what you read here to other font collections.

You can give your font another name by adding a prefix (say `osf-`) to all names, and moving them to a new directory. However, you now have a problem: since you have changed the name, you cannot use the style file (`pxfonts.sty`) anymore. But fortunately, there is a solution for this: you can make a new style file (say `osf-pxfonts.sty`) which imports the original style file and then tweak it a little, like this:

```
\RequirePackage{pxfonts,t1enc}
\AtBeginDocument{%
  \usefont{T1}{osf-pxr}{m}{n}
  \renewcommand{\sfdefault}{osf-pxss}
}
```

This defines a style file which uses T1 encoding and oldstyle figures for the roman (`osf-pxr`) and sans serif (`osf-pxss`) fonts. However, this is not enough: LaTeX uses font definition (`.fd`) files, one for each font family, to map fonts to various series (weights and widths) and shapes (normal, italic, slanted, small caps). These font definition files are named after, and refer to, the fonts we just renamed, so we need renamed copies of these files and we also need to rename the font references inside them.

This is not really complicated. We need new font definition files for only two families: roman and sans serif. Since we used the T1 encoding for our style file, we can confine ourselves to `t1pxr.fd` for the roman

fonts and t1pxss.fd for the sans serif fonts. We rename these to t1osf-pxr.fd and t1osf-pxss.fd[2].

If you have a look inside t1pxr.fd, you easily recognize the font names that have been listed when you ran the script to convert the pxfonts and for which new font files were produced with osf-prefixed names. So all there is to be done is to add osf- before all these names; and since all names start with pxr or p1x we can simply replace pxr with osf-pxr and p1x with osf-p1x everywhere.

The same can be done for t1pxss.fd; here however, we see that the sans serif fonts are actually borrowed from the txfonts. This means that for the pxfonts we need to convert the txfonts as well if we want to have oldstyle figures in sans serif. And, of course, we need to substitute pxss with osf-pxss and t1x (not p1x) with osf-t1x in the .fd file.

Now the good news is that for pxfonts and txfonts, predictable as it is, you don't have to do this editing yourself: the script does it for you and creates the necessary .fd files.

Switching between oldstyle and table figures

Although you may like oldstyle figures, there are occasions, even within one document, where you might like to switch back to table figures. For example, a table listing many numbers does not look very good with oldstyle figures.

As the style file presented above already suggests, it is easy to make switching between oldstyle and table figures possible, either by defining options to the package or, for switching on the fly, by defining switching commands. The following example does both. In addition, instead of boldly switching fonts with \usefont, it first checks which family is currently in effect and changes the font accordingly, so that one can say, for example, \bfseries\itshape\osfigures without the \osfigures resetting series and shape to the defaults:

```
\RequirePackage{pxfonts,t1enc}

\def\tb@rm{pxr}
\def\tb@sf{pxss}
\def\osf@rm{osf-pxr}
\def\osf@sf{osf-pxss}

\newcommand{\osfigures}{%
  \renewcommand{\rmdefault}{\osf@rm}%
  \renewcommand{\sfdefault}{\osf@sf}%
  \ifx\f@family \tb@rm \rmfamily\fi
  \ifx\f@family \tb@sf \sffamily\fi}

\newcommand{\tbfigures}{%
  \renewcommand{\rmdefault}{\tb@rm}%
  \renewcommand{\sfdefault}{\tb@sf}%
  \ifx\f@family \osf@rm \rmfamily\fi
  \ifx\f@family \osf@sf \sffamily\fi}

\DeclareOption{osfigures}{
   \osfigures
   \let\@Fam\osfigures
}
\DeclareOption{tbfigures}{
   \tbfigures
   \let\@Fam\tbfigures
}
\ExecuteOptions{osfigures}
\ProcessOptions
\AtBeginDocument{\@Fam}
```

For the predefined fonts, the script creates this style file also for you.

Configuration

Once having created a full-blown set of files for oldstyle pxfonts and txfonts, I thought I could as well let the script move the files to directories in the TEX-tree where LATEX expects them. So when converting predefined fonts, that is: when run with no arguments, the script will store the files in the usual subdirectories in the user tree ($HOME/texmf). And before finishing, the script will run mktexlsr, so that the files will be found.

Testing

You will probably want to test your newly created font and verify that the series- and shape-switching commands work. Here is a test source that does so for you– just change the second line if you want to test other fonts:

```
\documentclass{article}
\usepackage{osf-txfonts}
\parindent0pt
\def\text{Hello Wörld! 0123456789 }
\newcommand{\test}[2]{%
  \def\F{Sans}\def\Arg{#2}
  \ifx\Arg\F\let\F\sf\else\let\F\relax\fi
  \begin{tabular}{ll}
    \multicolumn{2}{l}{#1 #2}\\\hline
    normal:       &\F\text\\
    slanted:      &\F\textsl{\text}\\
    italic:       &\F\textit{\text}\\
    small caps:   &\F\textsc{\text}\\
    bold normal:  &\F\textbf{\text}\\
    bold italic:  &\F\textbf{\textit{\text}}\\
    bold slanted: &\F\textbf{\textsl{\text}}\\
```

```
    bold small caps:&\F\textbf{\textsc{\text}}\\
  \end{tabular}\\[2ex]
}
\pagestyle{empty}

\begin{document}
\test{Oldstyle}{Roman}
\test{Oldstyle}{Sans}
\tbfigures
\test{Table}   {Roman}
\test{Table}   {Sans}
\end{document}
```

Here is its output:

Oldstyle Roman	
normal:	Hello Wörld! 0123456789
slanted:	*Hello Wörld! 0123456789*
italic:	*Hello Wörld! 0123456789*
small caps:	Hello Wörld! 0123456789
bold normal:	**Hello Wörld! 0123456789**
bold italic:	***Hello Wörld! 0123456789***
bold slanted:	***Hello Wörld! 0123456789***
bold small caps:	**Hello Wörld! 0123456789**

Oldstyle Sans	
normal:	Hello Wörld! 0123456789
slanted:	*Hello Wörld! 0123456789*
italic:	*Hello Wörld! 0123456789*
small caps:	Hello Wörld! 0123456789
bold normal:	**Hello Wörld! 0123456789**
bold italic:	***Hello Wörld! 0123456789***
bold slanted:	***Hello Wörld! 0123456789***
bold small caps:	**Hello Wörld! 0123456789**

Table Roman	
normal:	Hello Wörld! 0123456789
slanted:	*Hello Wörld! 0123456789*
italic:	*Hello Wörld! 0123456789*
small caps:	Hello Wörld! 0123456789
bold normal:	**Hello Wörld! 0123456789**
bold italic:	***Hello Wörld! 0123456789***
bold slanted:	***Hello Wörld! 0123456789***
bold small caps:	**Hello Wörld! 0123456789**

Table Sans	
normal:	Hello Wörld! 0123456789
slanted:	*Hello Wörld! 0123456789*
italic:	*Hello Wörld! 0123456789*
small caps:	Hello Wörld! 0123456789
bold normal:	**Hello Wörld! 0123456789**
bold italic:	***Hello Wörld! 0123456789***
bold slanted:	***Hello Wörld! 0123456789***
bold small caps:	**Hello Wörld! 0123456789**

The script

This section presents a listing of the Ruby script. The first 100 lines are comment lines in `rdoc` format. You can convert those into `html` by running:

```
rdoc osf
```

This creates a subdirectory `doc`. Point to the file `index.html` inside it in your browser and you will see a nicely formatted page, which will also contain separate sections for all methods defined in the script. Clicking in the headers of those section shows their sources in a popup window.

```
#!/usr/bin/ruby

=begin rdoc

=osf - convert digits in virtual font files to oldstyle

==Synopsis

osf [virtual_font_name [replacing_font_name]]

==Description

*osf* converts one or more virtual font (<tt>.vf</tt> and
<tt>.tfm</tt>) files, replacing the digits with old style
variants.

There are two ways to run the script: with one or two arguments
or with no arguments at all.

===Running with argument(s)

The first argument, if any, is the name of the virtual font file
to be converted. If a second argument is present, it is assumed
to be the name of the virtual font file containing old style
digits. If no second argument is present, the directory where the
first argument's virtual font file lives is searched for other
virtual font files containing old style digits and you are
presented a list of those from which you can make a choice.

The converted virtual font (<tt>.vf</tt>) files are stored in
your working directory, together with the corresponding
<tt>.tfm</tt> files. As a result, TeX documents compiled in that
directory using the converting fonts (for example by using
\usepackage{pxfonts}) will produce output with old style digits.

===Running without any arguments

If the script is run without arguments, a list is presented of
predefined virtual font sets from which you can make your choice.
Currently these are either the txfonts or the pxfonts.

In this case, the new font collection is renamed by prefixing
file names with <tt>osf-</tt> and a new style file is created,
together with the necesssary font definition (<tt>.fd</tt>)
files; these files, too, are named after the original files by
prefixing them with <tt>osf-</tt>.

The new style file has two options to switch to oldstyle or
table figures:

osfigures::  start with oldstyle figures (this is the default)
tbfigures::  start with table figures

The style file also creates two commands with the same goal:
\osfigures:: switch to oldstyle figures
\tbfigures:: switch to table figures

```

```
== Testing your font
Here is a LaTeX source that can be used to test your changes to
the txfonts and the pxfonts:

  \documentclass{article}
  \usepackage{osf-pxfonts}
  \parindent0pt
  \def\text{Hello World! 0123456789 äëïöéèê}
  \newcommand{\test}[2]{
     \def\Fam{Sans}\def\Arg{#2}
     \ifx\Arg\Fam\let\Fam\sf\else\let\Fam\relax\fi
     \begin{tabular}{@{}p{4em}ll@{}}
        #1 & normal:          & \Fam\text\\
        #2 & slanted:         & \Fam\textsl{\text}\\
           & italic:          & \Fam\textit{\text}\\
           & small caps:      & \Fam\textsc{\text}\\
           & bold normal:     & \Fam\textbf{\text}\\
           & bold italic:     & \Fam\textbf{\textit{\text}}\\
           & bold slanted:    & \Fam\textbf{\textsl{\text}}\\
           & bold small caps: & \Fam\textbf{\textsc{\text}}\\
     \end{tabular}\vfill
  }
  \pagestyle{empty}

  \begin{document}
  \test{Oldstyle}{Roman}
  \test{Oldstyle}{Sans}
  \tbfigures
  \test{Table}   {Roman}
  \test{Table}   {Sans}
  \end{document}

  $$ y = x^{123} \mathrm{text}$$

  \end{document}

==Version
$Id: osf,v 1.5 2004/05/06 20:35:20 wybo Exp $

==Author
Wybo Dekker (<tt>wybo@servalys.nl</tt>)

=end

require 'ftools'

class Hash
   # return the sum of squares of the values of a hash
   def sum_of_squares
      s = 0
      self.each { |x,y| s += y*y }
      return s
   end
end

# Check if a file has oldstyle digits
# Typically, table digits have equal heights and zero depth
# Oldstyle digits 0, 1, 2, 6, and 8 have zero depth,
# while 3, 4, 5, 7, and 9 have significant depths

def has_oldstyle_digits(file)
   indigit = false
   ht = dp = 0
   dif = Array.new
   open("|vftovp #{file}").readlines.each { |line|
      case line
      when /CHARACTER C (\d)$/   then indigit = $1.to_i
      when /CHARHT.* ([\d\.]+)/  then ht = $1.to_f
      when /CHARDP.* ([\d\.]+)/  then dp = $1.to_f
      when /MAP/ then
         if indigit
            dif[indigit] = ht-dp
            indigit = false
         end
      end
   }
   if dif.size > 0
      if (dif[3]+dif[4]+dif[5]+dif[7]+dif[9])/
         (dif[0]+dif[1]+dif[2]+dif[6]+dif[8]) > 0.9
         return false
      else
         return true
      end
   else
      return false
   end
end

# print a message, then exit with a fatal error

def die(message)
   puts 'fatal: ' + message
   exit 1
end

# find basename and directory of a virtual font file

def findfont(name)
   font = `kpsewhich #{name}.vf`.chomp
   die "Could not find #{name}.vf" if font == ''
   return File.basename(font,'.vf'), File.dirname(font)
end

# from a virtual font, isolate the digit sections and the mapfont
# section defining them. Renumber the mapfont to fontnr and
# change the SELECTION commands in the digits to point to it

def find_mapfonts_and_digits(font,fontnr)
   vpl = open("|vftovp #{font}.vf")
   mapfonts = Array.new           # one of these is returned
   digits = Array.new             # all returned
   sel = -1                       # font used for digits
   while line = vpl.gets
      case line
      when /MAPFONT D (\d+)/ then    # mapfont section?
         i = $1.to_i
         mapfonts[i] = line
         while l = vpl.gets
            mapfonts[i] += l
            break if l =~ /^   \)/
         end
      when /CHARACTER C (\d)$/ then  # digit?
         i = $1.to_i
         digits[i] = line
         while l = vpl.gets
            if l =~ /SELECTFONT D (\d)/
               sel = $1.to_i
               l.sub!(/\d+/,fontnr.to_s)
            end
            digits[i] += l
            break if l =~ /^   \)/
         end
      end
   end
   if sel == -1 then # no SELECTONT found?
      sel = 0        # use the default
      # insert SELECTFONT command in the digits
      digits.each { |d|
         d.sub!(/MAP$/,"MAP\n      (SELECTFONT D #{fontnr})")
      }
   end
   # renumber the mapfont
   mf = mapfonts[sel].sub(/MAPFONT D.*/,"MAPFONT D #{fontnr}")
   return mf,digits
end

# convert digits in virtual font file to old style
# collection:   name of the font collection (like `pxfonts')
# font:         font with table figures to be converted
# oldstylefont: font containing oldstyle figures
```

```
208
209 def convert_font(collection,font,oldstylefont)
210    # collection undefined: save in current dir
211    prefix = ''
212    vfdir = tfmdir = '.'
213    # if collection is defined, save files in user tree
214    if collection
215       prefix = 'osf-'
216       d = "#{$textree}/fonts/@/osf-#{collection}"
217       vfdir = d.sub(/@/,'vf')
218       tfmdir = d.sub(/@/,'tfm')
219       File.mkpath(vfdir) or
220          die "Could not create directory #{dir}"
221       File.mkpath(tfmdir)
222    end
223    # the new vpl file:
224    newvpl = open("/tmp/#{$$}.vpl","w")
225    maxfont = 0
226    # read the vpl file to be corrected:
227    vpl = open("|vftovp #{font}")
228    while line = vpl.gets
229       case line
230       when /MAPFONT D (\d+)/ then
231          # remember the maximum font ident
232          maxfont = [maxfont, $1.to_i].max
233          newvpl.print line
234       when /CHARACTER C \d/ then           # digit?
235          while l = vpl.gets                # skip to...
236             break if l =~ /^   \)/         # ... end of digit
237          end
238          next
239       else
240          # print everything else to the new vpl:
241          newvpl.print line
242       end
243    end
244    # append map font and digits from the old style virtual font
245    newvpl.print find_mapfonts_and_digits(oldstylefont,maxfont+1)
246
247    vpl.close
248    newvpl.close
249    name = prefix + font
250    system <<-EOF
251       vptovf /tmp/#{$$} \
252       #{vfdir}/#{name}  \
253       #{tfmdir}/#{name} >/dev/null
254    EOF
255 end
256
257 # define the style file in terms of the names of:
258 # arg 1: the font collection (pxfonts, txfonts, ...)
259 # arg 2: the roman font (pxr, txr, ...)
260 # arg 3: the sans font (pxss, txss, ...)
261
262 def style(collection,roman,sans)
263    return <<-EOF.gsub(/^      /,'')
264       \\RequirePackage{#{collection},t1enc}
265
266       \\def\\tb@rm{#{roman}}
267       \\def\\tb@sf{#{sans}}
268       \\def\\osf@rm{osf-#{roman}}
269       \\def\\osf@sf{osf-#{sans}}
270
271       \\newcommand{\\osfigures}{%
272         \\renewcommand{\\rmdefault}{\\osf@rm}%
273         \\renewcommand{\\sfdefault}{\\osf@sf}%
274         \\ifx\\f@family \\tb@rm \\rmfamily\\fi
275         \\ifx\\f@family \\tb@sf \\sffamily\\fi}
276
277       \\newcommand{\\tbfigures}{%
278         \\renewcommand{\\rmdefault}{\\tb@rm}%
279         \\renewcommand{\\sfdefault}{\\tb@sf}%
280         \\ifx\\f@family \\osf@rm \\rmfamily\\fi
281         \\ifx\\f@family \\osf@sf \\sffamily\\fi}
282
283       \\DeclareOption{osfigures}{
284          \\osfigures
285          \\let\\@Fam\\osfigures
286       }
287       \\DeclareOption{tbfigures}{
288          \\tbfigures
289          \\let\\@Fam\\tbfigures
290       }
291       \\ExecuteOptions{osfigures}
292       \\ProcessOptions
293       \\AtBeginDocument{\\@Fam}
294    EOF
295 end
296
297 # list the fonts for which the DATA section contains ready input
298 # this is used if no fonts are given on the command line
299
300 predef = %w{pxfonts txfonts}
301 $textree = "#{ENV['HOME']}/texmf"
302
303 if ARGV.size > 0   # one or two arguments, say pxr (and pcxr):
304    font,dir = findfont(ARGV[0])
305    # pxr, .../vf/public/pxfonts
306    has_oldstyle_digits("#{dir}/#{font}.vf") and
307       die "#{font} has oldstyle digits already"
308
309    if ARGV[1]
310       osfont,osdir = findfont(ARGV[1])
311       # pcxr, .../vf/public/pxfonts
312       has_oldstyle_digits("#{osdir}/#{osfont}.vf") or
313          die "#{osfont} has no oldstyle digits"
314    else
315       # no oldstyle font given: propose one
316       osdir = dir
317       osf = Array.new
318       Dir["#{dir}/*.vf"].each { |fontfile|
319          f = File.basename(fontfile,'.vf')
320          next if f == font
321          osf.push(f) if has_oldstyle_digits(fontfile)
322       }
323       # osf array contains all oldstyle fonts found
324       case osf.size
325        when 0 then puts "I found no accompanying fonts with " +
326                         " old style digits in #{dir}"
327          when 1 then osfont = osf[0]
328          else
329             # more than 1 found: find best matching name:
330             nearest = distance = 1000
331           puts "I found #{osf.size} accompanying fonts with " +
332                  "old style digits:"
333             for i in 1..osf.size do
334                f = osf[i-1]
335                h = Hash.new
336                for j in f.split('') do
337                   h[j] = (h[j] || 0) + 1
338                end
339                for j in font.split('') do
340                   h[j] = (h[j] || 0) - 1
341                   h.delete(j) if h[j] == 0
342                end
343                printf("%2d %2d %s\n",i,h.sum_of_squares,f)
344                if h.sum_of_squares < distance
345                   nearest = i
346                   distance = h.sum_of_squares
347                end
348             end
349             puts "My guess is #{nearest} (#{osf[nearest-1]})"
350             print "Your guess [#{nearest}]: "
351             i = STDIN.gets.chomp
352             i = i == '' ? nearest : i.to_i
353             osfont = osf[i-1]
354       end
355    end
356    convert_font(nil,font,osfont)
357 else
```

```
    # no arguments: use DATA section
    puts "#{predef.size} font conversions have been predefined"
    (1..predef.size).each { |i|
        puts "#{i} #{predef[i-1]}"
    }
    choice = 0
    until choice > 0 && choice <= predef.size
       print "Please make your choice [1]: "
       choice = STDIN.gets.to_i
       choice = 1 if choice == 0
    end
    collection = predef[choice-1]
    puts "Converting #{collection}"
    latexdir = "#{$textree}/tex/latex/osf-#{collection}"
    File.mkpath(latexdir) or
       die "Could not create directory #{latexdir}"
    DATA.each { |line|
       break if line.chomp == collection
    }
   roman_sans = Array.new # will names of roman and sans families
                          # used for the style file
    DATA.each { |line|
       line.chomp!
       case line
          when '' then break
          when /^fd\s/ then
             dummy,fd,*pat = line.split
             # fd commands: roman must come first, sans second
           # first pattern must be the name of roman/sans family
             roman_sans.push(pat[0])
             fd =~ /^t1/ or
                die ".fd filename must start with 't1'"
             infd = `kpsewhich #{fd}.fd`.chomp
             die "Could not find #{fd}.fd" if fd == ''
           out = open("#{latexdir}/#{fd.sub(/^t1/,'t1osf-')}.fd",
                      'w')
             open(infd).each { |l|
                # put prefix before all patterns:
                for p in pat do
                   l.gsub!(/(#{p})/,'osf-\1')
                end
                out.print(l)
             }
             out.close
          else
             puts line
             font,osfont = line.split
             # if the .vf exists from a previous run delete it,
             # and its .tfm companion:
             if FileTest.exist?("#{font}.vf")
                File.delete("#{font}.vf")
                File.delete("#{font}.tfm")
             end
             convert_font(collection,font,osfont)
          end
    }
    out = open("#{latexdir}/osf-#{collection}.sty",'w')
    out.print style(collection,*roman_sans)
    system('mktexlsr')
end

__END__
pxfonts
fd         t1pxr pxr p1x
fd         t1pxss pxss t1x
p1xb       pcxb
p1xbi      pcxbi
p1xbsc     pcxb
p1xbsl     pcxbsl
p1xi       pcxi
p1xr       pcxr
p1xsc      pcxr
p1xsl      pcxsl
pxb        pcxb
pxbi       pcxbi
pxbsc      pcxb
pxbsl      pcxbsl
pxi        pcxi
pxr        pcxr
pxsc       pcxr
pxsl       pcxsl

txfonts
fd         t1txr txr t1x
fd         t1txss txss t1x
t1xb       tcxb
t1xbi      tcxbi
t1xbsc     tcxb
t1xbsl     tcxbsl
t1xbss     tcxbss
t1xbsssc   tcxbss
t1xbsssl   tcxbsssl
t1xi       tcxi
t1xr       tcxr
t1xsc      tcxr
t1xsl      tcxsl
t1xss      tcxss
t1xsssc    tcxss
t1xsssl    tcxsssl
txb        tcxb
txbi       tcxbi
txbsc      tcxb
txbsl      tcxbsl
txbss      tcxbss
txbsssc    tcxbss
txbsssl    tcxbsssl
txi        tcxi
txr        tcxr
txsc       tcxr
txsl       tcxsl
txss       tcxss
txsssc     tcxss
txsssl     tcxsssl
tyxb       tcxb
tyxbi      tcxbi
tyxbsc     tcxb
tyxbsl     tcxbsl
tyxbss     tcxbss
tyxbsssc   tcxbss
tyxbsssl   tcxbsssl
tyxi       tcxi
tyxr       tcxr
tyxsc      tcxr
tyxsl      tcxsl
tyxss      tcxss
tyxsssc    tcxss
tyxsssl    tcxsssl
```

Notes

1. The *osf* script can be downloaded from
`www.servalys.nl/tex/`
2. the T1 must stay in front, because LaTeX expects it there

Wybo Dekker
`wybo@servalys.nl`

Apple Symbols

Abstract
This Mac-specific My Way documents some fonts available exclusively on MacOSX 10.3, "Panther," and makes them available to Mac users with fairly minimal installation effort. I do not distribute the fonts themselves.

Introduction

This Mac-specific My Way documents some fonts available exclusively on MacOSX 10.3, "Panther," and makes them available to Mac users with fairly minimal installation effort. I do not distribute the fonts themselves.[1]
The fonts in question are `Apple Symbols.ttf` and `LucidaGrande.dfont`. The first is a generic font that makes available many of the symbolic/non-linguistic characters from the UNICODE character set, and the second is a large UNICODE font, optimised for screen viewing, with many additional Macintosh-specific font glyphs.

Installation

Installation is fairly simple, as long as you have a modicum of knowledge of the Terminal. Start by downloading (⌥-click in Safari) the package `symb-mac.tar.gz`,[2] and note where your download directory is. Go to the Terminal and change directory to wherever you keep your TEX local changes. That may be `~/Library/texmf` or `/usr/local/teTeX/texmf.local`. Once there, type:

```
tar xzvf /path/to/downloaded/symb-mac.tar
```

If that directory is not writeable, you will get some error messages, and will need to precede all of the commands in this section with `sudo`, and enter your password occasionally.

Assuming all goes well, the files should be installed on your machine, along with an alias to the `Apple Symbols.ttf` font, which PDFTEX should be able to read directly. `LucidaGrande` is in a format that PDFTEX cannot currently read, so you must convert the format manually.[3] For this, we need George Williams' `fondu`[4] utility, perhaps the Mac TEXer's best friend. From the directory where you just installed the `symb-mac` package, type:

```
cd fonts/truetype/apple/lucidagrande
fondu /System/Library/LucidaGrande.dfont
ls
```

At this point, you should see both `LucidaGrande.ttf` and `LucidaGrandeBold.ttf` listed in that folder. Finish the installation by letting TEX know where the new files are:

```
texhash
```

Usage

The fonts that are now installed can serve two purposes: there is a large set of UNICODE symbols enabled by the fonts, and there are Mac-specific symbols that may be useful for typesetting Macintosh documentation.

Unicode Symbols

The first usage of these fonts was to complement the `symb-uni.zip`[5] ConTeXt package, not only linking Unicode glyphs with ConTeXt symbolsets, but making the glyphs available so that Mac users can actually use these symbolsets in their documents.

To setup your document for the Unicode symbols, load the symbol file, the map file, and make the appropriate font synonyms:

```
\usesymbols [uni]
\loadmapfile[unicode-apple-applesymbols]
\definefontsynonym[UnicodeRegular20][applesymbols20xx-AppleSymbols]
\definefontsynonym[UnicodeRegular21][applesymbols21xx-AppleSymbols]
\definefontsynonym[UnicodeRegular24][applesymbols24xx-AppleSymbols]
\definefontsynonym[UnicodeRegular25][applesymbols25xx-AppleSymbols]
\definefontsynonym[UnicodeRegular26][applesymbols26xx-AppleSymbols]
\definefontsynonym[UnicodeRegular27][lucidasans27xx-LucidaSansRegular]
```

To actually place these symbols in your documents, follow the instructions in `UnicodeSymbols.pdf`.[6]

Mac-Specific Symbols

Once the fonts were installed for Unicode purposes, I started examining the glyphs that were left over. There were a myriad of interesting glyphs related to the Macintosh keyboard, historical Apple hardware, multimedia, and other miscellany. However, there are few ways of automating the creation of encodings (groups of 256 named glyphs) or of symbol sets, so these had to be created largely by hand.

To use the Mac symbols, load the `symb-mac` file with:

```
\usesymbols[mac]
```

By loading the symbols this way, the map files are loaded and the appropriate font synonyms are created in the header of that file, so no other commands are needed. However, as the map files are loaded at the time `\usesymbols` is called, be sure to call this before the first page is shipped out, ideally in the document's header.

From there, you can call the symbols like any other:

```
Some headphones: \symbol[Apple Audio][Headphones]
```

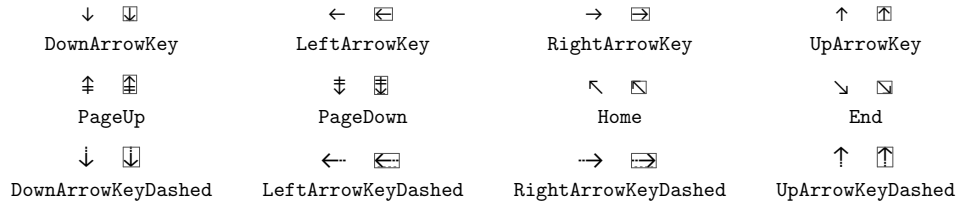
Some headphones: 🎧

Help wanted

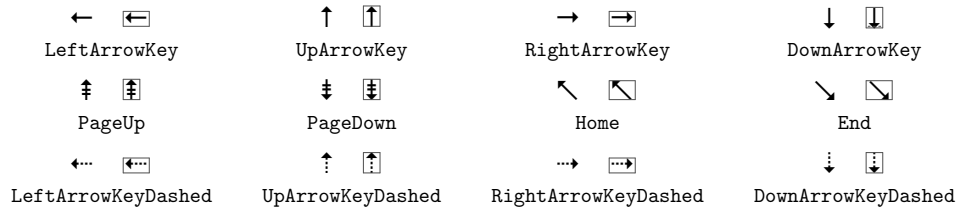
Naming things is hard, and this axiom applies to these signs and symbols. I've tried my hardest to name the symbols accurately, but it's fairly labour intensive and there are fairly obscure symbols. So, if there are enthusiastic Mac users willing to fill in the gaps with these glyphs, please, contributions are appreciated. Send contributions to the ConTeXt mailing list or to `atl@comp.lancs.ac.uk`.

Apple Symbol Examples

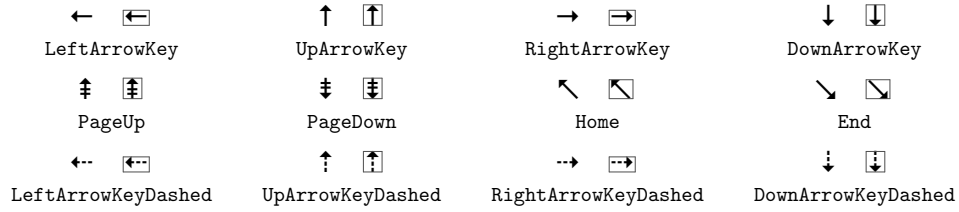
Apple Cursor Keys



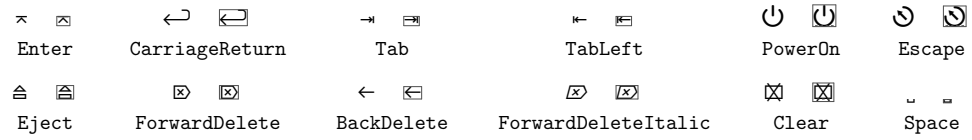
Apple Cursor Keys 2



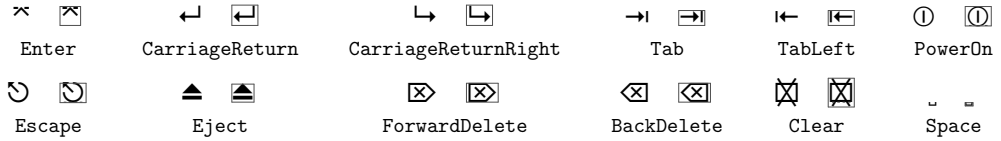
Apple Cursor Keys Bold



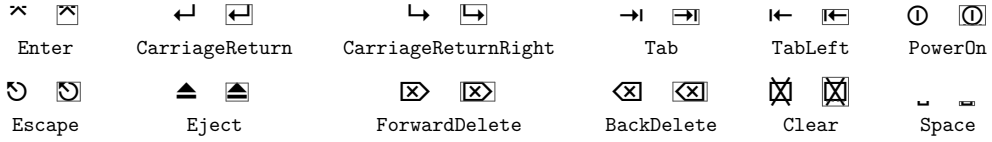
Apple Control Keys



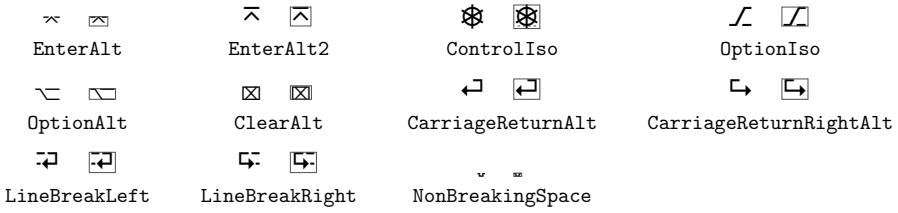
Apple Control Keys 2



Apple Control Keys Bold

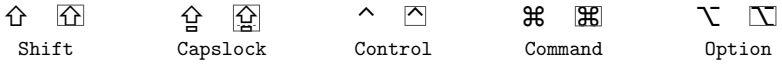


Apple Alt Control Keys



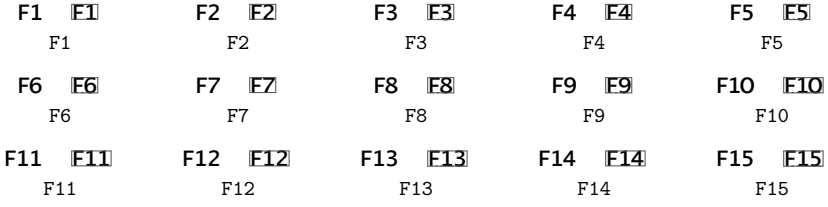
Apple Modifier Keys

Apple Modifier Keys 2

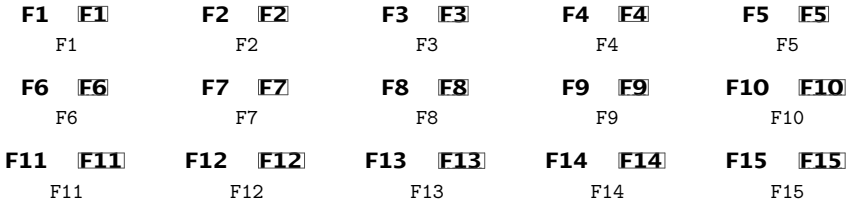


Apple Modifier Keys Bold

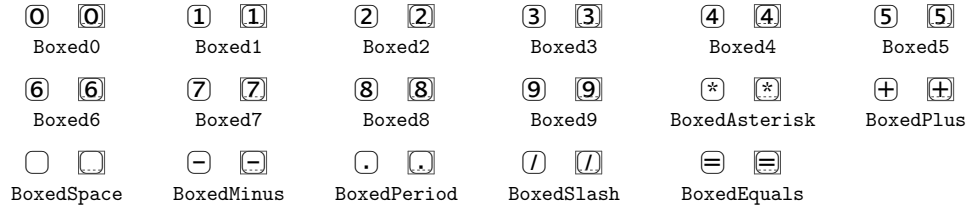
Apple Function Keys



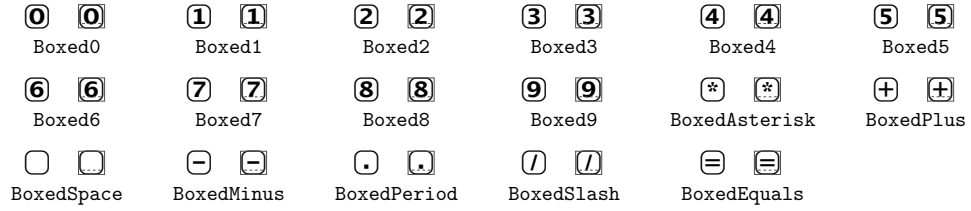
Apple Function Keys Bold



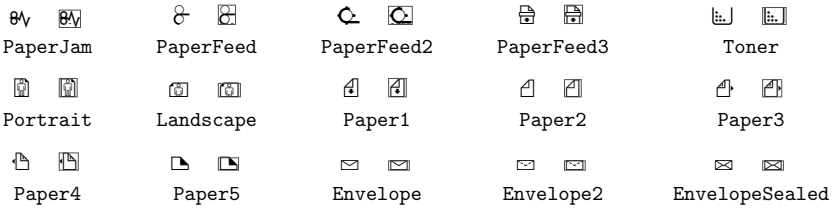
Apple Number Keys



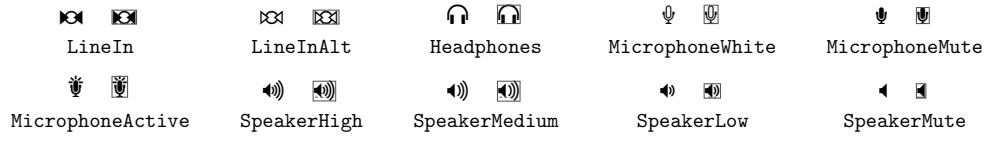
Apple Number Keys Bold



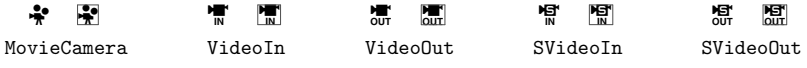
Apple Printing



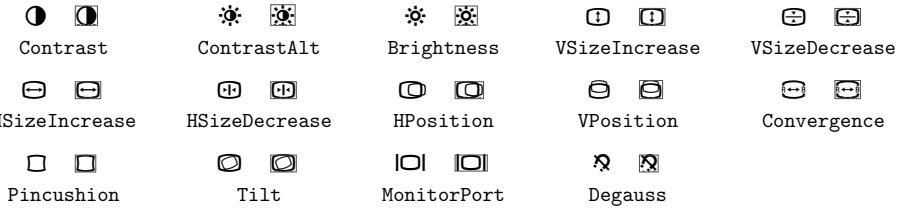
Apple Audio



Apple Video

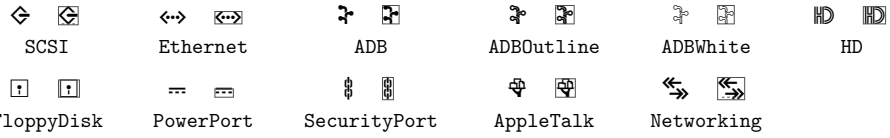


Apple Monitors

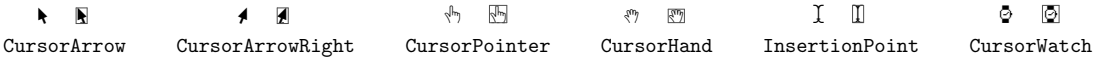


Apple Communications

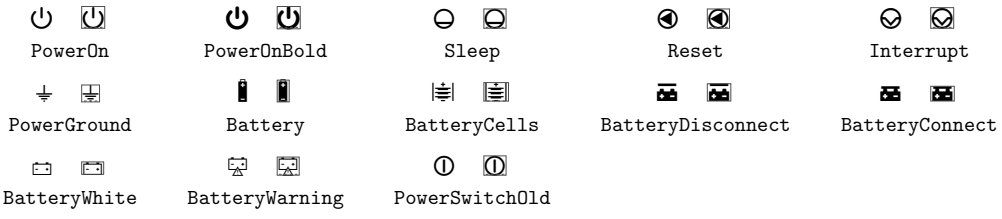
Apple Peripherals



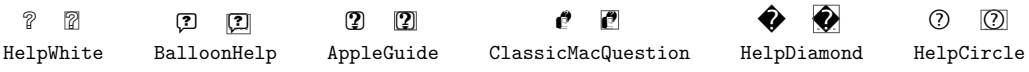
Apple Cursors



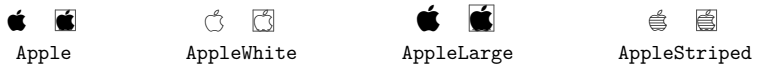
Apple Power and Battery



Apple Help



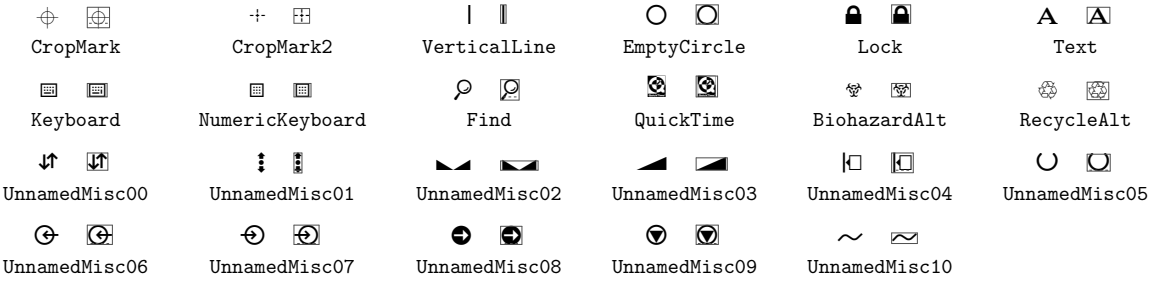
Apple Logo



Apple Newton

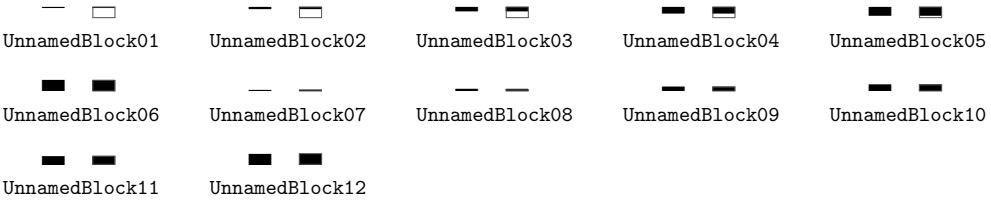
Apple Warning

Apple Miscellaneous



Apple Miscellaneous 2

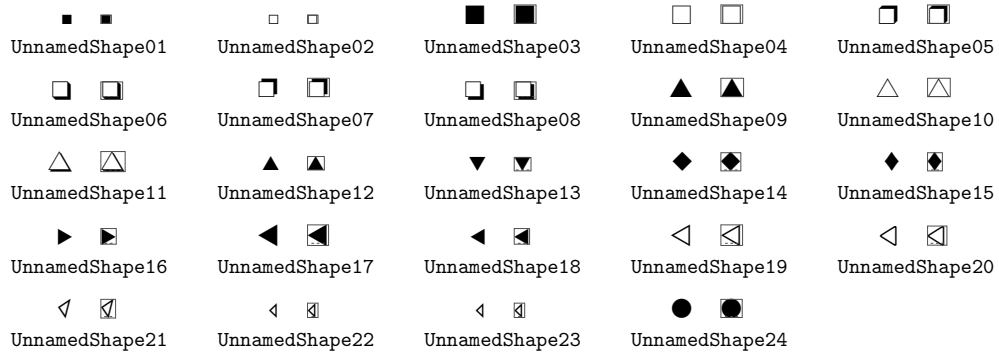
Apple Block Elements



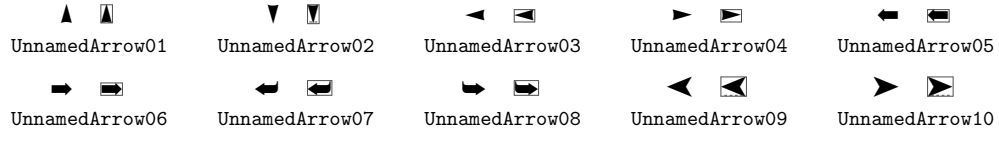
Apple Geometric Shapes

Apple Geometric Shapes Bold

Apple Geometric Shapes 2



Apple Arrows



Unicode Chart Forms

Notes

1. Nor will I respond to requests for them. Sorry, but I do need to respect Apple's intellectual property.
2. from http://homepage.mac.com/atl/tex
3. I leave it up to your conscience whether this local font conversion is legal and/or ethical.
4. downloadable from http://fondu.sourceforge.net/
5. from http://homepage.mac.com/atl/tex
6. from http://homepage.mac.com/atl/tex

Adam T. Lindsay
Lancaster University

Unicode Symbols

Abstract
The *Unicode* standard includes a number of signs, symbols, dingbats, bullets, arrows, graphical elements, and other miscellaneous glyphs. Prompted by finding a font dedicated to many such *Unicode* symbols on MacOSX systems, this magazine documents some ways of enabling these symbols on your own system.

Introduction

The UNICODE standard is dedicated to creating a universal character set for all of the languages on earth. Signs and symbols are often important components of and aids to printed communication. Appropriately enough, UNICODE dedicates a number of blocks to symbols, arrows, block elements, and geometric shapes that can be useful in some documents.

CONTEXT offers integrated support for symbols. As such, all that's necessary for CONTEXT support for UNICODE symbols is a font that supports those symbols, an encoding that reaches those glyphs, and a little bit of CONTEXT code to organise those symbols into symbol sets. As there are hundreds of these symbols, it's quite fortunate that this process is scriptable.

OpenType fonts formalise support for UNICODE, whether they be in TrueType (`ttf`) or PostScript (`otf`) glyph format. As such, this article can be seen as an extension of OpenType support in the script dimension.[1]

Getting the right encoding

TEX, rather infamously, is still saddled with an 8-bit limit when dealing with fonts. So a TEX font can only contain 256 glyphs. Supporting UNICODE fonts thus means subdividing a large font with over a thousand symbols into 256-glyph chunks. The particular glyphs in a chunk are identified by their postscript names, and this collection of 256 glyph names constitute an encoding, designated with an `.enc` file suffix.

Getting an encoding right is a bit of an art. The mapping from UNICODE name to postscript name is different with each font: there is no standard postscript name for most UNICODE entities. The approach for getting an encoding depends on how the font encodes its constituent glyphs. A simple indicator of what can be found within a font is by looking at the `afm` file. The glyph names are typically mostly named or mostly numbered.

Sequential Encoding

The first way that fonts can identify their glyph names is sequentially, by index. It's especially helpful if a font identifies glyphs in UNICODE order. Many of Adobe's OpenType fonts do this, with glyphs accessible from names like uni0041 and uni222A, the hexadecimal numbers referring directly to the unicode glyphs at the corresponding number.

An encoding can be synthesised directly with a tiny PERL script, `unienc.pl`:

```
#!/usr/bin/perl

print "/Unicode_$ARGV[0]_Encoding[ \n";

for ($n=0;$n<256;$n++)
  { printf ("/uni$ARGV[0]\U%02x\n",$n) }
```

```
print "] def\n";
```

There are other fonts that enumerate the glyphs in font order, rather than UNICODE or any other order, and, worse, don't give any meaningful information in the glyph names. The `Apple Symbols.ttf` font was like this, with glyphs labelled as gid65 and gid1146. A simple modification of the above PERL script will gladly spit out 256 sequential gid–prefixed glyph names. This is useful for accessing non-UNICODE glyphs in a font.

Named Encoding

If a font's glyphs are mostly named, then one can laboriously assemble an encoding by hand. It would be more useful if that process can be scripted. In order to do so, some mapping from UNICODE number to glyph name must be obtained. Non-standard font manipulation tools must be used for that.

Apple provides one such tool in their FTXTOOLS suite.[2] It can dump and manipulate fonts using XML as a data format. An XML dump of a font's `cmap` table is just what's needed for inspecting the character mapping. The command is:

```
ftxdumperfuser -A d -t cmap -u -p -n fontfile.ttf
```

Of more general use are the TTX FontTools, from Just van Rossum/LettError.[3] It also dumps and manipulates fonts using XML as an interchange format. In order to get a minimally useful `.ttx` file, a command would be:

```
ttx -t cmap -t name fontfile.ttf
```

This yields an XML file like:

```
<ttFont sfntVersion="\x00\x01\x00\x00" ttLibVersion="2.0b1">
    <cmap>
      <tableVersion version="0"/>
      <cmap_format_4 platformID="3" platEncID="1" version="0">
        <map code="0x2600" name="gid289"/>
        <map code="0x2601" name="gid290"/>
        <map code="0x2602" name="gid291"/>
        <map code="0x2603" name="gid292"/>
        <map code="0x2604" name="gid293"/>
      </cmap_format_4>
    </cmap>
    <name>
      <namerecord nameID="1" platformID="1" platEncID="0" langID="0x0">
        Apple Symbols
      </namerecord>
    </name>
  </ttFont>
```

It's pretty clear from inspection how the file relates UNICODE numbers (`codes`) with POSTSCRIPT glyph names (`names`). Not every font makes all of the necessary tables visible, so other strategies need to be used in those cases. If a complete `.ttx` file is available, however, then you can use `ttx2enc.xsl`, a stylesheet that transforms a TTX file into an enc file for use with TEXFONT and PDFTEX:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version=
"1.0" xmlns:date="http://exslt.org/dates-and-times">
    <xsl:output method="text"/>
    <xsl:strip-space elements="*"/>
    <xsl:param name="vector">
    </xsl:param>
    <xsl:variable name="hexdigits" select="'0123456789abcdef'"/>
    <xsl:template name="grab-glyph-name">
      <xsl:param name="char-value"/>
      <xsl:choose>
```

```
    <xsl:when test="map[@code = $char-value]">
      <xsl:text>/  </xsl:text>
      <xsl:value-of select="map[@code = $char-value]/@name"/>
      <xsl:text> %   </xsl:text>
      <xsl:value-of select="$char-value"/>
      <xsl:text>   </xsl:text>
      <xsl:value-of select="map[@code = $char-value]/following-sibling::comment()[1]"/>
      <xsl:text>   </xsl:text>
    </xsl:when>
    <xsl:otherwise>
      <xsl:text>/.notdef %   </xsl:text>
      <xsl:value-of select="$char-value"/>
      <xsl:text>   </xsl:text>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
<xsl:template match="/">
  <xsl:text>% Automatically generated encoding from ttx2enc.xsl % ttx2enc.xsl
by Adam T. Lindsay, 2004-01-23 % generated on   </xsl:text>
  <xsl:value-of select="date:date-time()"/>
  <xsl:text> % for the font:    </xsl:text>
  <xsl:value-of select="normalize-space(/ttFont/name/namerecord[@nameID='1'][1])"/>
  <xsl:text>   </xsl:text>
  <xsl:value-of select="normalize-space(/ttFont/name/namerecord[@nameID='2'][1])"/>
  <xsl:text> % for the vector:    </xsl:text>
  <xsl:value-of select="concat('0',$vector)"/>
  <xsl:text> /Unicode  </xsl:text>
  <xsl:value-of select="concat('0',$vector)"/>
  <xsl:text>Encoding [   </xsl:text>
  <xsl:apply-templates mode="cmap2glyph"/>
  <xsl:text>] def   </xsl:text>
</xsl:template>
<xsl:template match="/ttFont/cmap/cmap_format_4[1]" mode = "cmap2glyph">
  <xsl:call-template name="iterator"/>
</xsl:template>
<xsl:template match="*"/>
<xsl:template match="text()" mode="cmap2glyph"/>
<xsl:template name="iterator">
  <xsl:param name="outervalue">
    0
  </xsl:param>
  <xsl:param name="innervalue">
    0
  </xsl:param>
  <xsl:call-template name="grab-glyph-name">
    <xsl:with-param name="char-value">
    <xsl:value-of select = "concat('0x', $vector, $outervalue, $innervalue)"/>
    </xsl:with-param>
  </xsl:call-template>
  <xsl:choose>
    <xsl:when test="$outervalue='f' and $innervalue='f'">
    </xsl:when>
    <xsl:when test="$innervalue='f' and $outervalue!='f'">
      <xsl:call-template name="iterator">
        <xsl:with-param name="outervalue">
        <xsl:value-of select = "substring($hexdigits, string-length(
      substring-before( $hexdigits, $outervalue))+2, 1)"/>
        </xsl:with-param>   <xsl:with-param name="innervalue">0  </xsl:with-param>
      </xsl:call-template>
    </xsl:when>
```

```
        <xsl:otherwise>
          <xsl:call-template name="iterator">
            <xsl:with-param name="outervalue">
            <xsl:value-of select = "$outervalue"/>
            </xsl:with-param>    <xsl:with-param name="innervalue">
            <xsl:value-of select = "substring( $hexdigits, string-length(
          substring-before( $hexdigits, $innervalue))+2, 1)"/>
            </xsl:with-param>
          </xsl:call-template>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:template>
  </xsl:stylesheet>
```

(For MacOSX users, the included `cmap2enc.xsl` transform will accomplish the same thing, but with a `.cmap.xml` file input. This one was written first, in fact.)

The UNICODE `enc` files for one font were generated with different values for the `vector` parameter. The vector identifies the high byte of the UNICODE value, and therefore the group of 256 glyphs. Which vectors are interesting can be discovered through inspection of the font and by looking at the UNICODE file `Blocks.txt`.[4] For example, the "Miscellaneous Symbols" block is located in the range 2600 to 26FF, corresponding with vector 26.

Michael Kay's SAXON[5] was the XSLT processor of choice, but you can use the command of your choice:

```
saxon AppleSymbols.ttx ttx2enc.xsl vector=1e > applesymbols1exx.enc
```

Note, above, that the both the `vector` parameter and the name of the output `enc` file should be all lowercase. This yields a file like:

```
/Unicode26Encoding [
/gid289 % 0x2600 BLACK SUN WITH RAYS
/gid290 % 0x2601 CLOUD
/gid291 % 0x2602 UMBRELLA
/gid292 % 0x2603 SNOWMAN
/gid293 % 0x2604 COMET
        % ... 250 more glyphs ...
/.notdef % 0x26FF
] def
```

Perl from Unicode

The methods described above are only a couple possibilities. Another one is to use the `UnicodeData.txt`[6] file with PERL, RUBY, or another text processor of your choice.

Font installation

Be sure to install any newly generated encodings into a place where `kpsewhich` can find them, like `texmf-fonts/dvips/local/`.

Once you have the encoding files for the vectors of interest in place, you can run TEXFONT with the appropriate encoding, like:

```
texfont --make --install --ve=foo --co=bar --en=bar26xx
```

Support file

CONTEXT has very nice support for named symbols which can be loaded by named sets. It makes sense to use this mechanism. As these symbols are UNICODE entities, we may as well call UNICODE `\uchar` commands directly. As we have hundreds and

hundreds of glyphs to deal with, it makes sense to script the assignment of names, as well.

A modification of the Apple XSLT stylesheet (included in the distribution as `cmap2symb.xsl`) will take a vector and convert the all-caps unicode names in a `.cmap.xml` file into inter-capped symbol names within a symbol definition:

This results in ouput lines like the following:

```
\definesymbol[BlackSunWithRays][\uchar{38}{0}] % BLACK SUN WITH RAYS
\definesymbol[Cloud][\uchar{38}{1}] % CLOUD
\definesymbol[Umbrella][\uchar{38}{2}] % UMBRELLA
\definesymbol[Snowman][\uchar{38}{3}] % SNOWMAN
```

A little bit of manual effort to group the symbols into symbol sets followed, along with some shortening and correction of names. In general, the sets are named after the corresponding UNICODE block- or sub-block. The sets defined so far are as follows:

Symbol Set Name	**Block**	**Required Font**
Unicode Additional Punctuation	0x2000	UnicodeRegular20
Unicode Currency	0x20A0	UnicodeRegular20
Unicode Letterlike	0x2100	UnicodeRegular21
Unicode Letterlike Additional	0x2100	UnicodeRegular21
Unicode Script Letterlike	0x2100	UnicodeRegular21
Unicode Hebrew Letterlike	0x2100	UnicodeRegular21
Unicode Turned Letterlike	0x2100	UnicodeRegular21
Unicode Black-letter Letterlike	0x2100	UnicodeRegular21
Unicode Double-struck Letterlike Math	0x2100	UnicodeRegular21
Unicode Roman Numerals	0x2150	UnicodeRegular21
Unicode Small Roman Numerals	0x2150	UnicodeRegular21
Unicode Arrows	0x2190	UnicodeRegular21
Unicode Multi Arrows	0x2190	UnicodeRegular21
Unicode Optical Character Recognition	0x2440	UnicodeRegular24
Unicode Circled Digits	0x2460	UnicodeRegular24
Unicode Box Drawing	0x2500	UnicodeRegular25
Unicode Double Box Drawing	0x2500	UnicodeRegular25
Unicode Block Elements	0x2580	UnicodeRegular25
Unicode Shade Characters	0x2580	UnicodeRegular25
Unicode Terminal Graphics	0x2580	UnicodeRegular25
Unicode Geometric Shapes	0x25A0	UnicodeRegular25
Unicode Control Code Graphics	0x25A0	UnicodeRegular25
Unicode Weather and Astrological	0x2600	UnicodeRegular26
Unicode Miscellaneous	0x2600	UnicodeRegular26
Unicode Japanese Chess	0x2600	UnicodeRegular26
Unicode Pointing Hand	0x2600	UnicodeRegular26
Unicode Warning Signs	0x2600	UnicodeRegular26
Unicode Healing Signs	0x2600	UnicodeRegular26
Unicode Religious and Political	0x2600	UnicodeRegular26 & 27
Unicode Trigram	0x2600	UnicodeRegular26
Unicode Zodiac	0x2600	UnicodeRegular26
Unicode Chess	0x2600	UnicodeRegular26
Unicode Playing Card	0x2600	UnicodeRegular26
Unicode Musical	0x2600	UnicodeRegular26
Unicode Recycling	0x2600	UnicodeRegular26
Unicode Dice	0x2600	UnicodeRegular26
Unicode Go Markers	0x2600	UnicodeRegular26
Unicode Dingbats	0x2700	UnicodeRegular27

Unicode Checks and Xs	0x2700	UnicodeRegular27
Unicode Stars	0x2700	UnicodeRegular27
Unicode Snowflakes	0x2700	UnicodeRegular27
Unicode Shadowed Shapes	0x2700	UnicodeRegular27
Unicode Bars	0x2700	UnicodeRegular27
Unicode Dingbat Punctuation	0x2700	UnicodeRegular27
Unicode Hearts	0x2700	UnicodeRegular27
Unicode Negative Circled Digits	0x2700	UnicodeRegular27
Unicode Circled Sans-serif Digits	0x2700	UnicodeRegular27
Unicode Negative Circled Sans-serif Digits	0x2700	UnicodeRegular27
Unicode Dingbat Arrows	0x2700	UnicodeRegular27
Unicode Shadowed Arrows	0x2700	UnicodeRegular27
Unicode Tailed Arrows	0x2700	UnicodeRegular27

Usage

In order use these pre-defined symbols, load the definitions from the somewhat presumptuously named `symb-uni.tex`:

```
\usesymbols[uni]
```

As the symbol definitions depend on unicode fonts being defined, you need to load map files and define simple font synonyms:

```
\loadmapfile [applesymbols25xx-apple-applesymbols.map]
\loadmapfile [applesymbols26xx-apple-applesymbols.map]
\definefontsynonym [UnicodeRegular25] [applesymbols25xx-AppleSymbols]
\definefontsynonym [UnicodeRegular26] [applesymbols26xx-AppleSymbols]
```

Using the symbols is like others in ConTeXt.

```
\showsymbolset[Unicode Warning Signs][n=4]
```

SkullAndCrossbones CautionSign RadioactiveSign BiohazardSign

```
\setupsymbolset[Unicode Healing Signs]
Here's an Ankh: \symbol [Ankh] \quad and a Caduceus: \symbol [Caduceus]
```

Here's an Ankh: ☥ and a Caduceus: ⚚

```
Here's a hammer and sickle, called without loading the symbol set:
\symbol [Unicode Religious and Political] [HammerAndSickle]
```

Here's a hammer and sickle, called without loading the symbol set: ☭

```
\definesymbol[1][{\symbol[Unicode Miscellaneous][BallotBoxWithCheck]}]
\definesymbol[2][{\symbol[Unicode Pointing Hand][WhiteRightPointingIndex]}]
\startitemize[packed]
\item You can hook symbols into the itemize mechanism at different levels.
\startitemize[packed]
  \item This is not new, but you have many more typographic options
  available to you.
  \item Be sure to use typographic restraint and good taste!
\stopitemize
\stopitemize
```

☑ You can hook symbols into the itemize mechanism at different levels.

☞ This is not new, but you have many more typographic options available to you.
☞ Be sure to use typographic restraint and good taste!

Notes

1. see other articles at http://homepage.mac.com/atl/tex/.
2. available at http://developer.apple.com/fonts/OSXTools.html
3. available at http://sourceforge.net/projects/fonttools/
4. from http://www.unicode.org/Public/UNIDATA/
5. from http://saxon.sourceforge.net/
6. from http://www.unicode.org/Public/UNIDATA/

Adam T. Lindsay
Lancaster University

Woordafbreking op ë en ï

Abstract
LaTeX heeft moeite met het afbreken van woorden die een ë bevatten. Dit verhaal laat zien hoe dat probleem op te lossen: gebruik \"e of \"{e} of ë voor een eindstandige ë en "e voor alle andere. Analoog voor de ï.

Keywords
hyphenation diaresis umlaut

Wie in het Nederlands een tekst schrijft waar veel landennamen in voorkomen, heeft een gerede kans om problemen te krijgen met namen die op ë eindigen. De gebruikelijke manier om dit karakter in te voeren is door \"e te schrijven of, om het helemaal netjes te doen: \"{e}. Echter: woordafbreking vindt voor zo'n ë niet plaats en in de volgende voorbeeldtekst is het gevolg veel te veel wit in de tweede zin:

```
Itali\"e, ge\"eindigd
nooit afbreking vóór \"e
```

We gaan naar Indië en niet naar België of Italië want we houden van de tropen.
Maar…uiteindelijk zijn we in Zuid-Frankrijk geëindigd.

Bij gebruik van het BABEL package kan men in het Nederlands de ë ook invoeren met als "e en zo'n ë gedraagt zich anders (we zullen straks zien waarom): woorden kunnen ervoor afgebroken worden, zelfs, helaas, als die ë de laatste lettergreep van het woord is; het gevolg is een losse e aan het begin van de volgende zin:

```
Itali"e, ge"eindigd
```
afbreking vóór "e zelfs als het de laatste lettergreep is

We gaan naar Indië en niet naar België of Italie want we houden van de tropen.
Maar…uiteindelijk zijn we in Zuid-Frankrijk geeindigd.

Een andere mogelijkheid zou kunnen zijn, een kanten-klare ë uit een latin1-font te gebruiken, maar die gedraagt zich net als \"e: woorden worden daarvoor nooit afgebroken:

```
Italië, geëindigd
```
nooit afbreking vóór ë

We gaan naar Indië en niet naar België of Italië want we houden van de tropen.
Maar…uiteindelijk zijn we in Zuid-Frankrijk geëindigd.

De conclusie is daarom: gebruik \"e voor een eindstandige en "e voor alle andere ë's:

```
Itali\"e, ge"eindigd
```
de oplossing:

We gaan naar Indië en niet naar België of Italië want we houden van de tropen.
Maar…uiteindelijk zijn we in Zuid-Frankrijk geeindigd.

De discussie betreffende dit probleem op de tex-nl mailing list vermocht Johannes Braams, de auteur van BABEL, te verleiden tot het volgende commentaar:

Inderdaad "e is gebaseerd op de dubbel-quote als actief teken. Dit stamt uit het 7-bit tijdperk toen ë nog niet mogelijk was als invoer. Bovendien zorgde \"e ervoor dat een woord in het geheel niet werd afgebroken vanwege het invoegen van het \accent primitief. De constructie "e doet dus een aantal dingen:

- er wordt een \hskip van 0pt ingevoegd waardoor het woord voor TeX in twee woorden uiteenvalt. Daarbij wordt ervoor gezorgd dat als de regel tussen die twee woorden wordt afgebroken toch een afbreekteken verschijnt.
- er verschijnt een trema op de e.
- die trema is trouwens wel degelijk lager gezet dan wat het accent primitief normaal doet.
- Tot het repertoire behoort overigens ook "i (geeft ï, vroeger als \"{\i} in te voeren).

Dat het in dit geval met het afbreken misgaat omdat die ë in z'n eentje een woord gaat vormen (zo ziet TeX het althans) is ietwat onfortuinlijk. Italië breekt overigens alleen goed af als je ook de 8-bits afbreekpatronen van Piet Tutelaers gebruikt...

Wybo Dekker
wybo@servalys.nl

LaTeX uitvoer genereren vanuit C programma's

Abstract
This article describes a simple way to generate LaTeX output from C programs.

Keywords
LaTeX, automatisch opmaken, C

Inleiding

Onlangs heeft de auteur een C-programma geschreven om mechanische en thermische eigenschappen van laminaten gemaakt van vezelversterkte kunststoffen te berekenen[1]. De uitvoer van dit programma bestaat uit een tweetal tabellen—die een overzicht van de opbouw van een laminaat en de globale eigenschappen weergeven—en een tweetal 6×6 matrices.

Om de resultaten van dit programma makkelijk in documenten te kunnen gebruiken, is het voorzien van de mogelijkheid om naast platte tekst ook HTML en LaTeX uitvoer te genereren. Deze laatste mogelijkheid is onderwerp van dit artikel.

Hoewel hier gebruik is gemaakt van de taal C, kan dezelfde aanpak ook zonder problemen gebruikt worden in andere programmeertalen.

Werkwijze

Uit oogpunt van flexibiliteit, is ervoor gekozen om géén compleet LaTeX document laten genereren, maar alleen datgene wat nodig is—bijvoorbeeld samengebonden in een `table`-omgeving. Aangezien het met behulp van `\input` heel makkelijk is om de gegenereerde code op te nemen in een document, leek mij dit de meest flexibele oplossing. Het geeft de gebruiker namelijk maximale vrijheid in het kiezen van documentstijlen en opties.

Bij uitvoeren van LaTeX-code vanuit C—en andere programmeertalen—moet rekening gehouden worden met een aantal valkuilen, waarover later meer. Daarom is het raadzaam om de volgende werkwijze aan te houden:

- □ maak een prototype van de gewenste uitvoer in LaTeX.
- □ vertaal dit naar uitvoerstatements in C broncode
- □ verfijn het resultaat

Op deze manier kun je het probleem opsplitsen in onafhankelijke delen, waarvan sommige geautomatiseerd kunnen worden.

Valkuilen

In *strings* in C[2] wordt de '\' gebruikt voor het aangeven van een aantal speciale karakters, met name aanhalingstekens en regeleindes, zie [KR90, blz. 51]. Om een letterlijke '\' in de uitvoer te krijgen, moet '\\' in de C tekenreeks gebruikt worden!

Voor het afdrukken van tekst word gebruik gemaakt van verschillende functies uit de C standaardbibliotheek, `puts` [KR90, blz. 336] en `printf` [KR90, blz. 331]. Met printf kun je ook gegevens uit het programma afdrukken, terwijl puts alleen maar letterlijke strings afdrukt. Ook voegt puts automatisch aan het eind van de regel een regeleinde (*newline*, '\n') toe, terwijl je die bij printf expliciet moet toevoegen. Als printf een '%' in de tekenreeks ziet, interpreteert hij dit als een *conversiespecificatie*—een plaats waar een variabele uit het programma afgedrukt moet worden. Om een letterlijke '%' in de uitvoer te krijgen, moet '%%' in deze tekenreeks staan. Dit alles is samengevat in tabel 1.

Omzetting	Opmerkingen
\% → %	
\ → \\	
" → \"	
' → \'	
% → %%	in `printf`

Tabel 1. omzettingen van LaTeX naar C-*strings*

Omzetten

Hoewel het natuurlijk mogelijk is om een dergelijke omzetting met de hand te doen, is het makkelijker en minder foutgevoelig om het aan de computer over

te laten. Het PERL-programma [SC97] weergegeven in figuur 1 is in staat om de conversie uit te voeren. Het echte werk wordt gedaan door de vervangingsopdrachten met behulp van *reguliere expressies* [SC97, hoofdstuk 7] zoals 's/\\%/%/g;'. Deze opdracht betekent: zoek alle tekenreeksen '\%' en vervang ze door '%'. Door een aantal van deze vervangingsopdrachten in de juiste volgorde te gebruiken, kan de LaTeX-code moeiteloos omgezet worden in C-statements die dezelfde uitvoer genereren.

```
#!/usr/bin/perl -pa
chomp;
s/\\%/%/g;
s/\\/\\\\/g;
s/\"/\\\"/g;
s/\'/\\\'/g;
if (/%[^%\s]/) {        # use printf
    s/%([\\\s])/%%\1/g;
    printf("printf(\"%s\\n\", );\n", $_);
} else {            # use puts
    printf("puts(\"%s\");\n", $_);
}
```

Figuur 1. conversie script

Neem als voorbeeld de LaTeX-code weergegeven in figuur 2.

```
\begin{table}[htbp]
  \centering
  \begin{tabular}{lll}
    \toprule % gebruik het booktabs package
    Symbool & Waarde\\
    \midrule
    $E_1$ & \%g MPa\\
    $E_2$ & \%g MPa\\
    $E^f_1$ & \%g MPa\\
    $E^f_2$ & \%g MPa\\
    $E^m$ & \%g MPa\\
    $\nu_{12}$ & \%g -\\
    $\nu^f_{12}$ & \%g -\\
    $\nu_m$ & \%g -\\
    $V^f$ & \%g \%\\
    $\eta$ & 0,5\\
    \bottomrule
  \end{tabular}
  \caption{\label{symbcomp}mechanische
    eigenschappen UD laminaten}
\end{table}
```

Figuur 2. voorbeeld invoer

Na omzetting door het in figuur 1 weergegeven script, ziet dit er uit als in figuur 3.

Hiermee is de omzetting natuurlijk nog niet compleet. De gegenereerde code moet nog opgenomen worden in een C-functie, en in de printf functies moeten nog de juiste variabelen toegevoegd worden achter de comma's. Maar het meest foutgevoelige gedeelte van het werk is al gedaan.

```
puts("\\begin{table}[htbp]");
puts("  \\centering");
puts("  \\begin{tabular}{lll}");
puts("    \\toprule % gebruik het booktabs package");
puts("    Symbool & Waarde\\\\");
puts("    \\midrule");
printf("    $E_1$ & %g MPa\\\\\n", );
printf("    $E_2$ & %g MPa\\\\\n", );
printf("    $E^f_1$ & %g MPa\\\\\n", );
printf("    $E^f_2$ & %g MPa\\\\\n", );
printf("    $E^m$ & %g MPa\\\\\n", );
printf("    $\\nu_{12}$ & %g -\\\\\n", );
printf("    $\\nu^f_{12}$ & %g -\\\\\n", );
printf("    $\\nu_m$ & %g -\\\\\n", );
printf("    $V^f$ & %g %%\\\\\n", );
puts("    $\\eta$ & 0,5\\\\");
puts("    \\bottomrule");
puts("  \\end{tabular}");
puts("  \\caption{\\label{symbcomp}mechanische");
puts("    eigenschappen UD laminaten}");
puts("\\end{table}");
```

Figuur 3. voorbeeld uitvoer

Conclusies

Toen de gewenste lay-out van de uitvoer eenmaal vast stond, was het relatief makkelijk om het produceren van de gewenste LaTeX-code op te nemen in het programma. De kwaliteit van de uiteindelijke uitvoer weegt zeer zeker op tegen de tijd die het kost om het op te zetten.

Het feit dat LaTeX-commando's uit platte tekst bestaan—en uitstekend gedocumenteerd zijn—maakt het genereren hiervan kinderspel. Samen met het `\input` mechanisme geeft het een ongekend gemak en flexibiliteit in het opnemen van programmaresultaten in documenten.

Referenties

[KR90] Brian Kernighan and Dennis M. Ritchie. *C handboek*. Academic Service, 1990. ISBN 90-6233-488-1.

[SC97] Randal L. Schwartz and Tom Christiansen. *Learning Perl*. O'Reilly & Associates, 1997. ISBN 1-56592-284-0.

Notes

1. `lamprop`, te vinden op `http://www.xs4all.nl/~rsmith/software/`
2. en vele andere programmeertalen.

Roland Smith
`rsmith@xs4all.nl`

Help!
The Typesetting Area

Abstract
Typesetting (large) documents presents significant challenges that have to be resolved before a satisfactory printed result is achieved; e.g. the internal structure of the document should be clear, and the document's typographical layout should match its content. This article, based on a presentation given at the NTG day in Arnhem on 13 November 2003, describes a traditional design technique known as the harmonic proportion.

Introduction
Designing a layout for a document, and specially for a book, is not an easy task. Many discussions have been dedicated to the 'best' typographical approach. There are different schools, and therefore different opinions, concerning correct page design, e.g. Ph. Taylor [5] and J. Tschichold [6]. In this article I explain what one might take into account to achieve an appealing, harmonic result based on the work of Jan Tschichold [6].

Papersizes
In earlier times the most common ratio of height to width of paper from the factory was 3 : 4. A sheet of paper which is folded once is called folio. Adding another fold will result in a quarto which turns into an octavo after the next fold. When starting with a 3 : 4 sheet the proportions in the folded format will become consecutively: 2 : 3 and 3 : 4 again. The octavo, which is a section with 16 pages, would have the proportion 2 : 3.

Nowadays in Europe the DIN (Deutsche Industrie Norm) formats are used. The characteristic of these formats is that the proportion between height and width is $1:\sqrt{2}$ – which is approximately 1 : 1.414. When folding such paper sheets the ratio always remains $1:\sqrt{2}$. The DIN formats begin with the base size of A_0 which has a surface of 1 m^2. The index number rises with each time the sheet is cut in half – see figure 1.

Grain of Paper
When designing a document, it is important to know the grain of the paper. Handmade paper has no grain because its fibers settle down in random directions. So it makes no difference which direction the paper is folded. Not so for paper produced in a continuous process. Due to the (fast) movement in the production direction, many paper fibers get arranged in the direction of the production process. The result is that paper so produced folds easier in one direction. This effect is called the grain of paper. In order to have a book which opens easily and where the pages turn softly, it is important to have the grain of the paper in the direction of the spine of the book. Concerning the DINformats even numbers have commonly the grain in the height/length of the sheet.

Choosing A Format For The Book
When designing a book one should keep in mind how the book will be used. The format of a book that is read while held in the hand is different from the format of a book that is read while laid open on a table. Hand held books should be taller than they are wide. Two traditional page formats (width : height) for tall books

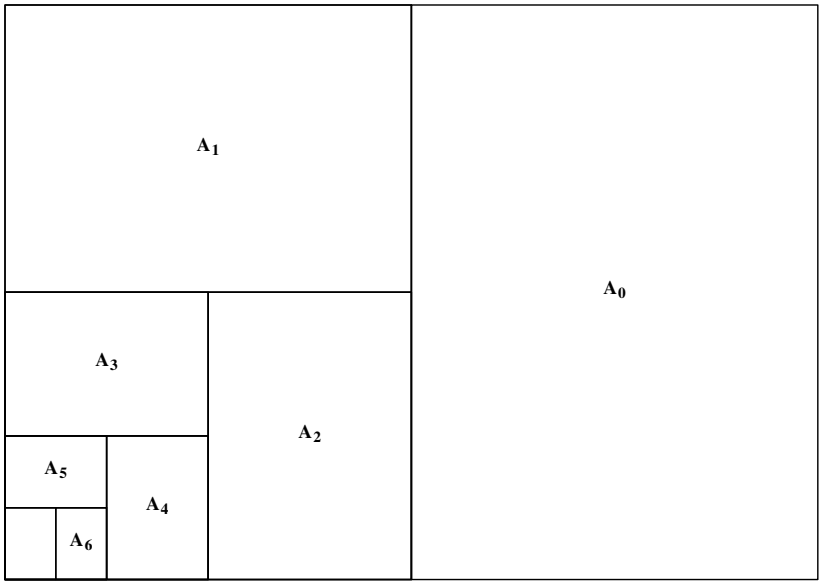


Figure 1 From A_0 to A_6

are are: 21 : 34 (golden ratio) and 2 : 3. For very small books, ratios of 1 : 1.732 ($1 : \sqrt{3}$) or 3 : 5 are fine.

Bad proportions of the page are 3 : 4 or 1 : 1.414 ($1 : \sqrt{2}$) – just try it yourself! Take an A_5 book and read it while holding it in your free hand. . . .

Conversely, large books that are studied laid open on a table can have a page proportion of 3 : 4 without any problem. Oblong books, where the height is less than the width, can also be read laid open on a table.

Figure 2 shows different page proportions.

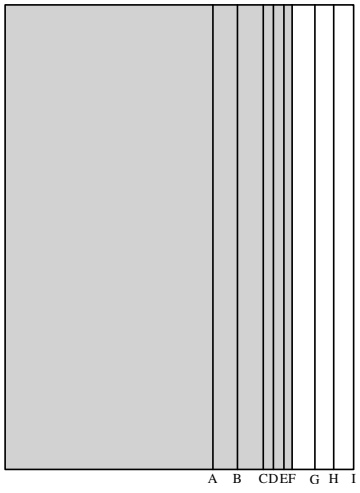


Figure 2 Page proportions (the gray area indicates the golden ratio)

The characters in figure 2 indicate the following ratios between width and height of the page:

A	:	1	: $\sqrt{5}$	
B	:	1	: 2	
C	:	5	: 9	
D	:	1	: $\sqrt{3}$	
E	:	3	: 5	
F	:	21	: 34	(golden ratio)
G	:	2	: 3	
H	:	1	: $\sqrt{2}$	
I	:	3	: 4	

Managing Readability

In order to make life easier for the reader, one should try to make the average line length some 40 to 70 characters long – including spaces. The emphasis lays on the

70 characters including spaces. The 70 characters rule is applicable for different European languages such as English, Dutch and German.

In addition to the number of characters, the number of words in one line should also be considered. For the German language, a line consisting of 8 to 12 words is optimal.

Care should be taken when choosing the font. There are, of course, discussions on whether or not to use sansserif fonts for the main text. The important things are that one should restrict the number of fonts used, and that the fonts should contrast well. The less decorative elements a font has, the more legible it will be. One should avoid setting running texts in calligraphic or italic fonts.

In order to fit the 70 characters on a line, one can choose for fonts which run narrower or broader. Compare texts typeset in Times Roman, which is a narrow-running font developed for the Times newspaper, to the same text typeset in Bookman or Garamond. The examples in the TEX Font Sampler booklet [1] are quite instructive.

Another possibility to fit the line length requirement is to change the font size. Normally font sizes less than 8pts are not easily read.

Placement Of The Typesetting Area On The Paper

Now then, where to place the typesetting area on the page? Typographers did and still do differ in opinion on this subject. It is interesting to know that J. Tschichold in his young years was a promoter of the asymmetrical style of typography associated with the modernist and Bauhaus movements. Later on he started to study medieval manuscripts and printed documents from the middle ages, and completely reversed his opinion. His credo became the harmony of the spread and the page with the printed area. By measuring countless documents he discovered that often the the proportions for the size of the margins (inner, top, outer, and bottom) were: 2,3,4, and 6 respectively. Furthermore he discovered that a *page* with a ratio of 2 : 3 permits a typesetting area whose height is equal to the width of the page – see figure 3).

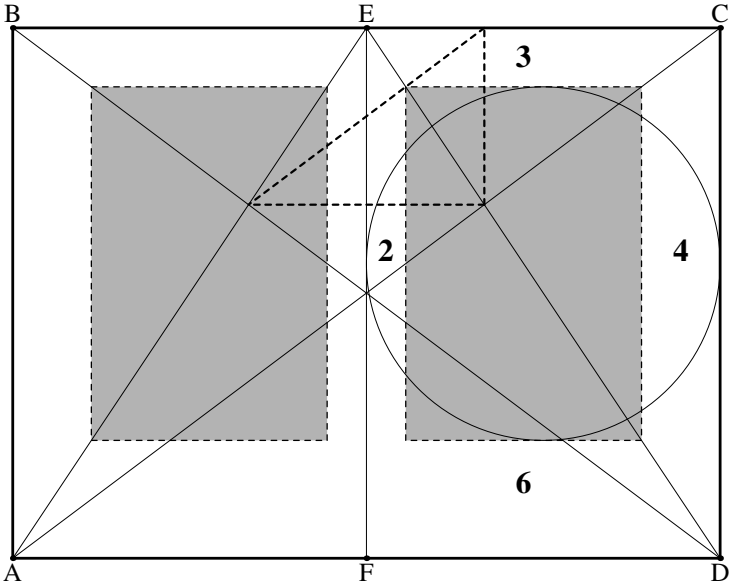


Figure 3 Page proportion 2 : 3, the text block height is equal to the page width.

These principles of book design formed a ('canon') that was used by such early printers as Gutenberg and Schöffer (calligraph).

In order to design a typesetting area that meets the requirements mentioned above, one needs to be able to divide the page width and height into ninths, since the inner top corner of each text block is one-ninth of the way across and one-ninth of the way down the page.

This division has been described by J.A. van de Graaf [7]. In 1955 Tschichold presented another approach to this using the knowledge of Villard de Honnecourt, an architect who lived in the first half of the 13th century, and the studies presented by H. Kayser [8]. The idea is that one can geometrically divide any length into thirds, fifths, and sevenths and so on. This construction was further improved by Goldenheim, Litchfield and Dietrich (GLaD-construction)[2], which yields odd and even divisions in separate diagrams. Kayser combined these methods in a single diagram. A Villard's diagram is presented in figure 4.

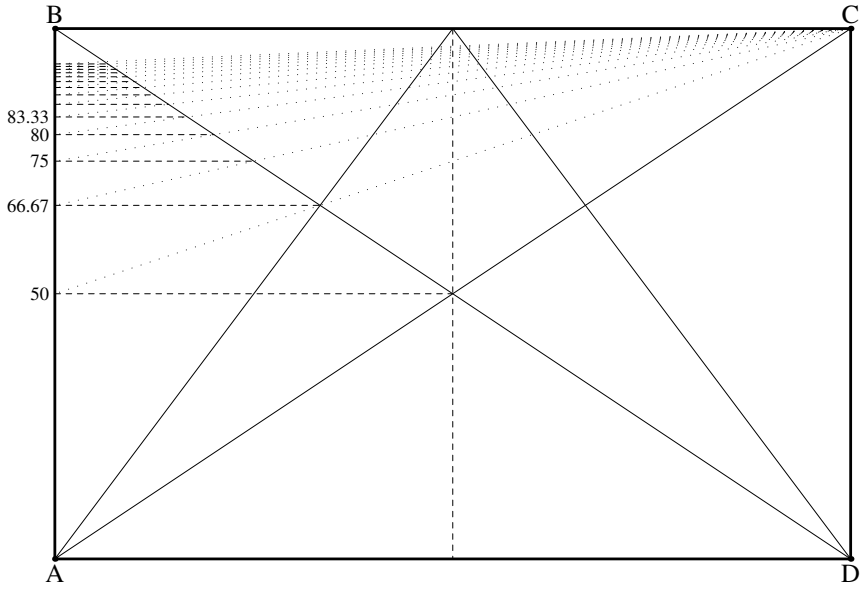


Figure 4 An example diagram according to Villard de Honnecourt

Tschichold applied Villard de Honnecourt's construction recursively, and since (1/3) $\times$ (1/3) = (1/9), this method ensures that the top inner corner of each text block (shown dashed in the figure 5) is located both one-ninth of the way across and down the page. Thus, the recursive Villard construction can be used to determine the size and position of the typesetting area, which have the same aspect ratio as the pages themselves.

Though Tschichold preferred a page ratio of 2 : 3, the same construction method can be applied to any page dimension and paper proportion. Moreover, one not even has to adhere to the division into ninths; other divisions like twelve (see figure 6) will also result in a harmonious proportion between the page and the typesetting area, and the portions of white space around the text block [6].

Determine Necessary Data In MetaPost

To calculate and draw harmonic ratios, the principles presented in the previous section were implemented in a MetaPost program. The program accepts the paper height, the aspect ratio of the spread, and the dividing factor of 9 or 12. Because the graphic dimensions can vary greatly, the program also accepts a scaling factor which tells MetaPost to scale the graphic, while leaving the calculated results unchanged.

Data used in figure 6:

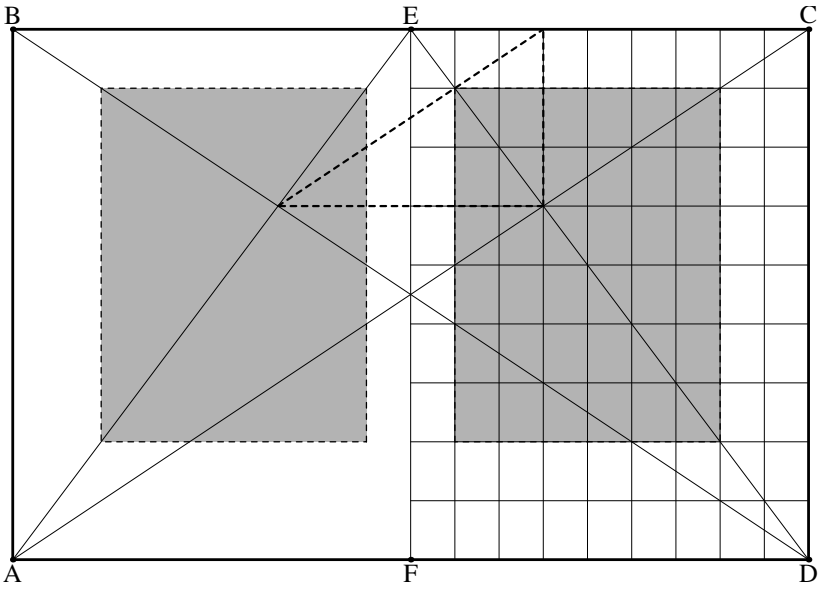


Figure 5 Finding the left upper corner of the typesetting area with the method presented by J. Tschichold (dashed) and J.A. van de Graaf

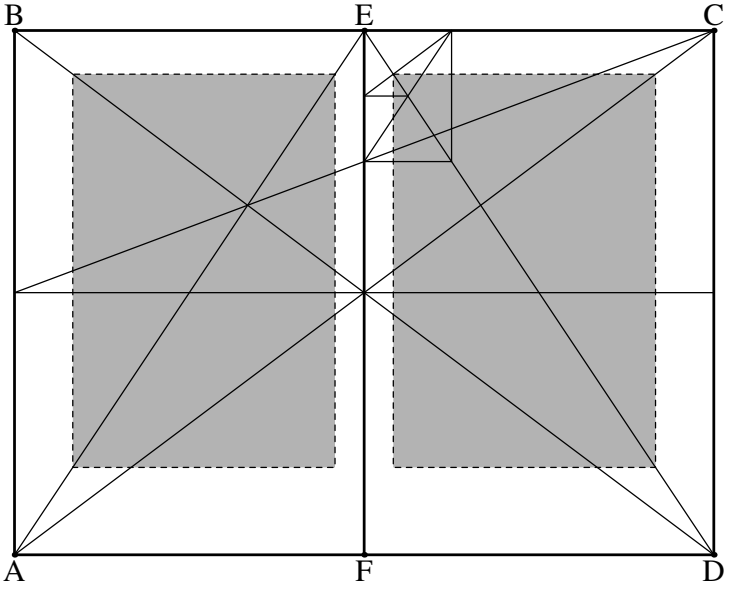


Figure 6 Finding the typesetting area with a division of 12 using the Villard de Honnecourt method

h : 22 cm
proportion A : 3
proportion B : 4
dividing factor : 12

Data for the layout as calculated by MetaPost are given in table 1.

Binding Correction
So far, only a single spread was used to calculate the typesetting area and the white space around it. When the document will be bound into a book, there must be

Proportion (height : width) of paper	3 : 4
Dividing factor	12
Page height (A - B)	220 mm
Page width (A - F)	146.5 mm
Inner margin	12
Top margin	18 mm
Outer margin	24 mm
Lower margin	37 mm
Height of typesetting area	165 mm
Width of typesetting area	110 mm

Table 1 Typesetting data as calculated by MetaPost

a correction for the optical loss of white space at the binding edge. How big this correction must be is difficult to tell because it depends on the weight and thickness of the paper, the thickness of the book, and, last but not least, on the type of binding used. So the binding correction can best be discussed with the printing house and the bindery.

Setup Of A Calculated Typesetting Area In Context

When setting up a Context document following J. Tschichold's principles, the proportions can best be calculated on a separate piece of paper. Draw the actual dimensions of a spread and the single page, and use the Villard geometrical division method to find the one-third and one-ninth point.

Then measure the relevant dimensions.

Another approach is to use MetaPost to calculate the dimensions.

The following `\setuplayout`-command-options are used to typeset the example presented table 1 in Context

```
\setuplayout
  [topspace=18mm,          % top margin
   header=0pt,
   headerdistance=0pt,
   backspace=16mm,         % inner margin + binding correction
   leftmargin=12mm,        % inner margin
   rightmargin=24mm,       % outer margin
   footerdistance=10mm,
   footer=4mm,
   height=179mm,
   width=110mm,
   location=doublesided]
```

As you can see, not all the data presented in table 1 are used in the `\setuplayout`-command-options.

Option	Comment
topspace	As in the table
header / headerdistance	Header and headerdistance get positive values if these have to be included in the typesetting area (running headers)
backspace	The backspace is enlarged by 4 mm for binding correction
left / right margin	As in the table
footer / footerdistance	Empirically set
height	In Context the height always includes header and headerdistance and footer and footerdistance.

width	As in the table
location	The document will be doublesided, and centered on the paper

In the near future Context will provide several predefined page setups. The predefinitions will contain single and doublesided layouts. The doublesided layout will be based on the ninths-division. Other setups will allow the layout to take character widths into account. Since the average number of occurrences of different characters varies between languages, the adjustments will be language dependent.

Setup Of A Decent Layout In Latex

Latex users are encouraged to have a closer look at the KOMA-script package [3][1]. KOMA-script provides single- and doublesided layouts. The doublesided layout is based on the ninths-division. It is possible to have the typesetting area adjusted for the font used (70 chars width of the line). The KOMA-script also accepts a binding correction.

Conclusion

- If the typesetting area must be as high as the width of the page, then the ratio of the paper (spread) must be 3 : 4. In this case the page has the proportion 2 : 3.
- The drawing method developed by van der Graaf en Tschichold can be applied to any paper size and paper proportion.
- The 'canon' of division by 9 or 12 is not mandatory.
- The typesetting area remains harmonious with respect to the page if it is related to the diagonals of the spread and the diagonals of the page.
- The calculated inner margin of a spread must be enlarged by the binding correction for compensation of optical loss of white space due to binding.
- For easy reading, the line length should not exceed 70 characters – including spaces.
- The line-length can be influenced by the font size and the choice of font (narrow-running vs. broad-running).

The interested reader might want to look at some examples of books. The Royal Library in the Hague has a website dedicated to 100 highlights out of their collection [4].

Literature

[1] Hagen H., Egger W.. TEX Font Sampler. NTG, Dante, Gutenberg. 2004.

[2] GLaD-construction: http://world.std.com/~wij/glad/tschichold.html

[3] Kohm M.. Satzspiegelkonstruktion im Vergleich. Die TEXnische Kommödie. 4, 2002. 28 – 48.

[4] Koninklijke Bibliotheek Den Haag: Honderd hoogtepunten uit de Koninklijke Bibliotheek. http://www.kb.nl/kb/100hoogte/index.html.

[5] Taylor Ph.. Book Design for TEX Users. MAPS 19, 19 – 22, 28 – 36. 1997.

[6] Tschichold J.. Ausgewählte Aufsätze über Fragen der Gestalt des Buches und der Typographie. Birkhäuser Verlag Basel. 2. Auflage. ISBN-3-7643-1946-1. 1987.

For further reading as cited by [6]:

[7] Graaf van de J.A.. Nieuwe berekening voor de vormgeving. In: *Tété*. Amsterdam. November, 1946.

[8] Kayser H.. Ein harmonikaler Teilungskanon. Occident Verlag Zürich. 1946.

Jan Tschicholds werk verschenen in het Nederlands:

— Jan Tschichold. Opstellen over typografie. Gerards & Schreurs. 1988.
— Jan Tschichold. De proporties van het boek. De Buitenkant. Amsterdam. ISBN 90 70386 36 4. 1991.

I would like to thank Michael Guravage for proofreading and helping to turn this text into correct English.

Notes

1. A manual is available from http://people.freenet.de/kohm/markus/komasatzspiegel.pdf in German or from Gutenberg on http://www.gutenberg.eu.org/pub/GUTenberg/publicationsPDF/42-kohm.pdf in French.

Willi Egger

TEX and prepress

Abstract
This article discusses preparing documents for professional printing with TEX and pdftex, including color printing and prepress standards.

Keywords
PDF, PostScript, color, Acrobat, separations, overprint

History

Most of us aren't graphics professionals. Still, now and then we have things that need to be printed professionally at a conventional printshop.

A bit of historical perspective: originally, we dealt with this by supplying 'camera-ready' laserprinter output to the printshop, from which printplates were created photographically. This method certainly prevented surprises, but was not the way to get quality output.

During the nineties, PostScript dumps became increasingly popular among TEX users as an alternative. Professional-quality output became a real possibility. But it might take some effort to find a printshop willing to process raw PostScript. The usual practice in the graphics industry was handing off application files. Of course, this had its drawbacks: it was easy to forget to include a font or a graphic file in the job, and the printshop from its side had to watch against reflow, *i.e.* changes in linebreaks. For TEX users, this practice was no option at all.

TEX users have for a long time been using Ghostscript for previewing, converting and printing PostScript. However, most printshops seem to have been unaware of such tools. And without such tools, a PostScript file is pretty much a black box.

Then Adobe developed PDF, a derivative of PostScript, and has had some success in persuading the graphics industry that a PDF-based 'workflow' is the way to go. By now, it is not that hard any more to find printshops accepting jobs in PDF format.

PDF tools

PDF has been developed both as a more tractable format for print production and as a format for various interactive uses. Whereas PostScript is a full-fledged programming language, PDF lacks programming features. Presumably, this made it easier to write software for it, and we certainly have seen a flood of software for PDF. Just pay a visit to `www.planetpdf.com` to convince yourself.

These include of course the Adobe Acrobat programs: the free Reader (which is now named Adobe Reader) and the various commercial editions of Acrobat. All these commercial editions include Distiller for converting PostScript to PDF. As of this writing, the latest versions (6.xx) of the Reader and the other Acrobat programs are available only for Windows and Mac OS X.

Other PDF tools include various third-party Acrobat plugins, for prepress functions such as color separation and page imposition, and for limited editing. Also toolkits/libraries for programmers, some of them open source. There are also commercial and free alternative PostScript-to-PDF converters, Ghostscript not the least among them. Mac OS X Panther contains a command-line utility pstopdf which is quite good. Many programs now can generate PDF directly.

The principal open source PDF readers are Ghostscript (via a suitable frontend such as gv or GSView), and xpdf. The latter is part of a suite. Xpdf itself requires X11, but the rest of the suite consists of some very useful command-line utilities which are also available for Win32. I'll mention some of them below.

Routes to PDF

The principal routes to generate PDF from TEX are:

- from TEX to dvi to PostScript, and then running the PostScript file through Distiller or another PostScript-to-PDF converter
- from TEX directly to PDF, using pdf[e]tex
- from TEX to dvi and then with dvipdfm[x] to pdf. Dvipdfm-cjk, a.k.a dvipdfmx, offers extended support for CJK (Chinese/Japanese/Korean) languages with their huge charactersets.

One reason for choosing the roundabout way via PostScript is when you use PostScript-specific features such as the pstricks package which haven't been adapted to PDF. Another reason is that you may need Distiller's extra prepress-related controls.

If you need pdftex-specific features but also Distiller's controls, then you can go from PDF to PostScript,

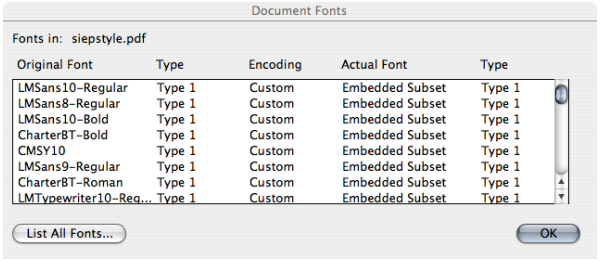


Figure 1. Adobe Reader: Document Fonts

and then back to PDF. For the first conversion, you can use either Adobe Reader or Ghostscript or pdftops (from the xpdf tools suite), for the second one either Distiller or one of its alternatives. This usually works just fine.

Ghostscript as a PDF generator. Many of Distiller's prepress-related controls are also available via Ghostscript; a fairly thorough description can be found in the ps2pdf manual that is included in the Ghostscript distribution.

Preventing font problems

Acrobat used to come with a base set of fonts: Courier, Helvetica, Times, Symbol and Zapf Dingbats. Therefore, these fonts were customarily not embedded. To the dismay of the TEX community, in Acrobat 4 Times was replaced with Times New Roman, and Helvetica with Arial. Grudgingly, we concluded that it was better to avoid ambiguity and embed *all* fonts for print, including the base 14, and just put up with the increase in file size. Fortunately, this version of Acrobat also introduced joboptions files, which were named sets of Distiller settings. This made it easier to switch between generating unambiguous pdfs for prepress and small pdfs for online viewing, where you may prefer to exclude the base-14 fonts.

Another point of concern is MetaFont-generated bitmapped fonts. Although these may look fine in print, they look pretty bad on screen, and PDF validation tools will probably flag them as undesirable or illegal.

Font embedding is controlled by map files. For teTEX/fpTEX/TEX Live, these used to be located under `texmf/dvips/` and `texmf/pdftex/`, but are being relocated to `texmf/fonts/map/`*engine*, *engine* being *e.g.* dvips or pdftex. Make sure that the relevant mapfiles contain entries for the Computer Modern fonts, and that all entries contain a font filename:

```
ptmr8r NimbusRomNo9L-Regu
  " TeXBase1Encoding ReEncodeFont "
  <8r.enc <utmr8a.pfb
```

(a single line), rather than

```
ptmr8r Times-Roman
  " TeXBase1Encoding ReEncodeFont " <8r.enc
```

The first version downloads the URW Times clone included in most free TEX distributions, the second references a version of Times which should be available to either Acrobat or the printer/typesetter.

With the 2003 editions of teTEX/fpTEX/TEX Live, maps are generated with a utility updmap, and configured either by editing `web2c/updmap.cfg` or with command-line parameters. Also check `texmf/pdftex/config/pdftex.cfg` to see which mapfiles are used by pdftex.

Changes are planned for future releases, so check the documentation if things don't work out.

As to MikTeX: The manual mentions updmap.cfg for manual configuration and the command `initexmf --mkmaps` for forcing regeneration of the mapfiles.

You can check your fonts with the Reader by *first scrolling through the entire document* and then either click File/Document Properties/Fonts... or by clicking the right-pointing arrow above the vertical scrollbar and select Document Fonts...; see figure 1.

If Acrobat doesn't support your platform, then use pdffonts from the xpdf suite instead:

```
> pdffonts siepstyle.pdf
name                    type   emb sub uni object ID
----------------------- ------ --- --- --- ---------
GZLRCN+LMSans8-Regular Type 1 yes yes no      10  0
EQOQAE+LMSans10-Bold   Type 1 yes yes no      13  0
...
```

Preventing problems with figures

Included figures also may cause problems:

□ Fonts: keep in mind that included pdfs may also contain fonts and font problems. If a font is embedded in a pdf that you are trying to include, and pdftex complains that it can't find the font, it may be that the font is present in the mapfile but absent from your installation. In that case, create a custom version of the mapfile without the entry. This will hopefully no longer be a problem with version 1.20.

□ Lines with width 0, as produced by several graphics programs when you select 'hairline'. Width 0 means one pixel wide. This looks fine with 300dpi output from a desktop printer, but becomes completely invisible with high-resolution typesetter output. A width of 0.3pt should be safe.

□ Resolution of pixel-based images. With the wrong Distiller settings, they might inadvertently get downsampled to screen resolution.

□ Inappropriate use of jpeg:

The left picture is a jpeg of 1138 bytes, the right one a png of 571 bytes. Jpeg is fine for photographs, but if your image contains large solid areas and sharp transitions, then lossless compression such as used by the png format is probably better.

Some of these problems can be spotted by zooming in on your figures in the Reader.

Page size and other properties

With the traditional LaTeX plus dvips plus Distiller route, you needed to tell all three programs about the desired page size. With pdftex, you only need to specify page dimensions once, in your TeX source. Use the pdftex primitives `\pdfpagewidth` and `\pdfpageheight`, or use the geometry package.

While you are at it, ensure also that the PDF version is no higher than it needs be, since your printshop may not have the latest versions of everything. A good version to aim for is version 1.3, which corresponds to Acrobat 4. This can be set either in pdftex.cfg or in your TeX source:

```
\pdfoptionpdfminorversion=3
```

Again, you can check either with the Reader, using either File/Document Properties/Summary or the Document Summary tab under the right-pointing arrow above the vertical scrollbar; see figure 2.
With the xpdf utilities, use pdfinfo:

```
> pdfinfo siepstyle.pdf
Title:          siepstyle
Creator:        TeX
Producer:       pdfTeX-1.11b
CreationDate:   20040601
ModDate:        20040601
Tagged:         no
Pages:          3
Encrypted:      no
Page size:      595.3 x 756 pts
File size:      148171 bytes
Optimized:      no
PDF version:    1.3
```

Page dimensions (pts) are in 'big points'.

Combining documents

With a journal or a proceedings, it often isn't practical to compile the entire document in a single TeX run. So you may end up with a separate pdf for each paper, which you have to combine into a single pdf somehow.

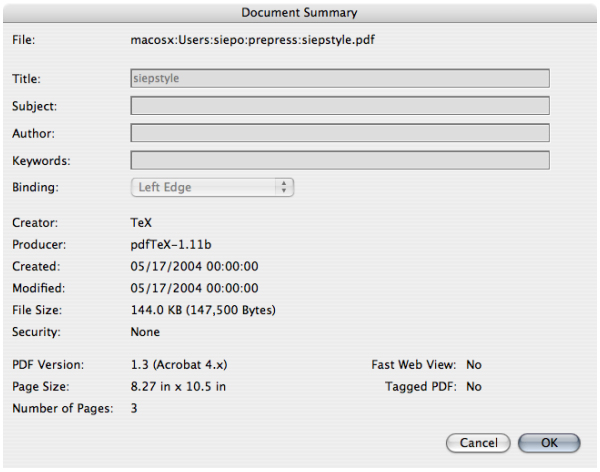


Figure 2. Adobe Reader: page dimensions and PDF version

With TeX. If you have separate pdfs of arbitrary origin then TeX can collate them for you: either use the LaTeX package pdfpages or use the ConTeXt utility texexec with the `--pdfarrange` switch. Including a file with pdfpages can be as simple as

```
\usepackage{pdfpages}
...
\includepdf[pages=-]{APaper}
\includepdf[pages=-]{AnotherPaper}
...
```

The teTeX/fpTeX/TeX Live distributions contain the necessary documentation for pdfpages and texexec.

With a Distiller driver file. Another option is to generate PostScript files and feed Distiller a driver file which loads them. Such a driver file may look as follows:

```
%!
/prun {
  /mysave save def      % save first
  dup = flush           % Shows name of PS file
  RunFile               % builtin Distiller proc
  clear cleardictstack % Cleans up
  mysave restore        % Restores save level
} def

(c:/temp/apaper.ps) prun
(c:/temp/anotherpaper.ps) prun
...
```

This is documented in the Acrobat documentation; see RunDirEx.txt and RunFilEx.ps. The location of these files vary per version and platform.

If you use this approach, it is best *not* to let dvips subset fonts. Then Distiller can create for each font a single subset for the entire volume, leading to a smaller pdf.

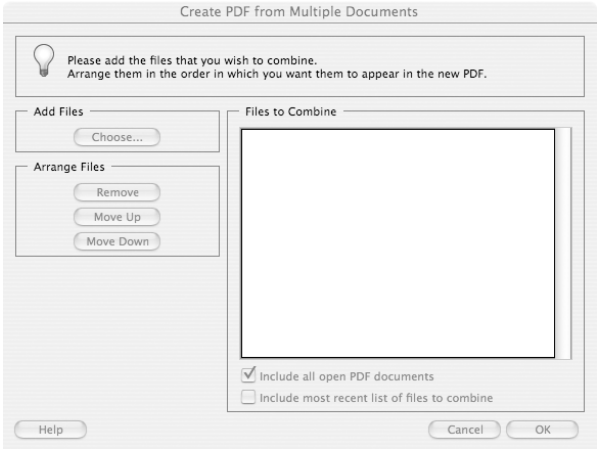


Figure 3. Combining pdfs interactively with Acrobat

With Acrobat. Finally, Acrobat lets you combine pdfs interactively, but since you probably end up repeating it quite a few times, the other options are almost certainly more convenient. See figure 3.

Color separation

If you want your document to be printed in color, then the printshop has to prepare one plate for each ink. For 'full color', these inks are usually cyan, magenta, yellow and black (CMYK). This style of color printing is called process color. The best way by far is to let the printshop handle this itself. After all, they should have the specialized software and the know-how.

However, TEX users do have a few options:

Using macros. You can generate a page several times, each with different definitions for colors:

```
\def\doseparation#1{%
  \ifcase #1 % composite
     \def\sepcyan{cyan}%
     \def\sepblack{black}%
     \def\sepfigure{CKfigure}%
  \or % cyan
     \def\sepcyan{black}%
     \def\sepblack{white}%
     \def\sepfigure{Cfigure}%
       % cyan rendered as black; black omitted
  \or % black
     \def\sepcyan{white}%
     \def\sepblack{black}%
     \def\sepfigure{Kfigure}% cyan omitted
  \fi
{\color{\sepcyan} Text in cyan\par}
{\color{\sepblack} Text in black\par}
\includegraphics{sepfigure}\newpage}

%\doseparation0
% for colored output; omitted for separations
\doseparation1
\doseparation2
```

Note that this requires preseparated external figures.

ConTEXt has macro-based color separation functionality built in; see `www.pragma-ade.com/general/manuals/msplit.pdf`.

Using dvips and colorsep.pro. The TEX Live distribution contains a PostScript header file `texmf/dvips/colorsep/colorsep.pro` for separation of process colors. If you run dvips as follows:

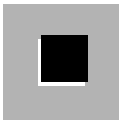
```
dvips -b 4 -h colorsep.pro filename
```

then dvips produces each page four times (`-b 4` switch), and each time the header file colorsep.pro redefines colors appropriately for each of the four printing plates.

Using Acrobat 6 Professional. Acrobat 6 Professional also offers color separation via the Print menu. I encountered some glitches so I recommend to have a really good look at the resulting PostScript- or pdf file before submitting it to your printer.

Overprinting

When printing black over a colored background, color separation software usually sets the other plates to white. However, any misregistration on the press will lead to slivers of white, which might be quite distracting; see the picture below.



If the background is light enough, then you can ignore the effect, but in other cases it is better to do something about it. One solution is to use a modified black with other color components added:

```
\color[cmyk]{0,0.5,0,1}
```

Another solution is to tell PostScript or PDF to let the color continue underneath the black. This is called overprinting. For a LaTEX stylefile and example which *tries* to implement this for dvips and pdftex, look at `http://tex.aanhet.net/overprint/`. You can judge the effect in Acrobat Pro, if you check Advanced/Separation Preview. Figure 4 shows this dialog in another context.

Spot colors

A popular use of color in a printed document is to print some elements such as headings or rules from

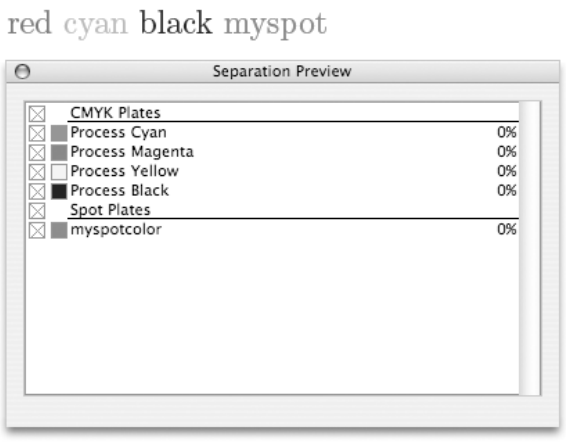


Figure 4. ConTEXt does spotcolors in addition to CMYK

a single premixed color. Printshops have books with color swatches to choose from. Pantone is the manufacturer and license holder of most of these swatch books. You can let one of the process colors, i.e. cyan, magenta or yellow, take the place of the spot color and tell the printshop which color you really want.

If you want spot color *in addition to* process color, then the above trick can't be used. However, ConTEXt offers real support for spot colors. You can do it as follows:

```
\definecolor[myspotcolor][c=.7,m=.2]
\definecolor[myspot][myspotcolor][p=1]
...
\color[myspot]{myspot}
```

Note the two-stage definition of `myspot`: if you want a separation plate for the spot color, you need to define `myspot` as a tint or fraction of a previously defined color. See also figure 4.

Color management

Rgb colors are represented by three values for the three components; process color by four values for the four process inks. These three or four values don't represent color itself but instructions for a device to apply certain colorants. The resulting color can and does depend on the device; we are all familiar with a wall of TV sets in an electronics store each displaying the same image with a different color cast. Matching screen colors with printed colors is a worse problem. We all have seen how screen images can become disappointingly dull when printed; many brilliant screen colors simply cannot be reproduced in print.

Because graphics professionals tend to care about color consistency, color management systems have been introduced, which try to guarantee color consistency from device to device. This means either specifying color in some device-independent way or to supply device profiles to go with the color elements in your document. This is one area where open source doesn't have much to offer.

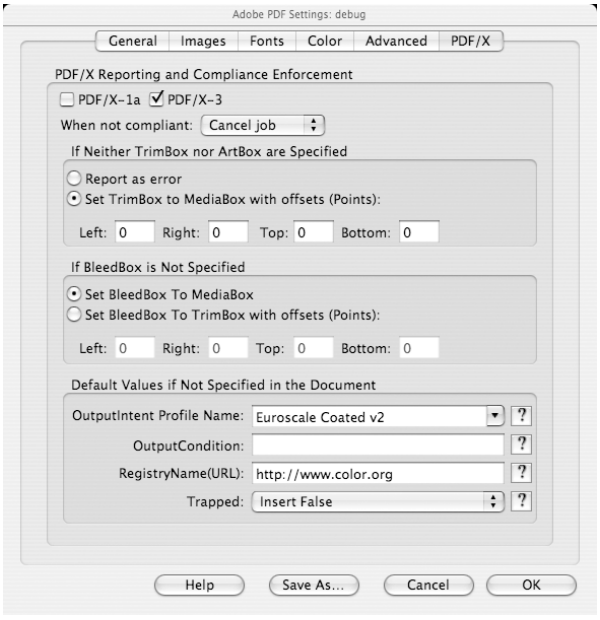


Figure 5. Distiller settings for PDF/X conformance. The Color tab (not shown) also contains relevant settings.

PDF/X and Certified PDF

PDF/X is an ISO standard for pdfs in prepress. There are two flavors: PDF/X-1a which allows process color and spot color, and PDF/X-3 which also accepts color-managed RGB. Since it is an ISO standard, you have to pay money to get the specification. However, you can download documentation and Distiller settings for free from `www.pdf-x.com`.

If you can avoid RGB color altogether, then it is possible to generate PDF/X with pdftex. However, don't convert existing images just for the sake of PDF/X conformance if you don't have to; check with your printshop first.

Code similar to the following should ensure that your pdf won't fail PDF/X for silly reasons:

```
\pdfpagewidth=595.3bp
\pdfpageheight=841.7bp
\pdfpageattr{/TrimBox [ 0 0 595.3 841.7] }

\pdfoptionpdfminorversion=3

\edef\pdfdate{%
  \the\year
  \ifnum \month < 10 0\the\month \else \the\month \fi
  \ifnum \day < 10 0\the\day \else \the\day \fi}
\pdfinfo{%
```

```
  /CreationDate (D:\pdfdate)
  /ModDate (D:\pdfdate)
  /Trapped (False)
  /GTS_PDFXVersion (PDF/X-3)
  /Title (\jobname)}

\pdfcatalog{
  /OutputIntents [ <<
    /Info (Euroscale Coated v2)
    /Type /OutputIntent
    /S /GTS_PDFX
    /OutputConditionIdentifier (OFCOM_PO_P1_F60)
    /RegistryName (http://www.color.org/)
    >> ]}
```

Actually, pdftex 1.11b already includes a creation date automatically. Hopefully, newer versions will do the same for modification date so that you can dispense with the date rigmarole altogether.

Acrobat Distiller also has options for color management and PDF/X; see figure 5.

Another initiative, from Enfocus Software, is Certified PDF. This is not just a set of requirements, but requires your pdfs to be stamped as certified by dedicated commercial software. I found no reference to this type of certification in the Acrobat documentation. See `www.certifiedpdf.net` for more information.

Preflight

The term preflight has become in use for ensuring that your pdf is safe for production. I already mentioned a few simple checks that are available with the Reader and with the xpdf utilities.

Acrobat Professional has a lot of preflight options built-in, including checks on PDF/X compliance. They can be found under the Document menu. Just as with Distiller options, there are also named sets of preflight options; see figure 6.

Much of the Acrobat preflight code has been taken from Callas' PDF/X Inspector. There also used to be a free version of this tool, called PDF/X-3 Inspector.

Conclusion

The main points are to check what you can, and to discuss with your printshop in what form they want your document. Maybe they have a Distiller options file; even if you don't use Distiller, then it would still be useful to look at; these are plain ascii files.

If there is color then it is highly desirable that the printshop is willing to do the separations. The same is true for page imposition.

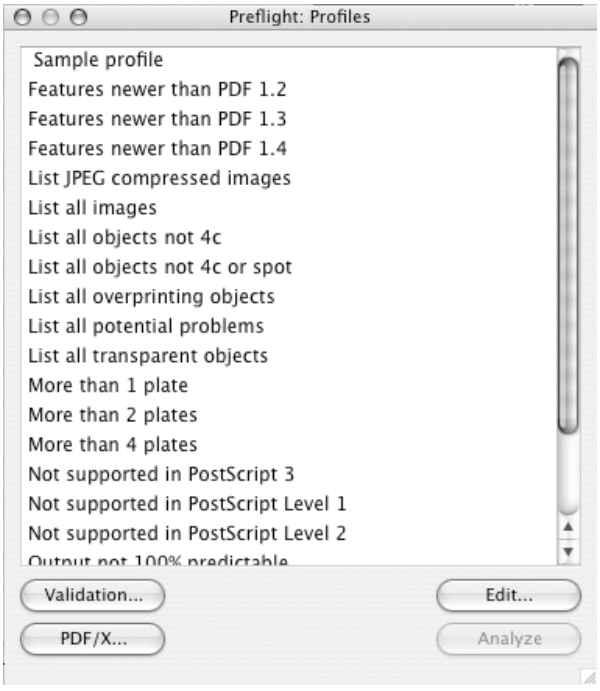


Figure 6. A long list of predefined sets of preflight profiles in Acrobat 6 Pro

Keep also in mind that there are lots of MS Office files which are being typeset somehow, so many printshops ought to be able to handle pdfs from outside the graphics industry.

All this being said, I do believe that Acrobat Pro is a worthwhile investment if you can afford it at all.

URLs

Adobe	`www.adobe.com`
Planet PDF	`www.planetpdf.com`
Xpdf	`www.foolabs.com/xpdf/`
DviPDFm project	`project.ktug.or.kr/dvipdfmx/`
Color separation in Context	`www.pragma-ade.com/general/manuals/msplit.pdf`
overprint.sty	`tex.aanhet.net/overprint/`
PDF/X support	`www.pdf-x.com`
Certified PDF	`www.certifiedpdf.net`
Callas	`www.callas.de/en/`

Siep Kroonenberg
`siepo@cybercomm.nl`

Een tutorial over het gebruik van BIBTEX

Abstract
Dit artikel beschrijft het gebruik van BIBTEX waarbij de nadruk gelegd wordt op die aspecten die voor veel mensen als moeilijk ervaren worden. Het is een uitwerking van een lezing die de auteur gegeven heeft op de NTG-dag in Arnhem op 13 november 2003.

Keywords
BIBTEX, bibliografie, literatuurlijst, referentie, LATEX, citeren, woordenlijsten.

Inleiding

BIBTEX [1] is een programma, geschreven door Oren Patashnik, dat gebruikt wordt om bibliografieën (literatuurlijsten) te beheren voor LATEX documenten. Het idee is om de vorm en de inhoud van de bibliografie van elkaar te scheiden. Hierdoor kan men een soort database van literatuurreferenties aanleggen en naar behoefte hieruit putten voor een specifiek document, terwijl de vormgeving van de literatuurlijst niet bepaald wordt (althans grotendeels) door de database maar door de eisen die aan het document gesteld worden. Het woord 'database' moet hier met een korreltje zout genomen worden: het is niet te vergelijken met de databases die bijvoorbeeld voor een administratie of een website gebruikt worden, maar eerder een eenvoudige kaartenbak, en dan nog van een heel specifieke vorm. De uiteindelijke vorm en inhoud van de literatuurlijst worden bepaald door de volgende onderdelen:

- De inhoud van de bibliografische database (een `.bib` file).
- De referenties die in het LATEX document voorkomen.
- De bibliografische stijl die gebruikt wordt (een `.bst` file).

De bibliografische database

De bibliografische database is een gewoon tekstbestand, meestal in ASCII. De moderne BIBTEX kan ook met 8-bit tekenverzamelingen werken, maar nog niet met Unicode.

Zo'n database bestaat uit een aantal records, voor elk opgenomen werk één. De opgenomen werken kunnen van verschillende soort zijn en elke soort heeft zijn eigen specifieke informatie (velden). Zo heeft een tijdschriftartikel een tijdschrift, en meestal ook een jaargang en nummer. Een boek heeft een uitgever en een jaartal. Bijna alle soorten hebben een auteur of redacteur (editor) en een titel. Welke soorten herkend worden hangt uiteindelijk af van de bibliografische stijl die gebruikt wordt. Het is mogelijk om eigen soorten te definiëren, maar dan moet men wel een bibliografische stijl maken die deze soort herkent en de juiste vormgeving ervoor produceert.

Elke soort wordt in de bibliografische database aangegeven door een keyword dat met @ begint, bijvoorbeeld @Book. Het maakt niet uit of het keyword met hoofdletters, kleine letters of een mix ervan gegeven wordt.

In tabel 1 staat een opsomming van de belangrijkste soorten records en de notatie die BibTeX gebruikt. De verplichte velden zijn ook aangegeven. Welke velden voor elke soort facultatief zijn kan in andere literatuur gevonden worden, zie [1, 2]. Bovendien bestaan er diverse programma's voor het beheren van de bibliografische database files, en die kennen meestal de relevante velden.

Keyword	Soort	Verplichte velden
@Article	Artikel in een tijdschrift	author, title, journal, year
@Book	Boek	author/editor, title, publisher, year
@Booklet	Boekje, bijvoorbeeld zelf uitgegeven	title
@InProceedings	Artikel in conferentie proceedings	author, title
@InCollection	Artikel in een bundel	author, title, booktitle
@InBook	Een deel (bijv. hoofdstuk) in een boek	author/editor, title, chapter, publisher, year
@Proceedings	Conferentie proceedings	title, year
@PhdThesis	Dissertatie	author, title, school, year
@MastersThesis	Afstudeerscriptie	author, title, school, year
@TechReport	Technisch rapport	author, title, institution, year
@Manual	Technisch manual	title
@Unpublished	Ongepubliceerd werk	author, title, note
@Misc	Diversen	

Tabel 1. Soorten werken in BibTeX

In appendix staat een voorbeeld bibliografische database, die overigens niet erg representatief is. Hij bestaat voornamelijk uit pathologische gevallen die verder in dit artikel besproken worden.

Citaties in het document

Als voorbeeld nemen we het volgende document:

```
\documentclass{article}
\begin{document}
As has been shown in \cite{swierstra01combinator} \ldots
\nocite{el_libro_latex}
\bibliographystyle{plain}
\bibliography{bibfile}
\end{document}
```

De uitvoer hiervan is:

As has been shown in [2] ...

References

[1] Bernardo Cascales Salinas, Pascual Lucas Saorín, José Manuel Mira Ros, Antonio José Pallarés Ruiz, and Salvador Sánchez-Pedreño Guillén. *El Libro De LaTeX*. Pearson, 2003.

[2] Doaitse Swierstra. Combinator parsers: From toys to tools. In Graham Hutton, editor, *Electronic Notes in Theoretical Computer Science*, volume 41. Elsevier Science Publishers, 2001.

We zien de volgende BibTeX elementen in dit document:

- `\cite{`*key*`}`
  Een werk dat geciteerd wordt en daarom in de bibliografie wordt opgenomen.
- `\nocite{`*key*`}`
  Een werk dat niet geciteerd wordt en toch in de bibliografie wordt opgenomen. Een alternatieve vorm is `\nocite{*}` waarmee alle werken uit de bibliografische database worden opgenomen zonder geciteerd te worden.
- `\bibliographystyle{`*style*`}`
  De bibliografische stijl: in dit geval 'plain' wat numerieke referenties genereert.
- `\bibliography{`*bibfile(s)*`}`
  De bibliografische database.

Wat betreft de vormgeving valt het volgende op:

- 'El Libro de LaTeX' is met hoofdletters
- 'Combinator parsers: From toys to tools' is met kleine letters (behalve de eerste letter en na de :)
- De referenties zijn gesorteerd op auteursnaam.

De hoofd- en kleine letters worden bepaald door de bibliografische stijl en niet door de database. In dit geval maakt het verschil of het een artikel of een boek betreft.

In het tweede voorbeeld gebruiken we een andere bibliografische stijl

```
\documentclass[openbib]{article}
\begin{document}
As has been shown in \cite{swierstra01combinator} \ldots
\nocite{el_libro_latex}
\bibliographystyle{alpha}
\bibliography{bibfile}
\end{document}
```

en de uitvoer hiervan:

As has been shown in [Swi01] ...

References

[CSLSMR+03] Bernardo Cascales Salinas, Pascual Lucas Saorín, José Manuel Mira Ros, Antonio José Pallarés Ruiz, and Salvador Sánchez-Pedreño Guillén.
El Libro de LaTeX.
Pearson, 2003.

[Swi01] Doaitse Swierstra.
Combinator parsers: From toys to tools.
In Graham Hutton, editor, *Electronic Notes in Theoretical Computer Science*, volume 41. Elsevier Science Publishers, 2001.

Er zijn twee verschillen:

1. De `openbib` optie maakt dat in de bibliografie verschillende elementen op een nieuwe regel beginnen.
2. De `alpha` bibliografische stijl zorgt ervoor dat alfanumerieke verwijzingen gebruikt worden. Andere standaard stijlen zijn `unsrt` (referenties worden niet gesorteerd maar in volgorde van referentie opgenomen), en `abbrv` (voornamen van auteurs worden afgekort). Meer stijlen worden verderop besproken.

Plaats en naam

Normaliter wordt een bibliografie aan het eind van een document geplaatst. Soms worden aparte literatuurlijsten aan het eind van een hoofdstuk of ander onderdeel geplaatst, zie pagina 82 voor meer informatie. In het gewone geval wordt de plaats van de bibliografie bepaald door het `\bibliography` commando. Dit geeft dus zowel aan welke bibliografische database gebruikt als waar de bibliografie komt te staan.

De literatuurlijst wordt bij de standaard klasse 'book' als hoofdstuk gezet en bij de klassen 'article' en 'report' als sectie. Andere klassen doen meestal iets soortgelijks hoewel sommige de mogelijkheid hebben een andere keus te maken bijvoorbeeld als sectie in een klasse met hoofdstukken.

De titel van de bibliografie is bij verstek 'Bibliography' in de klassen 'book' en 'report' en 'References' in 'article'. Bij gebruik van `babel` met de optie '`dutch`' wordt dit resp. 'Bibliografie' en 'Referenties'. Men kan ook eigen termen hiervoor gebruiken door (met `\renewcommand`) resp. `\bibname` (book/report) en `\refname` (article) te herdefiniëren. Bijvoorbeeld:

```
\renewcommand{\bibname}{Literatuurlijst}
```

De titels worden bij verstek zonder nummer gezet en komen niet in de inhoudsopgave. Wil men de bibliografie toch in de inhoudsopgave opnemen dan kan dit met het `\addtocontents` commando. Bijvoorbeeld in de 'article' klasse waar de bibliografie een sectie is:

```
\addcontentsline{toc}{section}{\refname}
\bibliography{...}
```

Er is een kleine kans dat de bibliografie net op een nieuwe pagina begint en dat op deze manier nog net het vorige nummer in de inhoudsopgave verschijnt. Dan moet

men een \newpage hiervoor geven. Bij 'report' en 'book' waar de bibliografie een hoofdstuk is dat in ieder geval op een nieuwe pagina begint gaat dit zonder verdere maatregelen zeker fout. Omdat het voorgaande hoofdstuk nog zwevende figuren of tabellen kan hebben moet hierbij zelfs een \cleardoublepage of clearpage gegeven worden afhankelijk van het feit of hoofdstukken altijd op een rechter- of ook op een linkerpagina beginnen.

```
\cleardoublepage % of \clearpage
\addcontentsline{toc}{chapter}{\bibname}
\bibliography{...}
```

Wanneer men het teveel werk vindt om uit te zoeken of hetzelfde probleem heeft met bijvoorbeeld de index, dan is het waarschijnlijk eenvoudiger om het werk uit te besteden aan het tocbibind package van Peter Wilson [3]. Bij verstek voegt dit package de bibliografie, index, lijst van figuren en tabellen en zelfs de inhoudsopgave zelf toe aan de inhoudsopgave. Met opties kunnen deze onderdrukt worden. Wil men bijvoorbeeld de inhoudsopgave en de lijst van tabellen niet opnemen dan kunnen de opties nottoc en notlot gebruikt worden. Andere opties die van belang zijn voor de bibliografie zijn: chapter en section om de bibliografie als hoofdstuk resp. sectie op te nemen en numbib om de bibliografie een sectie- of hoofdstuknummer te geven.

De opbouw van de bibliografische database

Zoals in appendix te zien is bestaat elk record in de database uit het keyword gevolgd door een sleutel (*'key'*) en de velden tussen accolades of ronde haakjes. De *key* kan zelf gekozen worden en wordt gebruikt als argument in het \cite commando. De velden worden gescheiden door komma's (het laatste veld mag ook door een komma gevolgd worden). Een veld bestaat uit de naam gevolgd door een =-teken en de waarde.

Voor de veldwaarden kunnen gebruikt worden:

- Strings tussen "".
  `publisher = "Elsevier Science Publishers"`
  De "" worden verwijderd. Aanhalingstekens mogen alleen binnen accolades voorkomen, ook wanneer het \" betreft.
- Strings tussen { }.
  `title = {Combinator Parsers:  From Toys to Tools}`
  De buitenste { } worden verwijderd.
  In beide soorten strings mogen geneste accolades voorkomen.
- Getallen zonder teken.
  `year = 2003`
- Macro's.
  `month = jan`
  Macro's worden gedefinieerd in de bibliografische stijl of in de bibliografische database met het @string commando:
  ```
  @string(jan = "Januari")
  @string{acmtr = "ACM Transactions on "}
  ```
- Bovenstaande items kunnen gecombineerd worden met de concatenatie-operator #:
  `journal = acmtr # "Multimedia"`

Behalve de bibliografische records kan de database nog bevatten:

- De reeds genoemde @string entries; deze hebben als inhoud hetzelfde als

een veld.

- Een `@Preamble`, die extra TEX code kan bevatten die in het document geplaatst wordt voor de bibliografie. Hierin kunnen bijvoorbeeld TEX macro's gedefinieerd worden die in de records gebruikt worden. Verderop wordt hiervan een voorbeeld gegeven.
- `@Comment` waarin commentaar staat.
- Alle tekst vanaf het begin of het eind van een entry tot de volgende `@` wordt ook als commentaar beschouwd, maar kan natuurlijk niet het `@`-teken bevatten, in tegenstelling tot de `@comment` entries.

Namen van personen

Namen van personen komen voor in de velden `author` en `editor` (en mogelijkerwijs andere). Wanneer deze velden meer dan één persoon bevatten dan moeten deze worden gescheiden door 'and', en niet door een komma (een fout die men in het begin gemakkelijk maakt). De bibliografische stijl bepaalt hoe de scheiding tussen de auteurs in het uiteindelijke document weergegeven wordt. Dit kan bijvoorbeeld met komma's zijn of met een taalafhankelijk woord. In sommige stijlen wordt bij een groot aantal auteurs zelfs een deel van de auteurs vervangen door *et. al.* of iets dergelijks. Wanneer met zelf *et. al.* wil weergeven dan moet dit met de tekst 'and others', die door BIBTEX dan weer omgezet wordt in de juiste (evt. taalafhankelijke) constructie.

Elke naam bestaat in BIBTEX in principe uit 4 delen, waarbij sommige leeg kunnen zijn: voorna(a)m(en), van, achternaam, toevoeging (bijv. Jr.). Titels zoals Prof. Dr. worden niet ondersteund (zie pagina 72 voor trucs om deze toch erin te krijgen). Er zijn drie toegestane vormen om namen in de bibliografische database te noteren:

- Piet van Oostrum
- van Oostrum, Piet
- van Oostrum, Jr, Piet

Het 'van' gedeelte wordt herkend doordat het met kleine letters geschreven wordt.

Namen die hiervan afwijken of die letters met accenten of vreemde tekens bevatten moeten apart behandeld worden. Deze aparte tekens moeten binnen een eigen accoladegroep behandeld worden. Ook samengestelde achternamen moeten zorgvuldig behandeld worden anders interpreteert BIBTEX deze misschien als voornaam.

Bijvoorbeeld: Bernardo Cascales Salinas. Hierbij is de achternaam (zoals in Spaanstalige landen gebruikelijk is) 'Cascales Salinas'. Zonder tegenbericht zal BIBTEX echter 'Cascales' als tweede voornaam interpreteren. We kunnen dit voorkomen door de naam te schrijven als: 'Cascales Salinas, Bernardo' of eventueel 'Bernardo {Cascales Salinas}'. Wat tussen accolades staat wordt door BIBTEX als ondeelbare eenheid beschouwd. Namen met een 'van' zijn in principe geen probleem, bijvoorbeeld 'Peter van Emde Boas'.

Een ander voorbeeld is 'Xavier {Sala-i-Martin}', hierbij is de achternaam tussen accolades geschreven om te voorkomen dat BIBTEX de 'i' in 'I' verandert.

BIBTEX let zelfs bijzonder op delen die tussen accolades staan en met een \ beginnen. Bijvoorbeeld '`Jos{\'e}`. BIBTEX kent de standaard LATEX geaccentueerde letters en behandelt deze dan ook als één letter. Bij andere commando's wordt voor het sorteren wordt de tekst na het commando gebruikt. Dit geldt alleen voor het eerste niveau accolades (afgezien van de eventuele buitenste accolades van de veldwaarde), en niet voor accolades die dieper genest zijn.

In de voorbeeld database vinden we ook '`Kre\v{s}imir {\v{Z}}igi\'{c}`'.

Hierbij zijn de `\v{s}` en de `\'{c}` niet tussen accolades geschreven hoewel dat eigenlijk wel aan te bevelen is. BibTeX geeft deze letterlijk door naar LaTeX zodat uiteindelijk toch de juiste tekst in de bibliografie gezet wordt (Krešimir Žigić). Als we dit echter bij `\v{Z}` ook gedaan zouden hebben dan zou BibTeX de naam op de 'v' sorteren in plaats van op de 'Z'. Bovendien zou de 'v' in 'V' veranderd worden en dus een onbekend commando worden.

Namen moeten zo mogelijk voluit geschreven worden, want BibTeX zal de voornamen afkorten tot voorletters wanneer de bibliografische stijl dat vereist. Hierbij worden samengestelde voornamen als Jean-Paul afgekort als J.-P. Zie hieronder een voorbeeld met de `abbrv` stijl:

As has been shown in [2] ...

References

[1] B. Cascales Salinas, P. Lucas Saorín, J. M. Mira Ros, A. J. Pallarés Ruiz, and S. Sánchez-Pedreño Guillén. *El Libro de LaTeX*. Pearson, 2003.

[2] D. Swierstra. Combinator parsers: From toys to tools. In G. Hutton, editor, *Electronic Notes in Theoretical Computer Science*, volume 41. Elsevier Science Publishers, 2001.

Moeilijke gevallen

In Sommige namen wordt het 'van' deel met hoofdletters geschreven: 'Juan De La Torre'. We kunne hierbij een truc met een macro gebruiken:
`Juan {\MakeUppercase{d}e La} Torre`.
Omdat we hierbij een accoladegroep op het eerste niveau hebben dat met een \ begint is de bovenstaande regel van toepassing. BibTeX kijkt dus naar de eerste letter na de macro (d) en ziet dit als 'van' deel.

Titels zoals 'Lord' hebben ook speciale behandeling nodig om niet als voor- of achternaam geïnterpreteerd te worden. Bijvoorbeeld 'Lord Kelvin' zou in een `abbrv` stijl als 'L. Kelvin' geschreven worden. We kunnen schrijven '{Lord} Kelvin', waarbij weliswaar Lord nog steeds als voornaam geïnterpreteerd wordt, maar in ieder geval niet afgebroken. Bovendien wordt gesorteerd op 'K'.

Een ander mogelijkheid is het gebruik van een macro. Deze truc kan ook in andere situaties gebruikt worden om de sorteringsvolgorde te beïnvloeden.

```
@preamble{"\newcommand{\noopsort}[1]{}"}
@book{...
  author = "\noopsort{Kelvin}Lord Kelvin"
```

Hierbij genereert de `\noopsort{Kelvin}` geen uitvoer, maar de sorteersleutel wordt 'KelvinLord Kelvin' wat in de meeste situaties goed genoeg zal zijn.

Een andere macro-truc is:

```
@preamble{"\newcommand{\Lord}[1]{Lord #1}"}
@book{...
  author = "{{\Lord{Kelvin}}}"
}
```

In dit geval is de sorteersleutel gewoon 'Kelvin'. Let op: wanneer gebruikt op het buitenste niveau dan kan BibTeX de gebruikte macro van 'case' wijzigen.

Andere velden (in het bijzonder titels)

De regels over accenten en accolades zijn op andere velden op de zelfde manier van toepassing. Omdat BIBTEX hoofdletters in kleine letters kan veranderen en omgekeerd (afhankelijk van de stijl) moet je voorzichtig zijn met macro's in deze velden. Het beste is om deze binnen accolades te schrijven: `\TeX` zou anders wel eens in `\tex` veranderd kunnen worden. Binnen accolades doet BIBTEX dit echter niet. Ook afkortingen zoals 'NTG' moeten daarom binnen accolades geschreven worden.

Andere bibliografische stijlen

Er zijn heel veel verschillende bibliografische stijlen: bijna elk tijdschrift dat LATEX accepteert heeft een eigen bibliografische stijl. Sommige stijlen hebben een bijbehorend LATEX package dat ook in het document gebruikt moet worden. Enkele voorbeelden zijn:

- `agsm.bst`: Australian Government Style Manual
- `chicago.bst`: Chicago Manual of Style (gebruikt `chicago.sty`)
- `apalike.bst`: American Psychology Association (gebruikt `apalike.sty`)
- `kluwer.bst`: Kluwer (gebruikt `harvard.sty`)
- `nederlands.bst`: Sorteert op achternaam, niet op 'van' zoals de meeste andere, en vertaalt termen in het Nederlands
- `cite.sty`: is niet een bibliografische stijl, maar een package dat numerieke referenties kan sorteren en samenvoegen, bijvoorbeeld [1, 3, 2, 6] ⇒ [1–3, 6]

Het zelf schrijven van een bibliografische stijl of zelfs het aanpassen van een bestaande is niet iets dat aan de gemiddelde gebruiker aanbevolen wordt. In plaats daarvan is het meestal beter om een bibliografische stijl de nemen die geparametriseerd kan worden, of het `custom-bib` pakket te gebruiken. Hierin zit een `makebst.tex` waarmee eigen stijlen gemaakt kunnen worden. Door gewoon in een commandoregel het commando 'tex makebst' te gebruiken krijgt men een dialoog waarbij de keuzes voor de diverse elementen van een stijl gegeven kunnen worden. Bijvoorbeeld over de hoofdlettering, de sortering, cursivering van velden etc. Uiteindelijk komt er een op maat gemaakte bibliografische stijl uit. De meeste wensen op bibliografisch stijlgebied kunnen hiermee verwezenlijkt worden. Zie pagina 78 voor meer details.

Twee belangrijke parametriseerbare bibliografische stijlen die ik hier nog wil bespreken zijn 'natbib' en 'jurabib'. Eigenlijk zijn dit meer LATEX packages met een configureerbare bibliografische layout dan BIBTEX stijlen in strikte zin.

Natbib

Natbib [4] is geschreven door Patrick W. Daly als vervanger van een groot aantal oudere stijlen en heeft de functionaliteit daarvan in zich. Het heeft een aantal eigen bibliografische stijlen die als vervanger van standaard LATEX stijlen gebruikt kunnen worden, nl. `plainnat`, `abbrvnat` en `unsrtnat` als vervanger van `plain`, `abbrv` en `unsrt`. Het kan echter ook met verschillende andere bibliografische stijlen gebruikt worden, bijvoorbeeld `harvard`, `apalike`, `chicago` en `agsm`. Het verschil is dan dat `natbib` ook andere citatiestijlen toelaat dan deze stijlen oorspronkelijk ondersteunen. De citatiestijl van natbib is bij verstek de 'auteur+jaar' stijl die in verschillende natuurwetenschappen gebruikelijk is. Bij gebruik van `natbib` samen met een bibliografische stijl is de werkverdeling globaal als volgt: `natbib` bepaalt

hoe de citaties eruit zien, terwijl de bibliografische stijl bepaalt hoe de literatuurlijst eruit ziet.

We beginnen met het volgende voorbeeld:

```
\documentclass{article}
\usepackage{natbib}

\begin{document}
As has been shown by \citet{swierstra01combinator} \ldots\\
As we have seen \citep[chapter~2]{el_libro_latex}\\
All the authors \citep*{el_libro_latex}
\bibliographystyle{newapa}
\bibliography{bibfile}
\end{document}
```

We gebruiken hier:

- Het `natbib` package
- De `newapa` bibliografische stijl
- Verschillende citeercommando's: `\citet` (in tekst citatie), `\citep` (citatie tussen ronde haakjes), `\citep*` (met een `*` wordt de gehele auteurslijst gegeven als de bibliografische stijl dit ondersteunt).
- Facultatieve parameters [chapter 2]

Bovenstaande voorbeeld geeft de volgende uitvoer:

As has been shown by Swierstra (2001) ...
As we have seen (Cascales Salinas et al., 2003, chapter 2)
All the authors (Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz and Sánchez-Pedreño Guillén, 2003)

References

Cascales Salinas, B., Lucas Saorín, P., Mira Ros, J. M., Pallarés Ruiz, A. J., and Sánchez-Pedreño Guillén, S. (2003). *El Libro de LaTeX*. Pearson.

Swierstra, D. (2001). Combinator parsers: From toys to tools. In Hutton, G. (Ed.), *Electronic Notes in Theoretical Computer Science*, volume 41. Elsevier Science Publishers.

Er zijn nog verschillende andere citeercommando's, met verschillende layouts en bijvoorbeeld voor alleen auteur of alleen jaar. Zie tabel 2. Ook zijn er varianten die met een hoofdletter geschreven worden zoals `\Citet`, die een 'van' in een naam met een hoofdletter uitvoeren.

Natbib heeft verder nog parametriseermogelijkheden zoals:

- Het definiëren van de interpunctie:
  `\bibpunct{[}{]}{/}{a}{,}{;}`
  Dit definieert resp: het openingshaakje als [, sluithaakje als], interpunctie tussen de referenties als meer dan één citatie in hetzelfde commando gegeven wordt, de stijl (a = auteur+jaar), de interpunctie tussen auteur en jaar als komma, de interpunctie tussen letters achter een jaar als puntkomma (bijvoorbeeld 2004a;b).
- Aliassen van citaties (d.w.z. een verwijzing naar `el_libro_latex` kan dan

`\citet {el_libro_latex}`	Cascales Salinas et al. (2003)
`\citet [chap.~2]{el_libro_latex}`	Cascales Salinas et al. (2003, chap. 2)
`\citet *{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén (2003)
`\citet *[chap.~2]{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén (2003, chap. 2)
`\citep {el_libro_latex}`	(Cascales Salinas et al., 2003)
`\citep [chap.~2]{el_libro_latex}`	(Cascales Salinas et al., 2003, chap. 2)
`\citep *{el_libro_latex}`	(Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén, 2003)
`\citep *[chap.~2]{el_libro_latex}`	(Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén, 2003, chap. 2)
`\citealt {el_libro_latex}`	Cascales Salinas et al. 2003
`\citealt [chap.~2]{el_libro_latex}`	Cascales Salinas et al. 2003, chap. 2
`\citealt *{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén 2003
`\citealt *[chap.~2]{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén 2003, chap. 2
`\citealp {el_libro_latex}`	Cascales Salinas et al., 2003
`\citealp [chap.~2]{el_libro_latex}`	Cascales Salinas et al., 2003, chap. 2
`\citealp *{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén, 2003
`\citealp *[chap.~2]{el_libro_latex}`	Cascales Salinas, Lucas Saorín, Mira Ros, Pallarés Ruiz, and Sánchez-Pedreño Guillén, 2003, chap. 2
`\citeauthor {el_libro_latex}`	Cascales Salinas et al.
`\citeyear {el_libro_latex}`	2003

Tabel 2. Verschillende citeercommando's in `natbib`

bijvoorbeeld als 'El Libro' uitgevoerd worden).

- Verschillende citatiestijlen: auteur+jaar, numeriek, alfanumeriek.
- Met de opties `numbers,sort&compress` kan hetzelfde effect bereikt worden als met `cite.sty`
- Met de `hyperref` en `hypernat` packages kun je hyperlinks op de citaties krijgen

Hier een voorbeeld waarbij de interpunctie is aangepast:

```
\documentclass{article}
\usepackage{natbib}
\bibpunct{[}{]}{/}{a}{,}

\begin{document}
As has been shown by \citet{Dat95,Del95}, \ldots\\
Also \cite{swierstra01combinator} \ldots\\
\Citet{helm98} has shown that \ldots
\bibliographystyle{apalike}
\bibliography{bibfile}
\end{document}
```

Bovenstaande voorbeeld geeft de volgende uitvoer:

As has been shown by Date [1995]/ Delobel et al. [1995], ...
Also Swierstra [2001] ...
Van der Helm [1998] has shown that ...

References

Date, C. J. (1995). *An Introduction to Database Systems*, volume I. Addison-Wesley Publishing Company Inc., Reading, Massachusetts, 6 edition.

Delobel, C., Lécluse, C., and Richard, P. (1995). *Databases: From Relational to Object-Oriented Systems*. International Thomson Publishing, Londen.

Swierstra, D. (2001). Combinator parsers: From toys to tools. In Hutton, G., editor, *Electronic Notes in Theoretical Computer Science*, volume 41. Elsevier Science Publishers.

van der Helm, F. C. (1998). *Test bibstyle*. IK.

Hier is nog wel een probleem dat 'van der Helm' bij de 'v' gesorteerd wordt en niet bij de 'H'. We zullen verderop zien (pagina) hoe dit probleem aangepakt kan worden. Dit valt echter buiten de competentie van `natbib` zelf. We zullen hiervoor een aangepaste bibliografische stijl moeten gebruiken.

Jurabib

`Jurabib` [5] van Jens Berger (met hulp van vele anderen) is een package met bijbehorende bibliografische stijlen die de manier van citeren ondersteunt die in de juridische wereld en de humaniora gebruikelijk is. Enkele kenmerken hiervan zijn het gebruik van de auteursnaam als citatie in de tekst, eventueel aangevuld met de (verkorte) titel wanneer de auteursnaam ambigu is, citaties als voetnoten, en het gebruik van '*ibid.*' of '*op. cit.*' wanneer eenzelfde werk meerdere keren achter elkaar geciteerd wordt.

`Jurabib` was oorspronkelijk bedoeld als bibliografische stijl voor de `jura` klasse, maar kan ook gebruikt worden met de standaard LaTeX documentklassen en de koma-script klassen.

We geven hier een voorbeeld:

```
\documentclass{article}
\usepackage[ibidem]{jurabib}

\begin{document}
As has been shown by \cite{swierstra01combinator} \ldots\\
As we have seen \footcite[chapter~2]{el_libro_latex}\\
Again it has been argued \footcite[chapter~3]{el_libro_latex} \ldots\\
This is a full cite: \fullcite{el_libro_latex}
\bibliographystyle{jurabib}
\bibliography{bibfile}
\end{document}
```

De uitvoer hiervan is:

As has been shown by Swierstra ...
As we have seen[1]
Again it has been argued[2] ...
This is a full cite: Cascales Salinas, Bernardo et al. El Libro de LaTeX. Pearson, 2003

References

Cascales Salinas, Bernardo et al.: El Libro de LaTeX. Pearson, 2003

Swierstra, Doaitse: Combinator Parsers: From Toys to Tools. In **Hutton, Graham, editor:** Electronic Notes in Theoretical Computer Science. Volume 41, Elsevier Science Publishers, 2001 ⟨URL: `http://math.tulane.edu/~entcs/`⟩

[1] Cascales Salinas et al. chapter 2.
[2] Ibid. chapter 3.

We zien hier:

- Een citatie in de tekst (alleen auteursnaam) met het commando `\cite`
- Twee citatie als voetnoot met het commando `\footcite`
- De tweede van die als '*ibid.*' uitgevoerd (gestimuleerd door de `ibidem` optie)
- Een citatie als volledige titel met het commando `\fullcite`
- De auteurs in de bibliografie zijn vet gedrukt. Dit is parametriseerbaar.

Er zijn veel uitbreidingen en parameterisatiemogelijkheden in `jurabib`. Opties kunnen gegeven worden in het `\usepackage` commando of in een apart `\jurabibsetup` commando in het document of een configuratiefile. Zo hebben de commando's `\cite` en `\footcite` een extra optionele parameter die de editor(s) van een aangehaald werk aangeven; deze wordt dan aan de auteurs toegevoegd in de citatie. Met behulp van de `natoptargorder` optie kan deze ook als eerste optionele argument gegeven worden.

In de bibliografische database kunnen extra velden worden opgenomen:

- `shortauthor` definieert een afkorting van de auteurs die gebruikt wordt in de citaties; wanneer deze niet gegeven is probeert `jurabib` zelf de afkorting te maken
- `shorttitle` geeft een afkorting van de titel voor de citaties.

Andere citeercommando's:

- `\cite*` citeert altijd zonder titel, ook al is dat ambigu.
- `\citetitle` citeert altijd met titel, ook al zou dit niet nodig zijn.
- `\citetitlefortype` declareert voor welke entry types altijd met titel geciteerd moet worden.
- `\citenotitlefortype` declareert voor welke entry types nooit met titel geciteerd moet worden.
- `\footcite` citeert als voetnoot en is ruwweg equivalent met `[1]`. Echter dit commando haalt voorafgaande witruimte weg en voegt een punt toe.
- `\footcite*` en `footcitetitle` zijn de tegenhangers van `\cite*` en

`\citetitle`.

- `\fullcite` en `\footfullcite` nemen de hele bibliografische entry op.
- `\citefield` geeft de mogelijkheid om een veld uit de referentie te citeren zoals author, shortauthor, title, shorttitle, year en enkele andere.
- Er zijn ook enkele `natbib`-achtige citeercommando's als `\citep` die dezelfde uitvoer als `natbib` genereren. Zowel de `t` als `p` versies met en zonder `al` en de `author` en `year` versies. Deze zijn ook als `\footcite` variant beschikbaar.

Het aantal mogelijkheden is teveel om hier op te noemen. Enkele parametrisatie-mogelijkheden wil ik nog noemen:

- `authorformat` geeft de mogelijkheid om de auteurs niet vetgedrukt, maar met kleinkapitalen of cursief af te drukken in de bibliografie.
- `titleformat` geeft de mogelijkheid de layout van titels in de bibliografie aan te passen
- `ibidem` geeft aan of '*ibid.*' gebruikt moet worden en hoe (bijvoorbeeld alleen op twee tegenover elkaar liggende pagina's).
- `idem` is vergelijkbaar maar wordt gebruikt als de auteur hetzelfde is als in de vorige citatie.
- `opcit` geeft aan of '*op. cit.*' gebruikt moet worden en wanneer.
- `bibformat` geeft verschillende layoutopties voor de bibliografie bijvoorbeeld genummerd, het onderdrukken van herhaalde auteurs, uitlijning.

`Jurabib` heeft ondersteuning voor de volgende talen: Engels, Duits, Frans, Nederlands, Spaans en Italiaans. Andere talen zijn gemakkelijk toe te voegen.

Voor mensen die deze manier van citeren moeten gebruiken of het misschien zelfs wel prettig vinden is dit een onmisbaar pakket.

Het maken van een eigen bibliografische stijl

`Custom-bib` [6] van Patrick W. Daly is een pakket waarmee eigen bibliografische stijlen gemaakt kunnen worden. Hoewel hiermee niet elke denkbare bibliografische stijl hiermee gemaakt kan worden heeft dit pakket voor de meeste wensen wel een optie. Opties zijn welk font voor titels, auteurs e.d. gebruikt wordt, of er numerieke of alfabetische verwijzingen gebruikt worden, of de volledige voornaam of alleen voorletters gebruikt worden, of deze voor of na de achternaam komt, etc. Men beruikt dit door het commando 'tex makebst' uit te voeren in een commando-regel en dan de (meerkeuze) vragen te beantwoorden. Er zijn zo'n 32 vragen met gemiddeld ongeveer 7 mogelijke antwoorden per vraag. Op deze manier kunnen zo'n 30 miljard verschillende bibliografische stijlen gegenereerd worden. De auteur blijkt ook zeer bereid te zijn opties toe te voegen als met een situatie tegenkomt die nog niet gedekt is. Er zijn ook gepredefinieerde instellingen voor diverse talen. Momenteel zijn dat Engels, Nederlands, Catalaans, Deens, Esperanto, Fins, Frans, Duits, Italiaans, Noors, Pools, Portugees, Sloveens en Spaans.

Hierbij een voorbeeld van een paar vragen en bijbehorende antwoorden:

```
ORDERING OF REFERENCES (if author-year citations)
(*) Alphabetical by all authors
(l) By label (Jones before Jones and James before Jones et al)
(k) By label and cite key instead of label and title, as above
(d) Year ordered and then by authors (for publication lists)
(r) Reverse year ordered and then by authors (most recent first)
(c) Citation order (unsorted, only meaningful for numericals)
```

```
  Select:

\ans=d
  You have selected: Year ordered

ORDER ON VON PART (if not citation order)
(*) Sort on von part (de la Maire before Defoe)
(x) Sort without von part (de la Maire after Mahone)
  Select:

\ans=x
  You have selected: Sort without von part
```

Op deze manier heb ik eens vrij simpel een bibliografische stijl gemaakt die grotendeels hetzelfde is als de `agsm` stijl, maar de Nederlandse sortering gebruikt ('van Oostrum' bij de 'O' gesorteerd in plaats van bij de 'v'). Het enige dat eigenlijk ontbrak is een file met de antwoorden die bij verstek gekozen worden voor een standaard stijl zoals `agsm`. Wanneer iemand echter een keer de antwoorden gegeven heeft worden deze in een file bewaard, en zou dus in principe gedistribueerd kunnen worden. Er is echter nog geen mechanisme om aan te geven dat deze natwoorden dan bij verstek in een volgende sessie gebruikt moeten worden, maar ze kunnen dan wel met een teksteditor aangepast worden. Zie het volgende voorbeeld voor het gebruik van deze aangepaste stijl, waarbij 'van der Helm' nu voor 'Swierstra' gekomen is:

```
\documentclass{article}
\usepackage{natbib}

\begin{document}
As has been shown by \citet{Dat95,Del95}, \ldots\\
Also \cite{swierstra01combinator} \ldots\\
\Citet{helm98} has shown that \ldots
\bibliographystyle{agsmnl}
\bibliography{bibfile}
\end{document}
```

met als uitvoer:

As has been shown by Date (1995); Delobel et al. (1995), ...
Also Swierstra (2001) ...
Van der Helm (1998) has shown that ...

References

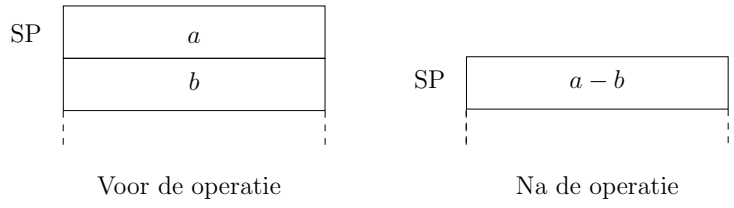
Date C.J. (1995), *An Introduction to Database Systems*, vol. I, Addison-Wesley Publishing Company Inc., Reading, Massachusetts, 6 edn.

Delobel C., Lécluse C. & Richard P. (1995), *Databases: From Relational to Object-Oriented Systems*, International Thomson Publishing, Londen.

van der Helm F.C. (1998), *Test bibstyle*, IK.

Swierstra D. (2001), Combinator parsers: From toys to tools, *in* G. Hutton, ed., *Electronic Notes in Theoretical Computer Science*, Elsevier Science Publishers, vol. 41.
URL `http://math.tulane.edu/ entcs/`

Tenslotte, als men iets wil dat met bovengenoemde methodes niet realiseerbaar is dan kan men zelf een bibliografische stijl schrijven [7]. De BIBTEX stijlen zijn geschreven in een speciale programmeertaal, die werkt op een stack-machine. In een stackmachine staan de waarden waarop berekeningen uitgevoerd worden op een *stack* (stapel), te vergelijken met een stapel borden in een kantine: men pakt altijd het bovenste bord van de stapel en schone borden worden ook weer bovenop geplaatst. Het bovenste element wordt aangegeven door een *stackpointer* (SP). Vergelijkbare programmeertalen zijn Forth en Postscript. In figuur 1 is een voorbeeld van een aftrekking die op de bovenste twee elementen van een stack plaatsvindt. De twee elementen worden verwijderd en vervangen door het resultaat van de aftrekking.



Figuur 1. Aftrekking in een stack-machine

Het programmeren van zo'n stack-machine is niet eenvoudig, zeker niet voor degenen die hier geen ervaring mee hebben. Men moet voortdurend in de gaten houden wat er op de stack staat en maakt hier heel makkelijk fouten mee.

In BIBTEX zitten verschillende ingebouwde operaties, bijvoorbeeld het veranderen van hoofdletters in kleine letters, en het formatteren van namen. Hier enkele voorbeelden:

```
title
"t"
change.case$
```

De operatie `change.case$` verandert de 'case' van de eerste parameter aan de hand van de specificatie in de tweede parameter ("t" = title case (Eerste Letter Hoofdletter), "l" = lowercase, "u" = uppercase).

Het tweede voorbeeld formatteert een aantal namen die op de stack staan gevolgd door het aantal ervan en een specificatie die aangeeft hoe de naam geformatteerd moet worden.

```
name1
name2
...
namei
i
format
format.name$
```

De specificatie bevat letters voor de verschillende onderdelen: 'f' voor de voornaam, 'l' voor de achternaam, 'v' voor het 'van' gedeelte, 'j' voor de achtervoegsels. Een dubbele letter betekent: neem het hele deel, een enkele: gebruik de afkorting. Verder kunnen er interpuncties in de specificatie voorkomen. De volgende specificatie: `"{vv~}{ll}{, jj}{, f}"` zal o.a. de volgende uitvoer genereren:

```
van~Oostrum, P.
```

Merk op dat ontbrekende delen (zoals 'jr.') geen uitvoer genereren. De formattering wordt gebruikt zowel voor de uitvoer als voor het genereren van sorteersleutels.

De taal van de bibliografie

Een probleem met veel bibliografische stijlen is dat ze op het Engels georiënteerd zijn. Dit betekent dat er Engelse woorden direct in de stijl opgenomen zijn. Het betreft hier termen als: 'chapter', 'editor', 'and' e.d. Wanneer men een artikel in een andere taal schrijft wil men deze termen natuurlijk ook in de betreffende taal hebben. We hebben al gezien (pagina 73) dat de stijl `nederlands` dit doet, maar dan zit men wel vast aan deze specifieke stijl. Ook hebben we gezien dat jurabib (pagina 78) ondersteuning heeft voor diverse talen. En verder kan men met `custom-bib` een nederlandstalige stijl op maat maken (pagina 78).

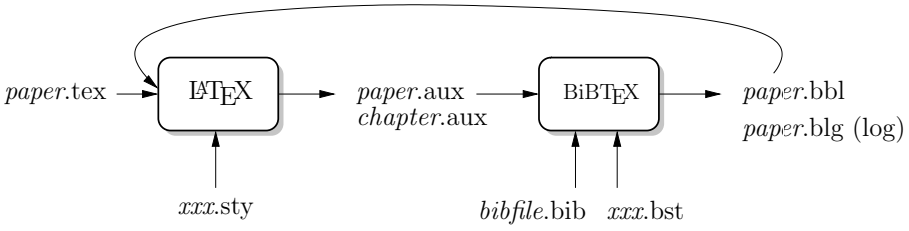
Een andere oplossing is het gebruik van `babelbib` van Harald Harders [8]. `Babelbib` heeft een aantal eigen bibliografische stijlen die afgeleid zijn van de standaard stijlen `abbrv`, `alpha`, `unsrt` en `plain`, en `amsplain` van de $\mathcal{AMS}$. De aangepaste stijlen heten resp. `bababbrv`, `babalpha`, `babplain`, `babunsrt` en `babamspl`. Bovendien zijn er nog `bababbr3` and `babplai3` die in geval van meer dan 3 auteurs de constructie eerste auteur met *et el.* gebruiken, natuurlijk in de juiste taal.

`Babelbib` kan op twee manieren gebruikt worden: Of de hele bibliografie in dezelfde taal; bij verstek is dat de taal die met het `babel` package gekozen is. Of elke bibliografische entry in de taal die in de bibliografische database gegeven wordt. Daartoe is een nieuw veld `language` dat aangeeft in welke taal deze entry gezet moet worden. `Babelbib` lost het taalprobleem op door in zijn bibliografische stijlen de vaste woorden door macro's te vervangen die dan in het package gedefinieerd worden.

In feite is ook `babelbib` maar een gedeelte van de oplossing omdat het alleen met de bijgeleverde stijlen werkt. Een echte oplossing zou zijn als alle bibliografische stijlen de vatse woorden zouden vervangen door macro's. Dan zou er ook een standaard moeten komen voor de namen van deze macro's. Zolang dit nog niet het geval is mote men in het uiterste geval zelf een stijl maken bijvoorbeeld met `custom-bib`.

Het verwerken van de bibliografie

Om de bibliografie in het LaTeX document te krijgen moet men na het LaTeXen BibTeX draaien en daarna weer LaTeX (vaak 2 keer). Zie figuur 2. De operaties zijn:



Figuur 2. Het BibTeX proces

- ☐ `latex paper` ⟶ `paper.aux`
- ☐ `bibtex paper` ⟶ `paper.bbl`
- ☐ `[pdf]latex paper` ⟶ `paper.dvi/pdf`

Hier is een voorbeeld van de output (`.bbl` file) van BibTeX voor het eerste voorbeeld uit dit artikel. De `\newblock` genereert een nieuwe regel als de `openbib` document optie wordt gebruikt.

```
\begin{thebibliography}{1}
```

```
\bibitem{el_libro_latex}
Bernardo Cascales~Salinas, Pascual Lucas~Saor{\'i}n,
  Jos{\'e}~Manuel Mira~Ros, Antonio~Jos{\'e} Pallar{\'e}s~Ruiz, and
  Salvador S{\'a}nchez-Pedre{\~n}o~Guill{\'e}n.
\newblock {\em El Libro de \LaTeX}.
\newblock Pearson, 2003.

\bibitem{swierstra01combinator}
Doaitse Swierstra.
\newblock Combinator parsers: From toys to tools.
\newblock In Graham Hutton, editor, {\em Electronic Notes in Theoretical
  Computer Science}, volume~41. Elsevier Science Publishers, 2001.

\end{thebibliography}
```

De locatie van de files

Waar vindt BibTeX de files die het nodig heeft? Het betreft hier zowel de bibliografische database als de bibliografische stijlen.

De bibliografische stijlen zijn meestal standaard aanwezig en staan dan in standaard plaatsen die bijvoorbeeld voorgeschreven worden door de TDS (TeX Directory Standard). In systemen die op teTeX zijn gebaseerd (ook fptex en TeXlive), is dat meestal `.../texmf.../bibtex/bib` voor de bibliografische databases (`.bib` files) en `.../texmf.../bibtex/bst` voor de stijlen (`.bst` files). Hierbij kunnen er meerdere `texmf` directories zijn, bijvoorbeeld `texmf`, `texmf.local` en een directory voor de individuele gebruiker.

Specifieke bibliografische databases en stijlen die bij één document horen kunnen in dezelfde directory als het document gezet worden.

Wanneer men een eigen verzameling bibliografische databases en stijlen heeft die niet bij één document horen en die men dus graag op een centrale plaats in zijn eigen files wil neerzetten dan kan in systemen die op teTeX gebaseerd zijn de omgevingsvariabele (environment variable) `BIBINPUTS` resp. `BSTINPUTS` gebruikt worden. Deze bevat een rij directories waarin gezocht wordt, gescheiden door `:` op Unix gebaseerde systemen, en door `;` op MS-Windows. Meestal zet men voor- of achteraan een extra `:` of `;` om aan te geven dat daar het standaard pad ingevoegd moet worden, en hoeft men dus alleen zijn eigen pad op te geven. De standaard paden staan overigens in een file met de naam `texmf.cnf` en kunnen eventueel daar aangepast worden.

Bij gebruik van MikTeX is de configuratie iets anders. MikTeX heeft een configuratiebestand `MikTeX.ini` en daarin zit een sectie voor BibTeX.

```
[bibtex]
  Input Dirs=texmf\bibtex
```

Hier wordt verwezen naar een directory waar zowel de bibliografische databases als de stijlen gezocht worden. Door deze parameter aan te passen kan men eigen directories toevoegen. De omgevingsvariabelen `BIBINPUTS` en `BSTINPUTS` werken niet bij BibTeX.

Meerdere bibliografieën

Soms heeft men behoefte aan meerdere bibliografieën in een document. Bijvoorbeeld een bibliografie per hoofdstuk, een bibliografie per onderwerp, of een bibliografie met gerefereerde werken en één met aanvullende literatuur. Er zijn enkele pakketten waarmee dit gedaan kan worden. We beperken ons hierbij tot de op-

somming van deze pakketten, zonder hierbij in detail in te gaan op de werking ervan:

- `chapterbib`
  Bibliografie per hoofdstuk.
- `bibunits`
  Bibliografie per eenheid (hoofdstuk, sectie etc.)
- `multibib`
  Meerdere bibliografieën (voor verschillende onderwerpen)
  Deze is incompatibel met `jurabib`'s `\footcite`
- `bibtopic`
  Verschillende bibliografieën voor verschillende 'topics' waarbij voor ieder 'topic' een andere bibliografische database gebruikt kan worden. Kan met `jurabib` gebruikt worden wanneer men expliciet 'short titles' meegeeft.

In het algemeen zal men bij het gebruik van deze pakketten ook meerdere keren BIBTEX moeten aanroepen. Het schema in figuur 2 zal dan ook navenant uitgebreid moeten worden.

Woordenlijsten met BIBTEX

Als toegift een aardigheidje: Voor het genereren van verklarende woordenlijsten (glossaries) gebruikt men vaak een package als `nomencl` van Boris Veytsman en Bernd Schandl of `GlossTeX` van Volkan Yavuz. Deze werken met `makeindex` als extern programma voor het sorteren. Dit heeft tot gevolg dat men wel voor ieder document en nieuwe lijst met woorden resp. afkortingen moet maken.

Een andere aanpak is het package `gloss` van Javier Bezos. Hierbij stopt men de definities in een bibliografische database, waarbij men dus niet de normale entries heeft, maar speciaal hiervoor aangepaste. De entries heten `@GD` (glossary definition), en de velden zijn: `word` en `definition`.

Meegeleverd met het package zijn enkele 'bibliografische' stijlen om de woordenlijst te formatteren, maar men kan zelf natuurlijk ook nieuwe maken als men BIBTEX programmeren beheerst (`custom-bib` is hier natuurlijk niet bruikbaar).

Tijdens het LATEXen en BIBTEXen worden resp. `paper.tex.gls.aux` en `paper.tex.gls.bbl` files aangemaakt. Hierbij een voorbeeld van het gebruik ervan:

```
\usepackage{gloss}
\makegloss
...
\gloss[options]{gl}
...
\printgloss{database}
```

en een woorden-database:

```
@GD{gl,
  word       = "glossary",
  definition = {A list of definitions of terms}
}
```

Voorbeeld bibliografische database

```
@PREAMBLE{" \def\Lord#1{Lord #1}"}
```

```
@InProceedings{Atk90,
  author =      {M. Atkinson and Bancilhon, F. and De Witt, D.
                 and Dittrich, K. and Maier, D. and {Zdoononnik\-}, S.},
  title =       {The Object-Oriented Database System Manifesto},
  booktitle =   {Proceedings of the Deductive and Object-Oriented
                 Databases Conference -- DOOD'89},
  editor =      {Kim, W. and Nicolas, J.-M. and Nishio, S.},
  year =        {1990},
  address =     {Noord-Holland, Amsterdam},
  pages =       {223--240}
}

@INCOLLECTION{Brunsfeld01,
  author =      {Brunsfeld, S.J. and Sullivan, J. and Soltis, D.E.
                 and Sotis, P.S.},
  editor =      {Silverton, J. and Antonovics, J.},
  year =        2001,
  title =       {Comparative Phylogeography of North-Western North America:
                 A Synthesis.},
  booktitle =   {Integrating Ecology and Evolution in a Spatial Context. The
                 14th Special Symposium of the British Ecological Society.},
  chapter =     15,
  pages =       {319-339},
  publisher =   {British Ecolological Society, Blackwell Science Ltd.},
  institution = {Royal Holloway College, University of London}
}

@Book{Dat95,
  author =      {Date, C. J.},
  title =       {An Introduction to Database Systems},
  publisher =   {Addison-Wesley Publishing Company Inc.},
  year =        {1995},
  volume =      {I},
  address =     {Reading, Massachusetts},
  edition =     6
}

@Book{Del95,
  author =      {Delobel, C. and L{\'e}cluse, C. and Richard, P.},
  title =       {Databases: From Relational to Object-Oriented Systems},
  publisher =   {International Thomson Publishing},
  year =        {1995},
  address =     {Londen}
}

@article{zigic,
  author =      "Kre\v{s}imir {\v{Z}}igi\'{c}",
  title =       "{Intellectual Property Rights Violations and Spillovers in
                 North-South Trade}",
  journal =     "The Test Journal",
  year =        1998,
  volume =      42,
  number =      9,
  pages =       "1779--1799",
  month =       nov
}

@Book{lord,
  author =       { {Lord} Kelvin and Tait, P. G.},
  title =        {The Lord},
```

```
  publisher =    {XYZ},
  year =         2000
}

@Book(lord2,
  author =       {{\Lord{Kelvin}} and Tait, P. G.},
  title =        {The Lord},
  publisher =    {XYZ},
  year =         2000
)

@article{sala,
  author =       "Xavier {Sala-i-Martin}",
  title =        "{Regional Cohesion: Evidence and Theories of
                  Regional Growth and Convergence}",
  journal =      "EEr",
  year =         1996,
  volume =       40,
  number =       6,
  pages =        "1325--1352",
  month =        jun
}

@book{kauf01,
  Year =         {{2000}},
  Title =        {Everyday Strings of English and the Rhetorical Priming of
                  Audience},
  Publisher =    {Lawrence Erlbaum Associates},
  Address =      {Mahwah, New Jersey},
  Author =       {Kaufer, David S. and Ishizaki, Suguru and Butler, Brian S.
                  and Collins, Jeff}
}

@inproceedings{ swierstra01combinator,
  author =       "Doaitse Swierstra",
  title =        "Combinator Parsers: From Toys to Tools",
  booktitle =    "Electronic Notes in Theoretical Computer Science",
  volume =       "41",
  issue =        "1",
  publisher =    "Elsevier Science Publishers",
  editor =       "Graham Hutton",
  year =         "2001",
  url  =         "http://math.tulane.edu/~entcs/"
}

@Book{el_libro_latex,
  author =       {Cascales Salinas, Bernardo and
                  Lucas Saor{\'i}n, Pascual and
                  Mira Ros, Jos{\'e} Manuel and
                  Pallar{\'e}s Ruiz, Antonio Jos{\'e} and
                  S{\'a}nchez-Pedre{\~n}o Guill{\'e}n, Salvador},
  title =        {El Libro de \LaTeX},
  publisher =    {Pearson},
  year =         2003
}
```

Referenties

[1] Patashnik, Oren. *BibTeXing*, februari 1988.
URL CTAN:biblio/bibtex/contrib/doc/btxdoc.pdf

[2] Mittelbach, Frank, Michel Goossens, Johannes Braams, David Carlisle en Chris Rowley. *The LaTeX Companion*. Addison-Wesley, 2e druk, 2004.

[3] Wilson, Peter. *The tocbibind package*. Catholic University of America, februari 2003.
URL CTAN:macros/latex/contrib/tocbibind

[4] Daly, Patrick W. *Natural Sciences Citations and References*, mei 1999.
URL CTAN:macros/latex/contrib/supported/natbib

[5] Berger, Jens en Stefan Ulrich. *The jurabib Package*, september 2002.
URL CTAN:macros/latex/contrib/supported/jurabib/

[6] Daly, Patrick W. en Arthur Ogawa. *Customizing Bibliographic Style Files*, augustus 1999.
URL CTAN:macros/latex/contrib/custom-bib

[7] Patashnik, Oren. *Designing BIBTEX Styles*, februari 1988.
URL CTAN:biblio/bibtex/contrib/doc/btxhak.pdf

[8] Harders, Harald. *Multilingual bibliographies: Using and extending the babelbib package*. *TUGboat*, 2004. Nog te verschijnen.
URL CTAN:biblio/bibtex/contrib/babelbib/tugboat-babelbib.pdf

Piet van Oostrum
Instituut voor Informatica en Informatiekunde
Universiteit van Utrecht
mailto:piet@cs.uu.nl
http://www.cs.uu.nl/~piet/

De TEX flyer: doe er wat mee!

Op de volgende twee bladzijden hebben we onze TEX flyer nog eens afgedrukt. De voorkant beschrijft de sterke punten van TEX, en de achterkant bevat de nodige informatie om mensen een snelle start te geven met TEX.

Als je denkt dat er potentiële TEX-gebruikers in je omgeving bestaan, vraag dan een stapeltje van deze flyers aan bij het bestuur (`ntg@ntg.nl`); we hebben er nog genoeg van. Wellicht kun je ze met deze flyer ze over de streep trekken.

NEDERLANDSTALIGE TEX GEBRUIKERSGROEP

TEX: meer dan tekstverwerken

TEX is een elektronisch zetsysteem. Oorspronkelijk speciaal voor wiskundig zetwerk ontwikkeld door Prof. Donald E. Knuth, wordt TEX nu gebruikt voor vrijwel alle denkbare tekstuele toepassingen: brieven, boeken, handleidingen, database uitvoer, flyers zoals deze, presentaties en interactieve documenten. TEX is gratis en is beschikbaar voor vrijwel alle computer-platforms. Er is een levendige online gebruikersgemeenschap.

TEX werkt met opmaakcodes, net als html/xml. Een voorbeeld van TEX-invoer:

```
\documentclass{article}
\begin{document}
Hello world!\par
$ \phi = \sqrt x $
\end{document}
```

Hello world!
$\phi = \sqrt{x}$

TEX maakt uit deze invoer naar keuze een pdf-bestand of een afdruk op papier.

Met een wysiwyg tekstverwerker bent u waarschijnlijk sneller aan de slag, maar als u mocht besluiten de sprong naar TEX te wagen dan zult u merken dat TEX een praktisch en efficiënt stuk gereedschap is met enorme mogelijkheden.

TEX is betrouwbaar

Technieken die goed werken voor een document van vijf pagina's werken even goed voor een boek van vijfhonderd. De extra inwerktijd verdient u ruimschoots terug omdat TEX gewoon zijn werk blijft doen, hoe lang en complex het dokument ook wordt.

TEX is professioneel

TEX levert kwaliteitszetwerk af, dankzij ondersteuning van kerning en ligaturen, en gebruik van het beste afbreekalgoritme ter wereld. TEX kan goede kwaliteit PostScript en pdf genereren. TEX is dan ook in gebruik bij wetenschappelijke uitgevers.

TEX is aktueel

- Rechtstreekse ondersteuning van pdf, met inbegrip van de interactieve voorzieningen hiervan. Met pdflatex en het pdfscreen-pakket kunt u gemakkelijk presentaties maken. Met ConTEXt is er nog veel meer mogelijk.
- XML/SGML: TEX wordt hier ingezet voor het genereren van pdf- en gedrukte uitvoer. Pakketten zoals xmlTEX en ConTEXt kunnen xml rechtstreeks verwerken.
- Het Omega-project werkt aan ondersteuning voor Unicode.

TEX kan in specialistische behoeften voorzien

Dankzij het open karakter en de programmeerbaarheid van TEX kunnen slimme TEX-gebruikers oplossingen bedenken voor hun specifieke problemen, die ze dan aan de gemeenschap ter beschikking stellen. Hieraan hebben wij het bestaan te danken van BibTEX voor het automatisch genereren van bibliografieën uit een database, en van Makeindex voor het automatisch genereren van registers. Voor niet-westerse talen, bladmuziek, presentaties en diagrammen zijn oplossingen van hoog niveau ontwikkeld voor gebruikers door gebruikers, getuige de voorbeelden op deze pagina.

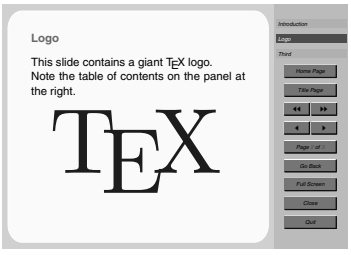
TEX leent zich voor automatische verwerking

In Unix-omgevingen wordt TEX dan ook dankbaar ingezet om automatisch pdf te genereren uit bijvoordeeld DocBook (Linux Documentation Project) of texinfo (het bronformaat voor documentatie voor GNU software).

Herkomst van de figuren. Het arabtex voorbeeld is ontleend aan de ArabTEX documentatie. Het is de aanhef van een traditionele vertelling. Het xypic diagram en het lilypond muziekfragment zijn eveneens ontleend aan de documentatie van de corresponderende pakketten. De MetaPost-figuur is gebaseerd op een bijdrage van Huib van de Stadt.

$$\sum_{i=1}^{qT-1} u_i^2 + \sum_{i=0}^{T-1} \mu_i \left\{ \left[\sum_{j=0}^{q-1} x_j u_{iq+j} \right] - v_i \right\}$$

Wiskunde



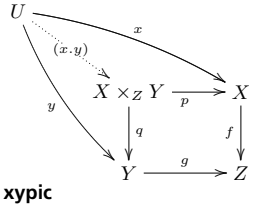
pdfscreen

جُحَا وَحَمِيرُهُ العَشَرَةُ

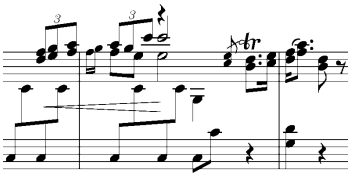
اِشْتَرَى جُحَا عَشَرَةَ حَمِيرٍ. فَرِحَ بِهَا وَسَاقَهَا أَمَامَهُ، ثُمَّ رَكِبَ وَاحِدًا مِنْهَا. وَفِي الطَّرِيقِ عَدَّ حَمِيرَهُ وَهُوَ رَاكِبٌ، فَوَجَدَهَا تِسعَةً. ثُمَّ نَزَلَ وَعَدَّهَا فَرَآهَا عَشَرَةً فَقَالَ:

أَمْشِي وَأَكسِبُ حِمَارًا، أَفضَلُ مِن أَن أَركَبَ وَأَخسَرَ حِمَارًا.

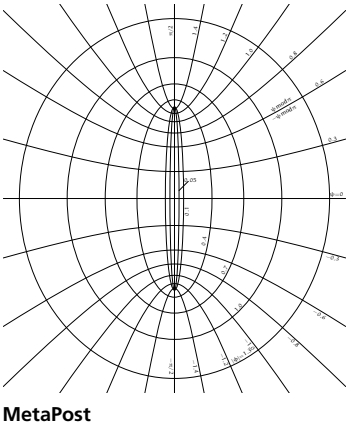
ArabTEX



xypic



Lilypond



MetaPost

Beginnen met TEX

Websites om u nader te oriënteren:

- `www.ntg.nl`, de website van de Nederlandstalige TEX Gebruikersgroep
- `www.tug.org`, de website van de internationale TEX Users Group. Hier vindt u ook een goede 'Getting Started' pagina.
- Speciaal voor Mac gebruikers: `www.esm.psu.edu/mac-tex/`.

TEX op cd

Zowel commerciële als sommige niet-commerciële leveranciers bieden TEX-systemen op cd. Informatie hierover vindt u op hun respektievelijke websites.
De TEX Live cd, die u toegestuurd krijgt als u lid wordt van de NTG, ondersteunt een groot aantal systemen: Windows, Mac OS X, Linux en diverse Unix varianten. U kunt naar keuze TEX vanaf deze cd draaien of op uw harde schijf installeren.
U kunt de TEX Live cd ook bestellen bij bijvoorbeeld Lehmanns Bookshop (`www.lob.de`). Deze boekhandel heeft ook een aantal boeken over TEX en LATEX in zijn catalogus.

TEX software downloaden

U kunt de meeste niet-commerciële en sommige commerciële TEX-systemen ook van het Internet downloaden. In de eerste plaats zijn er de *CTAN* servers. CTAN staat voor 'Comprehensive TEX Archive Network'. Vrijwel alles wat u met betrekking tot TEX zou willen downloaden is hier te vinden. CTAN servers in de buurt zijn:

- `ftp.ntg.nl/pub/tex-archive`
- `ftp.dante.de/pub/tex`
- `ftp.belnet.be/packages/CTAN/`

Al deze servers voeren dezelfde bestanden. De TEX Live cd-images vindt u onder `systems/texlive`.

Voor Linux is echter de meest praktische oplossing om de teTEX-packages uit de distributie te installeren.

LATEX en ConTEXt

Het gebruik van een formaat zoals LATEX neemt auteurs veel werk uit handen. LATEX voegt voorzieningen toe zoals puntenlijsten, automatische inhoudsopgaven en automatisch genummerde hoofdstukken en voetnoten.
Een modern alternatief is ConTEXt, dat uitblinkt in geavanceerde layout, in ondersteuning van de interactieve mogelijkheden van pdf en in integratie van grafische elementen. ConTEXt volgt aktuele ontwikkelingen op de voet. Neem een kijkje op `www.pragma-ade.nl` en abonneer u daar op de ConTEXt discussie-lijst.

De NTG

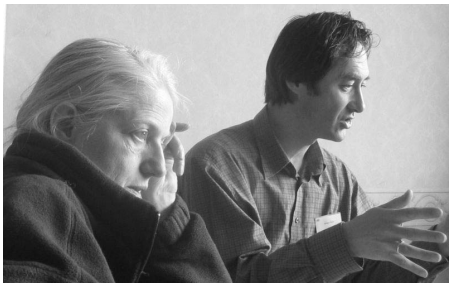
TEX-gebruikers vinden elkaar in TEX gebruikersgroepen; voor Nederland en België is er de NTG, voluit de Nederlandstalige TEX Gebruikersgroep.

De NTG houdt tweemaal per jaar een gebruikersbijeenkomst met interessante lezingen. Het lijfblad van de NTG, de MAPS, verschijnt eveneens tweemaal per jaar, en bevat zowel artikelen voor beginnende gebruikers als informatie over de nieuwste ontwikkelingen. Tevens steunt de NTG internationale initiatieven.

Onze website is `www.ntg.nl`. U kunt zich daar aanmelden als lid, of u abonneren op een aantal mailinglists, of kijken wat er zoal gebeurt op TEX-gebied in binnen- en buitenland.

NTG-dag

27 mei 2004



Foto's:
Frans Goddijn