Verslag

20e NTG-bijeenkomst, 11 november 1997

Frans Goddijn, notulist
fg@fgbbs.iaf.nl

aanwezig

A. Bakker; Phons Bloemen; H.J. Boersma (VU); Luc de Coninck (de Kraal); Wybo Dekker (Servalys); Gilbert van den Dobbelsteen (LOGIN BV); Wietse Dol; W. Dolman (HvA); Erik van Eynde (Kath. Universiteit Leuven); Erik Frambach (RUG); Maarten Gelderman (VU); Peter van Gent (Kluwer); Frans Goddijn (FGBBS); W.J. van de Guchte; Hans Hagen (Pragma); Michael van Hartskamp; Herman Haverkort; R. van der Heijden (Hogeschool van Utrecht); Jan Hellings (UvA/AMC); Peter van Hoeflaken (Thieme); Lico Hoekema (Univ. Maastricht); Taco Hoekwater (Kluwer); René van der Horst (UvA/AMC); Guus Jansen (TUD); Peter de Jong (TUD); Antoine van Kampen (UvA/AMC); Siep Kroonenberg (RUG); M.J. Krugers (Techn. Marketing Cons.); Kees van der Laan; Geert Lobbestael (UvA/AMC); Gerard van Nes (ECN); Piet van Oostrum (UU); Simon Pepping (Elsevier); Mark Potse (UvA/AMC); Ronald Rietman; Adwin Soos (UT); Pilippe Vanoverbeke; Gea Vlak; A. Vrij; Jos Winnink

Opening

Voorzitter Erik Frambach opent de twintigste bijeenkomst en hij dankt onze gastheer Jan Hellings voor de ontvangst in de prachtige zaal met uitstekende voorzieningen.

Vaststellen notulen en agenda

De notulen van de vorige vergadering zien er in de nieuwe MAPS mooi uit. Ze worden, evenals de agenda, ongewijzigd vastgesteld.

Ingekomen stukken en mededelingen

- Er zijn enqueteformulieren verspreid en het bestuur hoopt deze allemaal zo snel mogelijk terug te krijgen.
- Er liggen diverse binnengekomen tijdschriften van zustervereningingen ter inzage, onder andere van Poolse en Tsjechische gebruikersgroepen.
- Er liggen diverse binnengekomen tijdschriften van zustervereningingen ter inzage, onder andere van Poolse en Tjechische gebruikersgroepen.
- De kleurencode voor de badges is zoals gewoonlijk: rood voor bestuur, blauw voor sprekers, geel voor ereleden en wit voor overige deelnemers.

NTG ledenvergadering

Bestuurszaken

Secretaris Secretaris Gerard van Nes zal in het voorjaar van 1998 aftreden na 10 jaar trouwe dienst. Zijn beoogde opvolger, Maarten Gelderman, draait nu al mee met diverse bestuurstaken om de overgang zo soepel mogelijk te laten verlopen.

Werkgroepen Het bestuur stelt faciliteiten beschikbaar voor werkgroepen. De werkgroepen gaan hun eigen gang en kunnen op eigen initiatief activiteiten ontwikkelen en daarvan verslag doen.

Financiële zaken, begroting 1998

Het bestuur licht de begroting toe en beargumenteert dat er momenteel geen reden is voor contributieverhoging.

NTG-bijeenkomsten 1998

In dit memorabele jaar zal de NTG 10 jaar bestaan. Tijdens de voorjaarsbijeenkomst (in Arnhem op 11 juni) wordt dit tweede lustrum gevierd. Deze bijeenkomst zal een groter deel van de dag beslaan dan gewoonlijk, met de opening en de ledenvergadering op een aanmerkelijk vroeger tijdstip in de ochtend en een lopend buffet tot slot. Er moet nog veel werk ter voorbereiding worden verricht. Het breed opgezette thema thema zal *typografie* worden. Mensen die een bijdrage willen leveren, in de vorm van een lezing (15 minuten tot een uur) of artikel, zijn van harte welkom. De MAPS zal ten dele in kleur worden uitgegeven.

Miscellaneous

Verslag werkgroepen

SGML De werkroep SGML meldt bij monde van Taco Hoekwater dat er, behalve zijn eigen werk binnen SGML, geen werkroepactiviteiten (meer) zijn geweest.

Spelling Erik Frambach van de werkgroep Spelling is bezig met de conversie naar Ispell, Piet Tutelaers onderhoudt de ftp-site van de werkgroep maar overigens zijn de activiteiten van de werkroep marginaal. Een door velen gevraagde conversie van onze woordenlijsten naar WP en Word blijft moeilijk. Gert Hardeman heeft een WP woordenlijst ter beschikking gesteld.

Taco Hoekwater meldt dat de woordenlijstconversie naar Ispell bij hem lokaal goed draait.
De heer M.J. Krugers merkt op dat de geringe penetratie van “onze” woordenlijst in de buitenwereld ook het gevolg is van het gegeven dat wij de lijsten zonder marketingstrategie gewoon vrijgeven. Normaliter verkopen ondernemers zulke woordenlijsten voor een dollar per woord. Laten we hier kansen liggen? Het Amerikaanse bedrijf Proximity koopt en verkoopt woordenlijsten als de onze.
Jos Winnink stelt voor dat we een “keurmerk” van het Genootschap Onze Taal vragen voor onze woordenlijst. Dit zou de verkoopbaarheid ervan sterk kunnen verbeteren. Actie Gerard van Nes / Frans Goddijn?

TUG/LUG aangelegenheden

Een verslag van de bijeenkomst in San Francisco is verschenen in de MAPS.
De volgende TUG meeting zal in Polen worden gehouden (data zie komende MAPS)
De nieuwe EuroTeX vindt eind mei plaats in St. Malo (data zie komende MAPS)

Communicatiezaken

Piet van Oostrum meldt dat de server van de NTG door een snellere machine wordt vervangen. Via ftp is er 95 MB aan ConTeXt bijgekomen, ook via www bereikbaar. Het CPAN archief (voor PERL) wordt op onze server vanzelf gemirrored.
Frans Goddijn vertelt dat FGBBS in het nieuwe jaar over zal gaan op een ISDN verbinding. Een deel van de ConTeXt files is al op het BBS beschikbaar.

MAPS aangelegenheden

De MAPS vormgevers willen nog een aantal kleinigheden verbeteren in de komende uitgave. Reacties en suggesties worden verwelkomd.
De komende MAPS krijgt *typografie* als thema. In dit dubbeldikke nummer en de komende nummers zullen, zo is het streven van de redactie, ook kortere bijdragen worden geplaatst, columns, tips, bijdragen uit de “toolbox”-serie, informatie over (installatie van) font families en dergelijke.
De MAPS redactie dankt Siep Kroonenberg voor haar nieuwe ontwerp.
De MAPS award voor de beste bijdrage gaat ditmaal naar de man die intussen ruim 6000 MAPS exemplaren heeft verzonden, duizenden kilo’s voor de NTG heeft versleept: Gerard van Nes!

4allTeX

Erik Frambach vertelt dat er eindelijk een overeenkomst is getekend tussen het 4TEX team en Addison-Wesley. In de zomer van 1998 komt er een 4TEX boek uit bij deze uitgeverij, met daarbij de CD’s.
De ultieme NTG-versie, de vierde editie van 4allTeX zoals we die nu kennen, wordt binnen enkele weken geproduceerd. Aan het DOS-systeem ervan zijn slechts kleine aanpassingen gemaakt. Als extra’s zijn er een Windows editor en een GhostScript viewer te verwachten. Ter vergadering liggen er twee gold-cd’s met betaversies voor wie wil testen. Alle betreffende documentatie komt ook op de CD’s te staan, in PDF, TeX, DVI en PostScript formaat.
De nieuwe TeX versie die vanaf zomer ’98 zal worden gebruikt is web2c, en ook wordt gebruik gemaakt van het TDS systeem voor directories. Er is in principe geen bezwaar om ook standaard eTeX aan te bieden, zoals nu ook standaard de hyperversie van dvips wordt gebruikt.
Bij Addison-Wesley zal alleen een Windows-32 bits versie van 4allTeX verschijnen. Versie 4 wordt daarmee de laatste die geschikt is voor DOS gebruikers (al kan ook de DOS versie van web2c overweg met de TDS structuur).
emTeX heeft helaas zijn tijd gehad, nu Eberhard Mattes andere prioriteiten heeft het vervaardigen van een nieuwe emTeX-versie.

Wat verder ter tafel komt & Rondvraag

De heer A. Bakker vraagt zich af of er geen standaardisatiecommissie moet komen om de vele uitbreidingen en vertakkingen van TeX versies te stroomlijnen. Inderdaad, hier is geen leiding, maar plain TeX is bevroren voor wie geen verandering wil meemaken. Kees van der Laan noemt wiskundigen als voorbeeld van TeX users die liever stabiliteit hebben dan voortdurende verandering. Hij ziet als wiskundige met lede ogen aan dat allerlei “machtsblokken” binnen de TeX wereld hun eigen invalshoek onder de aandacht proberen te brengen. Hans Hagen stelt hier tegenover dat TeX wel in vernieuwde versies bepaalde typografische beperkingen moet oplossen, aangezien TeX anders een stille dood sterft terwijl andere pakketten wel deze oplossingen bieden voor wie typografisch bezig is. Met name voor uitgevers zou TeX dan een “dead end” zijn.
Tijdens komende bijeenkomsten willen we meer aandacht voor deze kwesties vragen.
De heer M.J. Krugers meldt dat er een nieuwe versie is van Scientific Workplace. De specificaties liggen ter inzage op de leestafel. Deze nieuwe versie is WEB-compatible. Er komt een 30-dagen trial CDrom in de MAPS van 1998. Deze kan ook direct bij de heer Kruger worden aangevraagd.
Taco Hoekwater meldt een bug in dvips. Deze veroorzaakt een melding als “premature eof in binary...”

Volgende bijeenkomst De leden worden uitgenodigd op de volgende vergadering, op donderdag 11 juni bij de Hogeschool Arnhem Nijmegen, te Arnhem. Thema van deze feestelijke bijeenkomst wordt “typografie”.

Van de voorzitter

Erik Frambach
Rijksuniversiteit Groningen
E.H.M.Frambach@eco.rug.nl

Elk jaar sta ik er weer versteld van hoeveel initiatieven er in de TEX-wereld ontplooid worden om alles nog beter, nog mooier en nog krachtiger te maken. Dit is des te opmerkelijker omdat bijna al dat werk op niet-commerciële basis wordt gedaan. En dat in een tijd waarin niemand tijd heeft...

Het afgelopen half jaar is daarop geen uitzondering. Laten we daarom eens wat wapenfeiten op een rij zetten.

...PDF...

PDF krijgt een steeds prominentere rol in de TEX-wereld. PDFTEX wordt nog steeds verder ontwikkeld en kan inmiddels behoorlijk volwassen worden genoemd. Knap werk! Parallel daaraan wordt DVIPDF ontwikkeld. Dat programma genereert PDF uitgaande van een "gewone" DVI-file, terwijl PDFTEX de stap via DVI helemaal overslaat. Beide methoden hebben hun voor- en nadelen maar toch denk ik dat uiteindelijk alleen PDFTEX overblijft. De toekomst zal het leren.

Nog een andere "gratis" manier om PDF te genereren is via het freeware programma Ghostscript. De nieuwste versies kunnen PDF zowel lezen als schrijven, zodat Adobe's Acrobat Distiller (commercieel) en Reader (freeware) bijna overbodig worden. In sommige opzichten is Ghostscript zelfs superieur aan Adobe's producten...

Web2c

Op het gebied van TEX-implementaties is ook goed nieuws te melden, met name voor MS-DOS en MS-Windows-gebruikers. Voor die laatsten zijn er tegenwoordig twee goede complete implementaties beschikbaar: MikTEX en Web2c. Vooral Web2c is van groot belang omdat die implementatie thans draait op Unix, MS-DOS, MS-Windows en Amiga. Die uniformiteit is in ieders belang: het voorkomt incompatibiliteiten en maakt het mogelijk werk te spreiden. Immers er zijn nu veel meer gebruikers die kunnen meedenken/werken, en het wiel hoeft niet telkens opnieuw te worden uitgevonden.

TEX Live 3

De nieuwe TEX Live 3 cdrom laat al een voorproefje zien van wat multi-platform-ondersteuning kan betekenen. 'Out of the box' kan daarmee TEX worden gedraaid op een keur aan Unix-systemen, Windows 95 en Windows NT. Het is daarom niet verwonderlijk dat de gebruikersgroepen TUG (internationaal), UKTUG (Engeland), Dante (Duitsland), CSTUG (Tsjechië en Slowakije) en natuurlijk de NTG deze cdrom naar al hun leden sturen.

XDVI

De MS-Windows-versie van Web2c bevat als klap op de vuurpijl een gloednieuwe DVI-viewer. Deze is gebaseerd op XDVI zoals die op Unix-systemen bekend is, en heeft ongeveer dezelfde functionaliteit. De user-interface is echter veel moderner. Uiteraard werkt deze viewer naadloos samen met de andere Web2c-programma's zodat bv. font-generatie geheel automatisch werkt, zowel voor Metafont- als PostScript-fonts.

Omega

Op meer fundamenteel niveau wordt ook aan de weg getimmerd. Omega, de 16-bits (Unicode) versie van TEX, schijnt nu ook op lichte machines (slechts enkele megabytes geheugen) goed te draaien. Veel font-encoding en input-encoding-problemen zoals die onder meer op de EuroTEX-conferentie naar voren kwamen kunnen door Omega moeiteloos worden opgelost. Maar de TEX-wereld schijnt nog wat huiverig te zijn om de overstap naar Omega te maken.

NTS

Intussen zit ook de NTS-groep (New Typesetting System) niet stil. Besloten is dit jaar daadwerkelijk met de implementatie te beginnen. NTS zal in Java worden geschreven. Daarmee zou platform-onafhankelijkheid al bij voorbaat ingebouwd zijn, en dat is een goede zaak. Laten we wel hopen dat het NTS in Java beter zal vergaan dan bv. de Corel Office Suite, die jammerlijk is gefaald door veel te slechte performance.

ϵ-TEX

Een ontwikkeling die wat dichter bij de oorsprong blijft is ϵ-TEX. De nieuwste versie daarvan voegt functionaliteit toe aan TEX die niet alleen voor programmeurs interessant is. Maar ook hier geldt dat het moeilijk is om een nieuwe standaard breed geaccepteerd te krijgen. Wellicht lukt dat pas als grote macropakketten zoals LATEX of ConTEXt er serieus gebruik van gaan maken. Dan volgt wellicht vanzelf een (grote) groep gebruikers.

ConTEXt

Het macropakket ConTEXt kan ook zonder meer beschouwd worden als een belangrijke nieuwe ontwikkeling. Mede dankzij indrukwekkende demonstraties begint het pakket internationaal naam te maken. Het is wellicht de enige serieuze concurrent voor LATEX. ConTEXt zou best eens heel wat LATEX-gebruikers kunnen doen overstappen, zeker zolang er geen enkel zicht is op LATEX 3.

Xindy

Voor wie regelmatig indexen maakt met behulp van het programma MakeIndex is er ook goed nieuws. Binnenkort komt het programma Xindy uit. Dit is een compleet nieuw programma waarin veel van de tekortkomingen van MakeIndex zijn verholpen. Het omslachtige coderen van sorteersleutels bv. kan nu systematisch worden opgelost, en ook multi-linguale en multipele (sub-) indexen zijn zonder gegoochel te realiseren. Overigens is Xindy net zo min als MakeIndex gebonden aan TEX, maar door hun manier van werken zijn Xindy & TEX wel ideale partners. MS-Word-gebruikers zie ik niet zo gauw Xindy gebruiken, hoewel dat zonder meer mogelijk is.

4allTEX

Het 4allTEX-team is inmiddels met de 4e editie van hun dubbelcdrom en handleiding uitgekomen. De nieuwste versie integreert nu veel beter met MS-Windows, bevat een aantal nieuwe mogelijkheden en draait daarbij sneller dan voorheen. Ook voor deze cdroms is internationaal veel belangstelling: TUG, UKTUG, Dante en NTG sturen de cdroms naar al hun leden. En dat is nog niet alles. Een echte MS-Windows-versie is intussen in beta-test en zal over ongeveer een half jaar beschikbaar zijn.

CTAN

Over cdroms gesproken. Dante gaat wederom het hele CTAN (Comprehensive TEX Archive Network) archief uitbrengen op cdrom. Dat is vooral erg handig voor wie geen (snelle) Internet-verbinding heeft maar toch regelmatig spullen van CTAN nodig heeft. Ook deze cdroms (3 stuks) worden door verschillende gebruikersgroepen (waaronder natuurlijk de NTG) aan alle leden toegestuurd.

TUG*India*

Zo af en toe zien we gebruikersgroepen komen en gaan. Recentelijk is er in India een nieuwe gebruikersgroep opgericht: TUG*India*. Ik hoop van harte dat zij de TEX-wereld nieuwe impulsen kunnen geven. Wat meer niet-westerse inbreng zou heel verfrissend kunnen zijn.

NTG

Dat de NTG dit jaar 10 jaar bestaat is natuurlijk ook een mijlpaal. Verderop in deze MAPS gaan we daar dieper op in, maar op deze plek wil ik alvast iedereen bedanken voor al het moois dat uit onze gelederen naar voren is gekomen in de afgelopen 10 jaar. Kijk bv. eens naar deze extra dikke en extra mooie MAPS. Ook de bijeenkomst waarbij we ons 10-jarig bestaan officieel vieren krijgt een extra feestelijk tintje door bijzondere sprekers, een borrel na afloop, aansluitend een smakelijke maaltijd, en een boeiende cursus de volgende dag. Dat deze dingen gerealiseerd kunnen worden geeft eens te meer aan met hoeveel enthousiasme en inzet er binnen de NTG gewerkt wordt. Hulde aan iedereen die hieraan heeft bijgedragen!

Mimi Jett
mimi@teleport.com

From the TUG President

At the 10th EuroTEX meeting in St. Malo, I apparently offended some purists by concluding that the common bond we share is 'typesetting mathematics'. Ooops! Of course, TEX does more than beautifully display equations; create complex page layouts; organize numbering, referencing, and indexing schemes; and provide a venue for dozens of languages. We all know the power of this program. But what amazed me early on, and continues to excite me today is the effect TEX has had on the world's scientific community.

Science fiction and New Age religion have published numerous accounts of a future time when all mankind will share in the wealth and comfort of an advanced civilization. Fictitous stories about the power of technology to eliminate hunger, strife, even war, abound in popular media. But where is this utopia? How will a faster computer chip, an advanced operating system, or wider bandwidth solve the problems of humanity? They can't, and won't. What Dr. Knuth did for humanity was give us the tools, and the opportunity to communicate *and* create the technology that will change the world. But wait. It is not the technology, but the people that will advance society into someplace nice to live. With his original philanthropy, he laid the groundwork for the global community we now enjoy.

EuroTEX was a fabulous conference, especially in light of what is happening around us. Not only did we see ingenious advances in the use and display of information using TEX, we were given the source code so we can do it ourselves! This is incredible in today's world of bottom-line focus. But in the TEX community, it is just the way it is. Many of the ideas presented could have been transformed into potentially profitable commercial products. But not one TEXie withheld the tools they had created. It is the norm for the people of our community to share their ideas, hopes, and macros. Let me be the first to admit that I did not understand this for several years. As a business owner, I held tightly onto the developments we had. Many times we were asked for our macros and routines, but fear of eroding some competitive edge limited our thinking. It was only through the openness and willingness of the TEX Users Group that I began to see what honest collaboration can do. And after nearly 10 years of membership and contribution, it is all becoming quite clear.

One tremendous example is the gift of 4allTEX to the TUG community by NTG. This is certainly a sought-after program, and the availability at a nominal cost allows us to distribute it to all of our members. In addition to the efforts of your group to make it available, the German group Dante has generously offered to fund the costs of manufacturing. Wow! Dante has been incredibly helpful over the years, but the recent actions of Joachim Lamarsch, Marion Neubauer and the entire Dante organization have left me once again smiling at the spirit in TUG. This could be a case study in people-helping-people.

There must be other organizations like TUG, but I have never experienced another. To see collaborators from dozens of countries working together for the common good, without borders or boundaries, is to understand what is possible.

Redactioneel

Siep Kroonenberg
Taco Hoekwater

Deze nieuwe MAPS is een jubileumnummer ter ere van het tienjarig bestaan van het NTG. Dat hebben we geweten ook. De MAPS is namelijk extra dik, en zelfs voor een deel in kleur (als u de volgorde hier en daar een beetje vreemd vindt: we hebben geprobeerd de stukken met kleur in een relatief klein aantal katernen te verzamelen; vandaar.).

Het was weer een zware bevalling: we zien nu op de klok maandag 11 mei, 17.45 's middags, en hij/zij is eindelijk 'af'. Natuurlijk leverde er weer vrijwel niemand zijn artikelen op tijd in (we gaan natuurlijk niet toegeven dat we de hoeveelheid werk wellicht een ietsiepietsie onderschat hadden), zodat het de afgelopen week een vreselijk gestress was om alles in orde te krijgen. En nu maar hopen dat straks op tijd wordt geleverd door de drukker. Het laatste artikel dat ook inderdaad nog is opgenomen verscheen vanochtend om 12 uur, en het is nu mooi geweest.

Niet helemaal de beloofde 400 pagina's, maar toch een heel eind in de goede richting, en de artikelen beslaan vrijwel het hele vakgebied rond TEX. De combinatie van nogal kleine lettertjes en de deze keer wel heel erg interessante inhoud zou garant moeten staan voor minstens enige zomeravonden leesplezier.

Diverse artikelen blikken terug op tien jaar TEX in Nederland en België (dat was natuurlijk te verwachten). Onder andere hebben we een reprint van de allereerste MAPS (in de 'originele' kwaliteit); een overzicht van de geschiedenis van de NTG en de 'uitslag' van de enquete van vorig jaar.

Ook een paar andere artikelen gaan over de geschiedenis, zoals het verhaal van Gerrit Oomen over hoe de wereld er vóór TEX uitzag, en Taco's artikel over oude boeken en tijdschriften. Als een link met het heden vinden we een een aantal recensies van programma's en verslagen van 'werk in wording'.

En dan hebben we nog diverse onderwerpen zoals fonts, het zetten van arabisch, gekleurde letters, oude cijfers en chemische formules. Sommige artikelen zijn ernstig van toon, andere artikelen zijn grappig. Sommige artikelen zijn moeilijk, andere artikelen zijn eenvoudig te begrijpen door iedereen. Sommige artikelen zijn (erg) kort, andere artikelen zijn (erg) lang.

Het is een cliché om te zeggen dat er 'teveel om op te noemen' in deze MAPS staat, maar het is wel te veel om overal uitgebreid op in te gaan en toch deze redactioneel nog interessant te houden. We wensen iedereen net als anders weer heel veel leesplezier en hopelijk tot ziens op 11 juni (we hopen nog vuriger dat u dit kunt lezen vóór die datum).

Productie

conTEXt

Een groot deel van deze MAPS is gezet met behulp van Hans Hagen's conTEXt formaat. LATEX heeft als voornaamste verdienste dat het een grondig ingeburgerde standaard is, maar LATEX is een vreselijk formaat om een goede opmaak in te implementeren. Met conTEXt gaat dit veel beter: hierin is het bijvoorbeeld mogelijk om zonder handmatig ingrijpen de regels van beide kolommen met elkaar te laten lijnen. Wij wensen conTEXt dan ook een zonnige toekomst.

Kleur

Enkele katernen zijn in kleur uitgevoerd. Er is een experimentele mogelijkheid om vanuit dvips kleurenseparaties te maken, maar in overleg met de drukker kozen wij voor een andere route: wij converteren de PostScript uitvoer naar pdf en laten de drukker de pdf-bestanden separeren. Hiertoe gebruiken zij Adobe Acrobat Exchange met de *Crackerjack* plug-in. Wij houden ons hart vast, want dit betekent een heel traject in het productieproces waar wij geen controle over hebben. Aan de andere kant is het de drukker en niet de MAPS-redactie die verstand heeft van kleurseparaties. Ervaringen van Siep met pdf (maar niet met Crackerjack) in commercieel drukwerk waren niet geruststellend, maar we zullen zien.

Woorden van dank

De redaktie bedankt deze keer met name Erik Frambach en Hans Hagen voor de indrukwekkende hoeveelheid werk die zij verricht hebben. Artikelen schrijven, én achter auteurs aan jakkeren, én proeflezen, én met de layout helpen, én allerlei gegevens controleren, terwijl beide personen ook nog een 'gewoon' leven hebben (en let wel: geen redaktielid zijn!) is een prestatie van formaat. Ook veel dank aan Maarten Gelderman die ons voor een heleboel ernstige taalvautten heeft behoedt (volgende keer sturen we de héle MAPS naar hem op). En natuurlijk ook alle andere mensen die geholpen hebben hierbij van harte bedankt!

Veel leesplezier!

Het Weten Waard

TEX kalender 1998

17–21 augustus 1998 TUG '98 Torun, Polen

Gebruikersgroepen

Hier een completere lijst van LUGs:

AsTEX (French-speaking)
CSTUG (Czech and Slovak Republics)
CyrTUG (Russia)
Dante (German-speaking)
Estonian User Group
Greek TEX Friends Group (Greek speaking)
GUST (Poland)
GUTenberg (French-speaking)
GUTH (Grupo de Usuarios de TEX Hispanoparlantes)
ITALIC (Irish)
iTEXnici (Italian)
JTUG (Japan)
Lithuanian TEX Users Group
Nordic TEX Users Group
NTG (Dutch-speaking)
TeXCeH (Slovenian TEX User Group)
Tirant lo TEX(Grup d'usuaris catalanoparlants)
TUG (International user group)
TUGIndia (Indian)
UK TUG (United Kingdom)

Bulletins/journals

Baskerville (UKTUG)
Cahiers GUTenberg (GUTenberg)
Zpravodaj (CSTUG)
TeXnische Komödie (DANTE)
GUST bulletin (GUST)
TTN (TEX and TUG News; TUG)
TUGboat (TUG)
TUGIndia Journal (TUGindia)
MAPS (Minutes and
APpendiceS; NTG)

Diversen

CTAN Comprehensive TEX Archive Network; sites waar men 'anonymous ftp' kan gebruiken om TEX/LATEX-achtig materiaal te verkrijgen. CTAN is de 'home' voor de officiële versie van LATEX etc. CTAN sites zijn: `ftp.dante.de`, `ftp.tex.ac.uk` en `ftp.cs.tug.org`
FGBBS NTG's Bulletin Board
AllTEX TEX, LATEX, etcTEX
ltxiii LATEX 3.0
4TEX Het volledige TEX *runtime systeem* voor MS-DOS PC systeem, gebaseerd op emTEX en 4DOS.
4allTEX De 4TEX applicatie *plus* alle mogelijke gerelateerde files en utilities, gedistribueerd op de diskette set en CD.
AMS American Mathematical Society
SGML Standard Generalized Markup Language

NTG's TEX Bulletin Board Systeem

Op het TEX Bulletin Board van de Nederlandstalige TEX Gebruikersgroep (FGBBS) is een zo volledig en actueel mogelijke TEX, emTEX, LATEX, TEX-NL en MusicTEX collectie beschikbaar voor alle bezitters van een modem. Het BBS is kosteloos toegankelijk voor iedereen en er zijn geen beperkingen aan de hoeveelheid bestanden die kunnen worden opgevraagd. Het systeem is aangesloten op een modem die zowel ISDN als lagere snelheden aankan.

De beheerders zijn Frans Goddijn en Henk de Haan. FGBBS is te bellen op 026-3217041.

NTG's winkel

Via de NTG is beschikbaar:

4allTEX *dubbel* cd

Ruim 1.2 Gigabyte (bijna 50.000 files) aan onder meer 4TEX utilities, fonts, TEX/LATEX/METAFONT/etc documentatie, de volledige MAPS 1 t/m 17, discussielijsten TEX-NL, TEX-HAX, UKTEX van de afgelopen 6 jaren, etc. etc.
Kosten inclusief het uitgebreide 4TEX manual *f* 60,–. Kosten cdroms apart: *f* 30,– Kosten manual apart *f* 30,–

TEXlive cd

Kant-en-klaar TEX systeem voor Linux, DEC, HP, IBM, NeXT, SGI, en SUN Unix systemen, alsook voor win32.
Door Thomas Esser en Sebastian Rahtz, gebaseerd op

Web2C van Karl Berry.
Kosten inclusief manual f 35,– (f 60,– voor niet-leden).

Bestellingen kunnen gedaan worden door overmaking van het verschuldigd bedrag op de postgiro van NTG (1306238) t.n.v. penningmeester NTG, Deil, met vermelding van hetgeen gewenst is.

NTG/TUG lidmaatschap

Het blijkt soms dat nieuwe NTG/TUG leden na ongeveer een half jaar nog geen TUGboat of TTN van TUG hebben ontvangen. Ondanks dat men een TUG lidmaatschap via NTG aanvraagt, blijkt in bijna alle gevallen de administratie- en verzendproblemen bij TUG zelf te liggen. Mocht na enige maanden tijd geen post van TUG ontvangen worden, dan worden de betreffende NTG/TUG leden dringend verzocht om kontakt op te nemen met het secretariaat van de NTG.

De NTG najaarsbijeenkomst

De NTG najaarsbijeenkomst 1998 zal op donderdag 22 oktober plaatsvinden aan de Katholieke Universiteit van Leuven. We verwachten met name aandacht te besteden aan fonts.

MAPS 98.2

Sluitingsdata voor het inleveren van artikelen, bijlagen, en/of mededelingen voor de volgende MAPS uitgaven zijn:

*1 oktober '98 (*MAPS *98.2; #21)*
*1 april '99 (*MAPS *99.1; #22)*

Aanleveren kopij voor de komende MAPS:

- *Bij voorkeur* in gebruikmakend van de LaTeX 2_ε class file `maps.cls` of de ConTeXt module `m-map-01`. Beide files zijn via de redactie te verkrijgen en beschikbaar op de TEX-NL fileserver, archive.cs.ruu.nl (ftp-site) en FGBBS (026-3217041).
- Daarnaast kunnen bijdragen ingestuurd worden gemaakt met `ltugboat.sty` of `article.sty` / `report.sty`.
- Verder zijn bijdragen vanzelfsprekend ook welkom in *plain*-TeX of ongeformatteerd.
- Plaatjes bij voorkeur als (Encapsulated) PostScript file plus het oorspronkelijke formaat; dit laatste om eventuele problemen beter te kunnen oplossen.

Daar MAPS *bijdragen in* plain *TeX altijd worden omgezet naar LaTeX of ConTeXt, verdient vanzelfsprekend aanbieding van materiaal in een van deze formaten de voorkeur.*

Eventuele nadere richtlijnen voor auteurs zijn op te vragen bij de redactie.

Bijdragen kunnen gestuurd worden naar:

Taco Hoekwater,
Singel 191
3311 PD Dordrecht
Email: `taco.hoekwater@wkap.nl`

Personen met een modem maar zonder internetaansluiting kunnen hun bijdrage ook via modem/PTT lijn naar de redactie sturen. Gaarne hiervoor eerst kontakt opnemen met Taco Hoekwater, tel. 078–6137806.

Financieel verslag 1997

Wybo Dekker
Penningmeester NTG

Inleiding

Voor u ligt het financieel jaarverslag van de Nederlandstalige TEX Gebruikersgroep over het jaar 1997. De financiële ontwikkeling van de vereniging wordt aan de hand van de staat van baten en lasten en de balans besproken. Beide zijn in de tabel opgenomen, samen met respectievelijk de begroting en de overeenkomstige gegevens van een jaar geleden om zo vergelijking mogelijk te maken. De balansgegevens van vorig jaar, die toen per 12 januari werden vermeld, zijn gecorrigeerd naar 1 januari.

De administratieve posten in de tabel van baten en lasten zijn anders benoemd en ingedeeld dan tot nog toe gebruikelijk. Door middel van liniëring is geprobeerd oude en nieuwe posten te groeperen.

Baten

Contibuties NTG+TUG

De contributies van de 300 leden zijn als volgt geïnd:

instituten	32 x	245 =	7840
instituten (1996)	1 x	245 =	245
instituten (1998)	1 x	245 =	245
aanvulling	4 x	65 =	260
leden	130 x	90 =	11700
leden (1995)	1 x	90 =	90
leden (1996)	2 x	90 =	180
leden (1998)	2 x	90 =	180
combileden	60 x	81 =	4860
combileden (1998)	1 x	81 =	81
studenten	10 x	60 =	600
studenten (1996)	1 x	60 =	60
studenten (1998)	1 x	60 =	60
teveel betaald	x	=	50
Subtotaal NTG			26451
TUG-leden	8 x	105 =	840
TUG-combileden	70 x	95 =	6650
TUG-combileden (1998)	1 x	95 =	95
TUG-studenten	2 x	70 =	140
TUG-studenten (1996)	1 x	35 =	35
Subtotaal TUG			7760
Totaal contributies			34211

Rente

De rente is hoger dan begroot doordat de in vorige jaren apart gehouden TUG contributies naar de NTG-rekening werden teruggeboekt.

Winstaandeel 4allTeX-cdrom

Per 31-12-97 (een datum waarop de derde editie goeddeels was uitverkocht en de verkoop van de vierde nog moest beginnen) is in overleg de winst over de verkoop van de 4allTeX-cdrom gelijk gesteld aan het banksaldo (*f* 36795) van het 4allTeX team. Uit deze winst is *f* 11795 gereserveerd voor investeringen in de vierde editie. Volgens afspraak komt de helft van de winst toe aan de NTG. Vandaar dat U hier een post van *f* 12500 vindt. Deze post heeft betrekking op de verkopen in 1995, 1996 en 1997, en heeft in vorige jaren als Pro Memorie-post gefigureerd.

Kleine verkopen

Dit betreft de verkoop van TeXLive-cdroms en andere kleine materialen.

Lasten

Kleine aankopen

Voor de MAPS-redactie werden een zipdrive en het Frutiger font aangeschaft. Verder werd een recensie-exemplaar van de “Web Builder” aangekocht.

Afschrijvingen

Deze post betreft een laserprinter die nu is afgeschreven.

Subsidies

Het FGBBS werd gesponsord met *f* 1000. Tevens werd eenzelfde in 1996 toegekende, maar toen niet uitbetaalde, subsidie geëffectueerd. De andere begrote *f* 1000 was bestemd als bijdrage voor het TeX Bursary Fund. Het bedrag werd weliswaar naar TUG overgemaakt maar kwam onbestelbaar retour na aftrek van bankkosten. Het bedrag staat nu daarom nog bij de crediteuren, zie hierna.

Zaalhuur
Betreft een bestuursvergadering.

Kantoorbehoeften
(Vroeger geadministreerd als administratie/ bestuurskosten.) Betreft aanschaf van papier, enveloppen, magnetische media, acceptgiroformulieren en dergelijke.

Representatiekosten
Betreffen kosten van lunches en diners tijdens bestuur-, redactie- en werkgroepvergaderingen en kosten van diners voor sprekers na de tweejaarlijkse bijeenkomsten.

Reis- en verblijfkosten
Betreffen reiskosten naar de zojuist genoemde vergaderingen.

Porti
Betreft voornamelijk porti voor verzending van MAPS, uitnodigingen en facturen.

Drukkosten
Betreffen voornamelijk de aanmaak van MAPS. Werden vroeger samen met porti begroot onder de noemer "Nieuwsbrief/verslagen". De kosten kwamen hoger uit dan de begroting door de groei van het aantal leden die ook in de contributies is terug te vinden.

Bijdragen
Hebben alle te maken met de Kamer van Koophandel: registratiekosten, deelname aan aldaar georganiseerde seminars, inschrijvingskosten et cetera.

Saldo
Dit jaar een hoog positief saldo, dat voornamelijk is veroorzaakt door het winstaandeel in de 4allTeX-cdrom verkoop. Gezien het feit dat deze winst over 1995, 1996 en 1997 is gecumuleerd is die geen reden om ons rijk te rekenen.

Crediteuren
Per 1-1-1998 bestond de post crediteuren uit:

- ƒ 8536: TUG-contributes 1995/1996; eigendom TUG; stond begin '97 nog op aparte rekening, maar werd in 1997 naar de NTG-giro overgemaakt. Wordt in 1998 naar TUG overgemaakt.
- ƒ 7760: TUG-contributies 1997; dito.
- ƒ 1000: tegoed TeX-bursary fund 1997; wordt binnenkort overgemaakt.
- ƒ 6972: betalingen 4TeX 1997; eigendom 4allTeX-team.
- ƒ 325: nog te betalen rekeningen
- ƒ 661: vooruitbetaalde contributies '98.

Debiteuren
Betreft twee nog te betalen rekeningen en het nog te ontvangen winstaandeel over de 4allTeX-cdrom-verkopen 1995-1997.

Baten en lasten over 1997

(bedragen in guldens)

	Begroting		Realisatie	
	Credit	Debet	Credit	Debet
Contributies NTG+TUG	23000		34211	
Afdracht TUG 1997				7760
Rente	600		1041	
Winstaandeel 4allTeX cdrom			12500	
Kleine verkopen			408	
Kleine aankopen		125		875
Afschrijving		1182		1182
Subsidies		2000		2000
Zaalhuur				125
Kantoorbehoeften				1192
Representatiekosten		200		1794
Reis- en verblijfkosten				245
Administratie		1000		
Bestuurskosten		750		
Porti				5195
Drukkosten				11177
Nieuwsbrief/verslagen		16500		
Bijdragen				103
Kamer van Koophandel		150		
Reisbijdragen		1500		
Bankkosten				43
Onvoorzien		193		
Saldo				16469
	23600	23600	48160	48160

Balans

	op 1-1-1997		op 1-1-1998	
	Activa	Passiva	Activa	Passiva
Giro en bankrekeningen	29240		46182	
Kas	100		109	
Contributies		413		
Debiteuren			12644	
Crediteuren		12100		25254
Kapitaal		16827		33681
	29340	29340	58935	58935

Jaarverslag NTG jan–dec 1997

Gerard van Nes
secretaris

Algemeen

Het jaar 1997 was voor de NTG een druk en wederom succesvol jaar. Twee NTG-bijeenkomsten (beide druk bezocht) vonden plaats met vele goede lezingen van sprekers uit hoofdzakelijk eigen geledingen. Er verscheen wederom een tweetal uitgebreide uitgaven van de MAPS (Minutes & APpendiceS). Daarnaast ontvingen alle NTG-leden zowel de nieuwe tetex UNIX CD-ROM als de nieuwe 4allTEX dubbel CD-ROM.
Natuurlijk zijn ook wederom diverse leden actief geweest, bij onder meer het beheer van het Bulletin Board FGBBS, het beheer van TEX-NL, de NTG-Website, de uitgebreide hulpverlening bij vragen op de nationale TEX-NL en de ondersteuning van de internationale 4TEX-discussielijst.

Het NTG-bestuur

Het NTG-bestuur bestond in 1997 uit de volgende personen:

- E.H.M. Frambach, voorzitter,
- G.J.H. van Nes, secretaris,
- W.H. Dekker, penningmeester (vanaf juni 1997)
- F. Goddijn, bestuurslid,
- T.C.A.J. Hoekwater, bestuurslid,
- J. Hagen, bestuurslid.

Op de NTG-bijeenkomst in juni werd Wybo Dekker bij acclamatie gekozen als bestuurslid en penningmeester. Wybo verzorgde reeds de financiën van de NTG sinds december 1996.

Philippe Vanoverbeke continueerde ook weer in 1997 zijn functie van NTG België-commissaris en zorgde daarbij voor optimale contacten met onze Vlaamse leden.

Het NTG-ledenbestand

Het ledenaantal kende in 1997 weer een kleine stijging. Het aantal instituutsleden is sinds de laatste jaren nagenoeg stabiel.

Eind	1988	1989	1990	1991	1992
Leden	84	90	117	159	193
Inst. leden	—	15	21	28	30

Eind	1993	1994	1995	1996	1997
Leden	232	265	286	285	292
Inst. leden	33	36	33	32	33

De contributie werd in 1997 niet verhoogd. De contributieinning bleek — evenals in voorgaand jaar — weer de nodige inspanningen te kosten. De inningen en de verwerkingen van de penningen werden door de nieuwe penningmeester gestroomlijnd. Besloten werd om leden die te laat (na versturing van de voorjaars-MAPS) hun achterstallige rekening betaalden, de extra kosten in rekening te brengen.

De samenwerking van NTG met TUG — waarbij het lidmaatschap gecombineerd kon worden overgemaakt — bleek te resulteren in 74 NTG/TUG-leden eind 1997 (eind 1996: 81).
De snelheid van toezending van het TUG-tijdschrift 'TUGboat', welke in 1996 sterk was verbeterd, bleek in 1997 om diverse redenen weer wat achterop te geraken.

In de voorjaars-MAPS werd de volledige NTG-ledenlijst gepubliceerd.

De NTG-bijeenkomsten

Er zijn in 1997 twee NTG-bijeenkomsten georganiseerd welke gekenmerkt werden door een levendige discussie en een hoge mate van informatie-uitwisseling.

1. Op donderdag 12 juni 1997 bij de Technische Universiteit Delft.
   Aanwezig waren 51 leden.
   Thema: 'TEX & databases, DVI & PDF en SGML & TEX'.
2. Op zaterdag 15 november 1997 bij het AMC te Amsterdam.
   Aanwezig waren 40 leden.
   Thema: 'Hacken zonder hoogtevrees'.

De volgende lezingen werden gehouden:

- *TEX, PDF & PDFTEX*
  door Hans Hagen (Pragma, Zwolle);
- *TEX and the database idea*
  door Kees van der Laan;
- *General markup versus dedicated typography*
  door Hans Hagen (Pragma, Zwolle);
- *de verhouding tussen TEX/LATEX en SGML*
  door Taco Hoekwater (Kluwer Academic Publishers, Dordrecht).

Op de najaarsbijeenkomst, welke als thema had 'Hacken zonder hoogtevrees', vonden de volgende lezingen plaats:

- *Programmeren in stijl; een nieuwe interface tussen CWEB en LATEX*
  door Mark Potse (Academische Medisch Centrum, Amsterdam);
- *Programmeursproblemen*
  door Hans Hagen (Pragma, Zwolle);
- *Hoe DVIPS & ConTeXt kunnen samenwerken*
  door Gilbert van den Dobbelsteen;
- *Vaak gevraagde aanpassingen aan LATEX classes*
  door Erik Frambach (Rijksuniversiteit Groningen);
- *Een toelichting bij de nieuwe MAPS class*
  door Siep Kroonenberg (Rijksuniversiteit Groningen);
- *Hacken voor iedereen*
  door Hans Hagen (Pragma, Zwolle);
- *Een paar macro's voor fixed-grid typesetting*
  door Taco Hoekwater (Kluwer Academic Publishers, Dordrecht);
- *Fancybox gebruik voor layout-trucs van 'Story Line' (Engels literatuuronderwijs)*
  door Herman Haverkort & Frans Goddijn.

Van bovengenoemde twee vergaderingen (en lezingen) verschenen verslagen met (uitgebreide) bijlagen in MAPS #18 en MAPS #19.
De leestafels met 'TEX-gerelateerde' boeken, eerdere uitgaven van de MAPS, diverse LUG- en andere tijdschriften, brochures en diverse aan TEX verwante geschriften, trokken zoals altijd veel belangstelling. Bij zowel de voorjaars- als de najaarsbijeenkomst konden de aanwezigen wederom Addison-Wesley (LA)TEX/PostScript boeken met korting aanschaffen.

De NTG-cursussen

In tegenstelling tot 1996, werden in 1997 door de NTG geen cursussen georganiseerd.

De NTG-MAPS

Ook in 1997 verschenen er twee MAPS uitgaven met totaal ongeveer 350 goed gevulde bladzijden met daarbij onder meer als inhoud: verslagen van nationale en internationale bijeenkomsten, artikelen m.b.t. de NTG-lezingen, tutorialen ander educatiemateriaal, TEX/LATEX & PostScript bijdragen, font-, TEX- en PC-artikelen, bijdragen m.b.t. TEX en LATEX macro's, en algemene plus technische TEX- en LATEX-bijdragen.

De MAPS-eindredactie was in 1997 in handen van hoofdredacteur Taco Hoekwater. Redactionele medewerking verleenden Gilbert van den Dobbelsteen, Siep Kroonenberg en Jos Winnink. Een MAPS-layoutverandering vond plaats.

De NTG-leden ontvingen met de voorjaars-MAPS de nieuwe tetex UNIX CD-ROM.

In 1997 verschenen geen MAPS-specials.

MAPS-Awards werden in 1997 uitgereikt aan Ton Biegstraten voor zijn uitgebreide en aantrekkelijke artikel over fonts, aan Gerrit Oomen (Kluwer) die aan Taco Hoekwater de ruimte en stimulans heeft gegeven voor de uitgebreide redactiewerkzaamheden t.b.v. de MAPS en aan Gerard van Nes voor al zijn MAPS-verzendingen.

De NTG-werkgroepen

Bij een inventarisatie bleken in 1997 ook de laatste NTG-werkgroepen van het eerste uur te zijn ingeslapen. Diverse werkzaamheden werden verricht binnen de werkgroep 'Spelling'.
Daarnaast kwamen de nodige activiteiten vanuit individuele leden. De veel gebruikte TEX-NL discussielijst zorgde enkele keren voor een vorm van 'overleg'.

De NTG 4allTEX CD-ROM

Eind december verscheen de vierde editie van de internationaal zeer gewaardeerde 4allTEX CD-ROM in een oplage van 3000 stuks. Deze nieuwe dubbel CD-ROM werd nog voor de jaarwisseling gratis aan alle NTG-leden toegestuurd. Daarnaast kregen alle leden van elke LUG's (Local TEX User Groups), waaronder die van Polen, India en Griekenland een gratis exemplaar van de derde editie.

Veel werk werd in het gehele CD-ROM ontwikkel- en produktieproces verzet door het koppel Wietse Dol & Erik Frambach. Verschillende leden van het NTG-bestuur waren betrokken op het logistieke- en voorlichtingsvlak.

Vooruitgang werd echter geboekt m.b.t. de onderhandelingen met de komende 4allTEX uitgever Addison-Wesley. Met ingang van de 5e editie (eind 1998) zal de distributie via deze uitgever plaatsvinden.

De NTG WWW-site

De NTG op WWW werd in 1997 druk bezocht. Met medewerking van Erik Frambach en Piet van Oostrum zijn vele WWW-pagina's en PDF-files (alle MAPS-uitgaven) beschikbaar gesteld via de (NTG) WWW-server bij de Universiteit van Utrecht. Het adres: `http://www.ntg.nl/`

Verdere Bestuursactiviteiten

De NTG wisselde informatie uit met de Locale TEX Gebruikersgroepen (LUG's): exemplaren van de MAPS werden verstuurd naar de meeste zusterorganisaties (inclusief TUG). Van onder meer de Duitse, Franse, Engelse en Tjechische gebruikersgroepen ontving de NTG de periodieken (zie de leestafel tijdens NTG-bijeenkomsten!).

Mede namens de NTG werden in 1997 de volgende bijeenkomsten bezocht:

- **BachoTEX'97 te Polen.**
  Van 1 tot 3 mei werd door Erik Frambach en Kees van der Laan deze Oost-Europese GUST'97 bijeenkomst bijgewoond.
  Een verslag van deze bijeenkomst is in MAPS #18 opgenomen.
- **TUG'97 te San Francisco**
  Van 27 juli tot 1 augustus bijgewoond door Erik Frambach, Taco Hoekwater en Hans Hagen (als spreker).
  Een verslag van deze bijeenkomst is in MAPS #19 opgenomen.

Op 20 mei (te Arnhem) en 27 oktober (te Deil) vonden bestuursvergaderingen plaats waarin naast de algemene gang van zaken binnen de NTG, onder meer de organisatie van de NTG-bijeenkomsten, Public Relations activiteiten, financiële zaken, de toekomst van TEX en de NTG, CD-ROM activiteiten, MAPS-aangelegenheden, lustrumactiviteiten, ledenwerving, ledenenquête, taakverdeling (opvolging) binnen het bestuur en de gebeurtenissen binnen TUG en andere zusterorganisaties ter sprake kwamen.

Binnen het bestuur werd regelmatig en uitgebreid gediscussieerd over het nut van TEX en NTG in het bijzonder, binnen de huidige en toekomstige wereld van documentverwerking. Eind 1997 werd aan alle NTG-leden een uitgebreide enquête gestuurd waarvan de resultaten in MAPS #20 zullen worden gepresenteerd.

Een lustrumcommissie werd in 1997 geformeerd om te werken aan de viering van het 10-jarig bestaan van de NTG in juni 1998.

Een wedstrijd werd uitgeschreven om te komen tot een nieuw ontwerp van het NTG-logo. Het resultaat zal op de voorjaarsbijeenkomst in 1998 gepresenteerd worden.

Diversen

- Van de TEX-NL discussielijst werd in 1997 wederom veel gebruik gemaakt. Het aantal abonnees bedroeg ruim 180. Het aantal uitgewisselde e-mails lag boven de 1100.
- Het NTG-FGBBS Bulletin Board van Frans Goddijn werd door zowel NTG-ers als niet NTG-ers regelmatig bezocht. Het FGBBS bevat naast materiaal van de 4allTEX CD-ROM, een indrukwekkende hoeveelheid aan (LA)TEX-materiaal.
- Vele leden maakten via FTP dankbaar gebruik van het CTAN mirror archief van Piet van Oostrum voor het verkrijgen van diverse soorten TEX-materiaal.

Nawoord Gerard van Nes

Dit is het laatste Jaarverslag van uw huidige NTG-secretaris. Mijn NTG-termijn zit erop. Tien jaar is mooi genoeg geweest. Een frisse wind vermindert nu eenmaal roestvorming. Zowel als secretaris als als MAPS-redacteur (t/m MAPS 15), heb ik met veel plezier een bijdrage mogen leveren aan de TEX-communicatie binnen Nederland en bij onze Zuiderburen.
Gedurende die tijd heb ik vele plezierige contacten erbij gekregen. Goede en snelle TEX-ondersteuning heb ik (ben nog steeds een eenvoudige LATEX-gebruiker...) mogen ontvangen; ondersteuning die ik zonder de NTG in principe nooit ontvangen zou hebben.

Tien jaar geleden gaf Kees van der Laan mij een LATEX-duwtje. Het *vrije systeemonafhankelijke logische documentconcept* sprak mij zeer aan. Alleen dacht ik *toen* dat ik op een eilandje zat. Dat was dus het eerste wat moest veranderen... Dat de NTG de 10 jaar zou halen hadden wij toen in het geheel niet kunnen voorspellen.
Ik gebruik (vanzelfsprekend) nu nog steeds LATEX. Zal het over 10 jaar ook nog steeds gebruiken. Met een glimlach zie ik ondertussen collega's alle versies van WordPerfect en vervolgens alle versies Word doorlopen. Waar zullen zij eindigen?

TEX heeft m.i. (nog steeds) een uitstekende toekomst; hetzelfde geldt ook voor bijvoorbeeld PERL, hetzelfde voor bijvoorbeeld Linux. En zo zijn er meer producten. Allen *Public Domain*, met dusdanige mogelijkheden en kwaliteiten, dat vele commerciële 'alternatieven' er niet aan kunnen tippen. Ondersteund door een *actieve* gebruikersgroep. Zoals een NTG.
Ik wens de komende nieuwe secretaris veel succes. Natuurlijk geldt dat ook voor de MAPS-hoofdredacteur en het gehele bestuur.

TEX is nog lang niet uitgestorven: TEX-gurus zijn levenslang aan 'hun' product verslaafd. De NTG is voor deze groep een 'gezellige club'. Voor TEX-beginners gaat — mede door de NTG — een nieuwe wereld open. In de ogen van deze laatste groep is de NTG een waardevolle en enthousiaste vereniging.
Door vooral rekening te houden met deze laatste groep kan de NTG t.z.t. succesvol een *tweede* set van 10 jaren afsluiten.

NTG, bedankt alvast!

De NTG en het Internet

Jules van Weerden
email: Jules.vanWeerden@let.uu.nl
mmv Maarten Gelderman
email: mgelderman@bik.econ.vu.nl

abstract

De NTG is niet alleen met webpagina's op het Internet aanwezig, maar ook middels een aantal discussielijsten. In dit artikel wordt kort aangegeven wat een discussielijst is, hoe je je aan- en afmeldt en wat je te wachten staat na aanmelding. Tevens wordt een overzicht gegeven van de in Nederland aanwezige TEX-gerelateerde lijsten.

keywords

discussielijsten, tex-nl

Het bekendste onderdeel van het Internet is ongetwijfeld het World Wide Web. Elders in deze MAPS staat een bijdrage van Piet van Oostrum over de web-pagina's van de NTG. In deze bijdrage wil ik het hebben over een onderdeel van Internet dat vaak, te vaak, over het hoofd wordt gezien: discussielijsten. Eerst zal ik globaal een beeld schetsen van wat zo'n lijst is en hoe een discussielijst werkt. Vervolgens kom ik toe aan het bespreken van de Nederlandse TEX-gerelateerde discussielijsten.

Discussielijsten

Veel mensen zijn zich niet bewust van het feit dat er discussielijsten bestaan. De reden hiervoor is eenvoudig. In tegenstelling tot bij voorbeeld het World Wide Web of ftp heb je voor het gebruik van discussielijsten geen aparte software nodig. Een email-programma (en natuurlijk toegang tot Internet) volstaan. De werking van een discussielijst zelf is eigenlijk te simpel voor woorden. Een discussielijst is voor de gebruiker niets anders dan een email-adres, bij voorbeeld `discussielijst@lijstencomputer.nl`. Alle mail die naar dit email-adres wordt gestuurd wordt automatisch doorgestuurd naar alle leden van de discussielijst. Het email-adres van de belangrijkste lijst in Nederland, tex-nl, is bij voorbeeld `tex-nl@nic.surfnet.nl`. Alle mail die naar dit adres wordt gestuurd komt bij alle 183 leden aan.[1] Op de lijstcomputer draait een softwarepakket (listserv en Majordomo zijn de populairste pakketten) dat er voor zorgt dat dit automatisch gebeurt. De op deze wijze gecreëerde lijst kan gebruikt worden om over van alles en nog wat te discussiëren. De TEX-lijsten hebben veelal een vraag- en antwoordkarakter. De leden van deze lijsten stellen op de lijst vragen over TEX-gerelateerde problemen en de andere leden (en met name Piet van Oostrum die hiermee terecht een erelidmaatschap van de NTG heeft verdiend) proberen deze vragen te beantwoorden.

Tot zover erg aardig natuurlijk, maar dan moet een lijst wel leden hebben. Ook dit proces is geautomatiseerd. Naast het adres `discussielijst@lijstencomputer.nl` bestaat er een adres waarmee het programma direct benaderd kan worden, bij voorbeeld `listserv@lijstencomputer.nl`. Het is niet de bedoeling dat er naar dit adres gewone berichten worden gestuurd, daar kan de software niets mee. In plaats daarvan moet de mail korte commando's bevatten. Je kan bij voorbeeld een mailtje met als inhoud

```
HELP
END
```

versturen naar `listserv@nic.surfnet.nl`, je krijgt dan een mailtje met uitleg over de beschikbare commando's terug.[2] Om je aan te melden als lid van bij voorbeeld tex-nl, kan je het volgende mailtje versturen naar `listserv@nic.surfnet.nl`:

```
SUBSCRIBE TEX-NL Voornaam Achternaam
END
```

De listserv-software kan vervolgens drie dingen doen.

1. Een mailtje terugsturen met de mededeling dat je geen lid kunt worden. Bij tex-nl is het uiterst onwaarschijnlijk dat dit gebeurt, maar bij andere lijsten (b.v. de bestuurslijst van de NTG) kan dit voorkomen.
2. Een mailtje terugsturen met de mededeling dat je lid bent geworden van de lijst. In dit mailtje wordt tevens uitgelegd hoe je je weer af kunt melden. Bewaar het!

1. Dit is niet helemaal waar. Indien het mailtje door één van de 183 leden wordt verstuurd, komt het normaal gesproken alleen bij de overige leden aan. Wil je het ook zelf weer terughebben stuur een melding aan het beheersadres (zie verder) met als commando 'repro <lijstnaam>'. Op sommige lijsten is het bovendien alleen aan leden van de lijst of soms zelfs alleen aan een lijstbeheerder toegestaan om mail te verzenden.

2. Helaas zijn de commando's voor listserv en Majordomo niet identiek.

3. Een mailtje terugsturen met de mededeling dat je voor de zekerheid nog even moet bevestigen dat je echt lid wilt worden van de lijst. Dit is met name om te voorkomen dat ongeldige emailadressen aan een discussielijst worden toegevoegd en om te zorgen dat mensen niet ongewenst aan een lijst worden toegevoegd.

Na deze eenvoudig actie ben je lid van tex-nl en kan je mail (met vragen over TEX en LATEX naar deze lijst versturen. Tevens ontvang je vanzelfsprekend alle mail die anderen naar deze lijst zenden (gemiddeld zo'n 25 mailtjes per week). Om je weer af te melden stuur je een mailtje met het commando `UNSUBSCRIBE TEX-NL` naar de listserver.

Lijsten in Nederland

Ook op het gebied van de electronische berichtenuitwisseling gaat het goed met de NTG. Op dit moment is een zevental lijsten geïnitieerd vanuit Nederland (voor zover ik weet ;-).

tex-nl Algemene discussie lijst over TEX in Nederland. [aantal abonnees: 183].

4tex Lijst voor vragen en mededelingen betreffende het 4TeX-systeem van Erik Frambach en Wietse Dol. [aantal abonnees: 275].

ntg-sgml Discussielijst van de SGML-werkgroep binnen de NTG. [aantal abonnees: 9].

ntg-tex-tools Vragen-, antwoorden- en opmerkingenlijst over allerlei programmatuur die gebruikt kan worden om de resultaten van TEX nog beter te maken. Of zelfs om TEX te genereren. [aantal abonnees: 27].

ntg-context Discussie lijst over ConTeXt, een parameter-gestuurd TEX macro pakket. [aantal abonnees: 11].

ntg-ppchtex Deze lijst gaat over PPCHTeX, een TEX macro pakket dat gebruikt kan worden om chemische structuurformules te zetten. [aantal abonnees: 14].

ntg-toekomsttex Discussielijst over de toekomst van TEX. [aantal abonnees: 7].

De eerste twee lijsten zijn te gast op de email-server LISTSERV van SURFNET. De lijsten die beginnen met 'ntg-' zijn te gast bij de MajorDomo email-server van de faculteit der Letteren in Utrecht.

Verder zouden nog de lijsten genoemd kunnen worden:

teTeX -lijst, waarop gediscussieerd wordt over het totaalpakket dat Thomas Esser heeft samengesteld van de basis-programmatuur die je nodig hebt om TEX op UNIX te draaien. Is de basis van de TEX -live CD. [aantal abonnees: 668].

CTAN-Ann - De beheerder(s) van CTAN gebruiken deze lijst voor de aankondigingen van nieuwe bestanden op CTAN. Handig om op de hoogte te blijven van het uitkomen van de nieuwste macro's en programma's. Je kunt er zlef (dus) niet posten.

Op alle lijsten wordt levendig gediscussieerd. Voor de lijsten tex-nl, 4TEX en teTeX houd ik (JvW) zelf een archief bij en omdat het niet erg geheim is, kan iedereen daar ook bij via de ftp-server ftp.let.ruu.nl in de directory: /pub/tex/<lijst> [lijstnamen in kleine letters].

Let verder op het verschil tussen de LISTSERV- en de MajorDomo-software. Voor de eerste kun je op de regel 'subscribe <lijstnaam>' ook nog een willekeurige tekst toevoegen. Bv

```
subscribe tex-nl  Jules van Weerden, CIM, Fac \
                                der Letteren, UU, NL
```

De vrije tekst wordt als kommentaar in de verzendlijst opgenomen en door de programmatuur genegeerd. Bij MajorDomo mag die extra informatie er NIET staan. Die wordt nl beschouwd als het email adres waar de berichten heen moeten.

Als je je voor een lijst wil aanmelden moet je een emailbericht sturen aan het bijbehorende adres. Het onderwerp is niet echt van belang, maar je krijgt het in de reactie terug, dus je kunt het er aan herkennen. In de tekst van het bericht neem je de regel op: subscribe <lijst>.

De verschillende aanmeld-adressen:

- Voor tex-nl en 4TEX: LISTSERV@nic.surfnet.nl.
- Voor ntg-tex-tools, ntg-ppchtex, ntg-context en ntg-toekomsttex: Majordomo@ntg.nl.
- voor teTeX: Majordomo@informatik.uni-hannover.de
- voor CTAN-Ann: listserv@urz.uni-heidelberg.de

De reden dat de lijsten met 'ntg-' beginnen is dat we (nog) geen scheiding hebben aangebracht tussen de lijsten van de faculteit der Letteren[3] en de lijsten van de NTG. Hopelijk kunnen we binnenkort alles splitsen. Dan krijg je ook de reactie terug van MajorDomo@ntg.nl en niet zoals nu van MajorDomo@let.(r)uu.nl.

Voor alle lijsten geldt dat als je een probleem met de lijst hebt dat je een bericht kunt sturen aan owner-<lijst>@<email-server> waar een 'echt' persoon achter zit, die kan helpen met het oplossen van het probleem.

Geniet van de lijsten en bestrijd de SPAM (ongewenste reclame).

3. In de maand mei gaat de hele universiteit over van de domeinnaam RUU.NL naar UU.NL. Tijdelijk komen de twee domeinen door elkaar voor, maar de 'ruu'-versie moet gaan verdwijnen

FGBBS op snelheid
Verslag van FGBBS

Frans Goddijn
(fg@fgbbs.iaf.nl)

abstract

In het afgelopen jaar heeft het FGBBS stabiel doorgedraaid, met een gestaag minderend bezoekersaantal. De maximale communicatiesnelheid is in 1998 verhoogd door aansluiting op ISDN. Met minder geld kan nu meer worden gedaan: gratis emailaccount voor NTG-leden.

keywords

BBS, bulletin board system, file requests, email, library

Stabiliteit en veroudering

De software waarmee FGBBS draait, is stabiel gebleken. Er staan wat kleine problemen te wachten rond het millennium, maar meer dan het verlies van wat berichten is daar niet te vrezen. Doordat ik geen essentiële onderdelen van de software heb verandered, is er ook geen grote vooruitgang op dit vlak geweest.

Doordat er weinig nieuwe files op FGBBS zijn geplaatst, heeft de collectie bestanden meer het karakter van een antiquariaat dan van een hippe TeX boetiek. Inmiddels heeft de CDrom de plaats ingenomen van het BBS als distributiemogelijkheid van softwareverzamelingen. Ik neem me voor de nieuwe 4TEX CD ook op FGBBS aan te sluiten, maar daar is het tot vandaag niet van gekomen.

Oud, maar rap ter been

De hardware is sinds het begin wel sneller geworden. FGBBS draait momenteel op een Pentium 166 machine en begin 1998 is er een ISDN aansluiting gerealiseerd, waardoor hogere transmissiesnelheden kunnen worden behaald.

Lagere kosten

Na de hoge kosten van de aanschaf van ISDN-apparatuur kwam er een meevaller: de provider IAF waar FGBBS een subdomein heeft, is overgegaan op lokale inbelpunten in heel Nederland. Voorheen bracht het ophalen van grotere files of grote hoeveelheden berichten (denk aan comp.text.tex) nog de nodige extra kosten met zich mee doordat de modem van FGBBS communiceerde met IAF via een interlokale verbinding van Arnhem naar Enschede. Tegenwoordig gaat dat allemaal op lokaal tarief én het gaat nogeens op minimaal de dubbele snelheid van voorheen.

NTG-lid@fgbbs.iaf.nl

Door deze lagere kosten kan ik NTG-leden op verzoek een eigen emailaccount aanbieden. NTG-leden die van deze mogelijkheid gebruik willen maken, hoeven hier niets voor te betalen. Dit kan aantrekkelijk zijn voor leden die op hun werkadres over email beschikken, maar thuis nog niet (terwijl thuis wel een PC met modem staat). Een apart abonnement bij een Internetprovider als Planet Internet wordt ermee bespaard, maar er moet wel per modem naar Arnhem worden gebeld en er kan niet worden gesurfd. . . . Software (shareware) voor installatie en gebruik van het gratis emailabonnement via FGBBS kan vanaf het BBS zelf worden opgehaald. Er moeten wel apart wat nummers en wachtwoorden worden afgesproken. Hoewel deze gratis abonnementen in eerste instantie ter beschikking staan van NTG-leden als gebruikers van TeX (TeX mailinglijsten zijn al op FGBBS aanwezig), vindt de uitwisseling van persoonlijke email gecomprimeerd (onzichtbaar voor andere inbellers en de beheerder) plaats zodat ook persoonlijke tekstberichten vrij zijn te verzenden en te ontvangen. Het verzenden en ontvangen van grote files via Internet email is via FGBBS ook mogelijk, maar een beetje onpraktisch. Dat wordt dan ook ontmoedigd. FGBBS gebruikers kunnen onderling wel simpel files van vrijwel onbeperkte grootte uitwisselen.

De automatische inbel-shareware draait onder Windows, is makkelijk te installeren en er wordt alleen contact gelegd met FGBBS om berichten (of files) te halen en te brengen. Het lezen en schrijven van berichten gebeurt terwijl de modemverbinding uit staat. Dit bespaart de kosten van het ‘on-line’ lezen.

Bezoekers

Er zijn diverse gebruikers van het systeem die regelmatig via automatische inbelsoftware raadplegen en zo hun post en andere bestanden ophalen. Dit zijn de ‘vaste klanten’ voor wie passwords worden bijgehouden. De ‘losse aanloop’ van interactief inbellende bezoekers die te allen tijde

toegang hebben tot alleen de TeX gerelateerde gebieden van FGBBS is gestaag aan het minderen. Belde er vorig jaar nog gemiddeld een persoon per dag, de laatste drie maanden geeft de statistiek aan dat er eens in de twee dagen zo'n bezoeker het BBS raadpleegt. Kort geleden leverde dat nog een nieuw NTG-lid op, maar de vraag kan gesteld worden of het resultaat van die voortdurende bereikbaarheid wel voldoende is in verhouding tot de investering die er in geld (o.a. subsidie van de NTG) en moeite in gaat zitten. Voor mij is het in elk geval niet teveel moeite en dankzij bovengenoemde investeringen en het lokaal inbellen naar IAF is er de komende tijd minder geld voor nodig.

4TEX bestellingen

Het FGBBS serveert ook een paar besloten mailinglijsten, zoals die voor het verwerken van de bestellingen van de 4TEX CD. De verkoop daarvan gaat dit jaar een keer ophouden, aangezien Addison-Wesley het dan overneemt van onze Gerard van Nes. Ik ben benieuwd of zij het net zo goed gaan doen als onze secretaris!

Nieuwe files laatste halfjaar

Als ik op FGBBS een lijst genereer van de files die er de laatste zes maanden (voor 1 april 1998) zijn binnengekomen, dan levert dat dit overzicht op:

Algemeen

filenaam	bytes	toelichting
FGBBS.ARJ	62675	allfiles op FGBBS public hard disk
FGBBS.LST	201652	allfiles op FGBBS public hard disk
ABNDAT02.ZIP	102338	De Algemene Bbslijst Nederland
ABN02.ZIP	162899	De Algemene Bbslijst Nederland

ConTEXt

filenaam	bytes	toelichting
BMAN-NL.ZIP	3714408	Beginner's manual, DUTCH
CMAN-NL.ZIP	4506636	Context manual, DUTCH
CONTEXT.ZIP	593082	CONTEXT package
CQRC-NL.ZIP	603831	a PDF file
PPCHTEX.ZIP	52146	PPCHTEX packaga
TEXUTIL.ZIP	66822	Utilities with CONTEXT

Geometry

filenaam	bytes	toelichting
GEOMETRY.DTX	64697	Geometry basis file
GEOMETRY.DVI	41516	printable file
GEOMETRY.INS	274	
GTEST.TEX	1561	test
LAYOUT.PS	52826	PostScript overzicht
README.HTM	1716	readme in HTML formaat
SHOWFRAM.STY	6425	SHOWFRAME.STY (afgekort)

TEXshells en editors

filenaam	bytes	toelichting
PMCSTEX.ZIP	442799	Macro package for OS/2 EPM editor
PMCSTEX.TXT	1429	readme file with PMCSTEX Macro package for OS/2 EPM editor
EMXRT.ZIP	539967	+ Run time library (Emx 0.9a, fix 03) voor Emacs, etc.

Perl

filenaam	bytes	toelichting
PERL502B.ZIP	1446122	Perl 5 for OS/2
DOSPERL.ZIP	2375453	PERL 5.003_93 works under DOS (originally for OS/2)
DOSPERLP.TXT	6635	text on Ilya Zakharevich's OS/2 port of PERL (works with DOS also)
DOSPERL.TXT	1301	explanation: how to install PERL 5
PERLBOOK.ZIP	2912108	perl 5.004001 documentation in PDF
COURSE.ARJ	55009	(PostScript) PERL course
COURSE.HTM	52562	same perl course in html format
COURSE.TXT	46144	same perl course in plain text format
BOOKS.HTM	14114	html format PERL manual reviews
BOOKS.TXT	10647	text format PERL manual reviews
ERR-APR.94	1444	errata "Learning Perl"
ERR-NOV.93	4009	errata "Learning Perl"
EXAMPLS.TXT	2231	PERL examples from LEARNING PERL
EXAMPLS.PL	53148	examples from the LEARNING PERL
CPAN.ANN	4282	Announcement from the CPAN maintainer (includes list of ftp sites)
JAPH	25687	Just Another Perl hacker scripts
LP.GIF	89532	Cover of Learning Perl book
LWQUOTES	17771	Quotes from Larry Wall
PPERL.GIF	107846	Cover of Programming Perl book
PPTEST	7109	Perl Purity test
BM.ZIP	13281	Netscape bookmarks <=> Internet explorer favorites. Written in perl
PERL2EXE.ZIP	278277	perl compiler for perl v. 5.003, probably needs perl32/win
VSPA.ARJ	43273	a set of perl scripts from the Vented Spleen BBS

Perl for OS/2

filenaam	bytes	toelichting
INSTALL.TXT	863	install how-to text
PLREADME.ZIP	16443	README.os2 and latest patches
PLINSTL0.ZIP	680875	Executables for installator
PERL5DOS.HTM	4174	Piet van Oostrum's advice how to use this version of PERL for DOS

		(also a DOS box under WIN and OS/2)
RSX510B.ZIP	175591	RSX, needed to run this version under DOS (with WIN & OS/2)
RSX510.TXT	1197	about RSX
PLINSTAL.ZIP	22861	Data for installator
PERL_EXC.ZIP	274412	Perl VIO and PM executables
PERL_MLB.ZIP	838796	Main Perl library
PERL_SH.ZIP	186101	Pdksh
PERL_AOU.ZIP	324537	Perl_ VIO executable (statically linked)
PERL_BLB.ZIP	542864	Tools to compile Perl modules
PERL_INF.ZIP	1614109	Perl manual in .INF format
PERL_MAM.ZIP	741868	Manpages for Perl modules
PERL_MAN.ZIP	758517	Manpages for Perl and utilities
PERL_POD.ZIP	554487	Source for Perl documentation
PERL_STE.ZIP	443402	Additional Perl modules
PERL_UTL.ZIP	188382	Executables for Perl utilities
PATCHES.ZIP	6248	<no description>

Words

filenaam	bytes	toelichting
AARDR-NM.DOS	17735	Aardrijkskundige namen, DOS versie
WPLEX.EXE	1448995	Gert Hardeman's WP woordenlijsten,
WPLEX.LME	4078	Toelichting door Gert Hardeman

Anti Virus Software

SCN-314E.ZIP	706181	VirusScan for DOS by McAfee, Inc.

Utilities van algemeen nut

filenaam	bytes	toelichting
EMXRT.ZIP	539967	The emx runtime 0.9c fixlevel 4.
FDAPXINF.ZIP	342746	FDAPX manual in PDF formaat
FIDOBOEK.ZIP	318407	boekje over FIDOnet, PDF formaat
FTNM0A4.ZIP	57598	FTN manager, netmail manipulator
OS2TEICO.ZIP	870	OS/2 TimEd Icons
WINTEICO.ZIP	2152	WIN TimEd icons
MR2I140.ZIP	1065849	MR/2 ICE mail program
PN2_100.ZIP	1415884	ProNews/2 News Reader, also for binary newsgroups
SB-V11.ZIP	1763503	long filename: SB16-32-64_V11.ZIP for SB16/AWE32/64 v1.1. drivers Creative Labs made for OS/2, and all utilities for these sound cards
SB-V11.TXT	1154	explanation
WNOTE105.ZIP	26927	Yellow notes on screen for OS/2
IDEDASD.EXE	88285	IBM IDE Hard File drivers for larger than 4.3 Gb Files.

Bericht

10 jaar NTG
wat vinden de leden

Maarten Gelderman
Hans Hagen
maps@ntg.nl

Keywords
NTG, MAPS, enquête

abstract

In this article we present the results of the survey the NTG organized on the occasion of its 10th anniversary. Membership figures of the NTG neared 300, of which over one third participated in the survey. In their response the members express their continuous need for the MAPS and the meetings as well as the widely used 4allTEX and TEXLive CDROM's. In general we can conclude that the members are quite content with the current situation, but also challenge the board not to forget the novice and not so experienced users.

Inleiding

Een half jaar geleden vroeg het NTG–bestuur u een vragenlijst in te vullen. Eén van de aanleidingen voor deze lijst was het uit te stippelen beleid voor de komende jaren. Er zijn 106 vragenlijsten verwerkt. Dit betekent dat ongeveer een derde van de leden de vragenlijst ingevuld heeft geretourneerd. Dit lijkt een geringe respons ten opzichte van het totale bestand van 300 leden, maar is nog altijd bijna het dubbele van het aantal bezoekers van de bijeenkomsten. Natuurlijk dienen we ons wel te realiseren dat met name de actieve leden hebben gereageerd. Aan de andere kant is het echter ook waarschijnlijk dat leden die ernstige klachten over de vereniging hebben deze gelegenheid hebben aangegrepen om die te uiten.

NTG

Opvallend is dat de meeste respondenten erg positief zijn over de NTG. Zo'n 95% van de respondenten vindt de NTG goed georganiseerd en is van mening voldoende waar voor zijn/haar geld te krijgen. Enig potentieel punt van kritiek is dat beginners (naar zo'n 25% zegt) bij de NTG niet evenveel aan hun trekken komen als gevorderden.[1] Gezien de tevredenheid van de leden moeten we ons echter wel afvragen of dit een probleem is. Misschien is de NTG wel een vereniging voor gevorderden. Dit blijkt ook uit de gemiddelde ervaring met TEX die maar liefst 8 jaar bedraagt (zie grafiek).

jaren	%	aantal
1	4.7	𝍸
2	2.8	𝍷
3	2.8	𝍷
4	6.6	𝍸 𝍷
5	9.4	𝍸 𝍸
6	7.5	𝍸 𝍷
7	11.3	𝍸 𝍸 𝍷
8	10.4	𝍸 𝍸 𝍷
9	6.6	𝍸 𝍷
10	22.6	𝍸 𝍸 𝍸 𝍸 𝍷
11	1.9	𝍷
12	.9	𝍷
13	.0	
14	.0	
15	7.5	𝍸 𝍷
16	.9	𝍷
17	.0	
18	.0	
19	.0	
20	.9	𝍷

Tabel 1 Aantal jaren ervaring (n=103).

Kijken we wat de leden aan de NTG waarderen, dan blijken vooral de met betrouwbare regelmaat verschijnende MAPS en de CDROM's een positieve bijdrage te leveren. De belangrijkste redenen om lid van de NTG te worden en blijven liggen in het warme hart dat men TEX toedraagt en de behoefte aan informatie, ondersteuning en kontakten.

Alhoewel de leden overwegend tevreden blijken met de koers van de NTG, is de enquête voor het bestuur beslist

1 Interessant is de relatie tussen TEX ervaring en de mate waarin men vindt dat beginners en gevorderden beiden aan hun trekken komen binnen de NTG: hoe meer ervaren de leden zijn, des te sterker hebben zij de angst dat dit niet het geval is. Het verband is niet heel erg sterk $r = -0{,}136$; $\alpha < 0{,}05$. Misschien onderschatten de ervaren gebruikers hun minder ervaren collegae wel een beetje.

geen aanleiding om het nu maar rustig aan te gaan doen. De veelheid aan suggesties en opmerkingen geven het bestuur de komende jaren voldoende stof tot discussie, niet in de laatste plaats omdat men als belangrijke taak voor de NTG ziet weggelegd „het promoten van TEX" en het beschikbaar stellen van TEX aan potentiële gebruikers.

In de antwoorden op de (overigens vaak niet ingevulde) open vragen over de sterke en zwakke kanten van de NTG klinkt enerzijds tevredenheid door, maar anderzijds ook enige verontrusting. Onder de sterke punten noemt men: het enthousiasme, de actieve kern van guru's, de goed gevulde MAPS, de CDROM's, de discussielijst TEX-NL en de bijeenkomsten. De sfeer is goed, de vereniging reageert accuraat en is goed georganiseerd, men krijgt voldoende informatie, ook over internationale ontwikkelingen, en de bijeenkomsten dragen een open karakter.

Daartegenover staat de sterke afhankelijkheid van een kleine groep actieve leden. Ook is men bang dat de vereniging niet altijd even toegankelijk is voor en niet gericht is op leken. Het verzorgen van meer cursussen zou hiervoor een oplossing zijn. Een enkeling denkt dat groei noodzakelijk is omdat de NTG anders te kwetsbaar wordt. Meerdere respondenten vinden dat de NTG best wat meer naar buiten mag treden: we zijn te bescheiden.

MAPS

De MAPS blijft het vlaggenschip van de NTG. De inhoud is volgens meer dan 90% van de leden interessant, actueel en divers. Eenzelfde percentage is tevreden over de omvang, vindt de MAPS van hoog niveau en de lengte van de artikelen goed. Tussen de 80 en 90% vindt de vormgeving aantrekkelijk en de artikelen van hoge kwaliteit. Tegenover al deze positieve geluiden staat wel het feit dat bijna een vijfde van de respondenten moeite heeft de artikelen te volgen.

Veel leden vinden de MAPS geen eenvoudige kost. Uit de opmerkingen blijkt echter dat men dit eigenlijk niet zo erg vindt. Ook zij die moeite hebben met het lezen van de MAPS realiseren zich dat ook de artikelen van een wat hoger niveau voor een belangrijke groep leden hun waarde hebben. Als bestuur zijn wij ons er echter wel van bewust dat wij ook onze wat minder ervaren leden voldoende moeten bieden. In volgende MAPS-en en op volgende bijeenkomsten zal er nadrukkelijker naar worden gestreefd specifiek op hen gerichte artikelen te bieden en lezingen te organiseren. In deze MAPS vindt u bij voorbeeld al de (naar wij hopen leesbare) bespreking van brandende vragen van TEX-NL en een nieuwe ToolBox.

Ook de open vragen met betrekking tot de MAPS zijn door slechts een kleine groep respondenten beantwoord. Uit deze antwoorden vallen de volgende plus- en minpunten af te leiden. Men mist in de MAPS artikelen over typografie, en dat terwijl TEX op dat vlak excelleert. Verder zag men graag een vraag/antwoord rubriek en hoort men graag nieuwtjes. Ook zouden ervaringen van gebruikers en het laatst nieuws over packages een vaste plaats in de MAPS moeten krijgen. Sommigen zien ook graag meer aandacht voor METAFONT en METAPOST. Dat dit noodzakelijk is blijkt ook wel uit het feit dat meer dan 70% van de gebruikers deze toepassingen nog nooit (bewust) gebruikt heeft. Meer voorbeelden van gebruik van TEX voor niet–wiskundige toepassingen kunnen naar de mening van een aantal respondenten ook geen kwaad.

Onder de minpunten scoren spel- en zetfouten hoog. Ook vinden enkele lezers dat de redactie moet toezien op beter taalgebruik (engels en nederlands). De indeling kan helderder en de vormgeving soms beter, maar over dat laatste lopen de meningen uiteen. Men prijst het niveau en de fraaie voorbeelden, hoewel de MAPS meer een showcase zou moeten zijn van wat TEX typografisch vermag. De inhoud wordt gekarakteriseerd als origineel, objectief, actueel, goed onderbouwd, overzichtelijk, consequent, gevarieerd en professioneel.

Als informatiebron naast de MAPS worden met name boeken en de TUGBOAT, het periodiek van de internationale TEX Gebruikersgroep TUG, genoemd (23%). Een enkeling leest tevens de periodieken van één of meer van onze andere zusterorganisaties.

Bijeenkomsten

Ook over bijeenkomsten blijken de leden over het algemeen positief. Meer dan drie kwart van de respondenten is tevreden over de inhoud van het programma. Laten we diegenen die het oneens of zeer oneens zijn met de stelling dat zij alle bijeenkomsten bezoeken weg, dan bedraagt dit percentage zelfs 90%. Van de respondenten is 37% het eens of zeer eens met de stelling dat men alle bijeenkomsten bezoekt. Meer bijeenkomsten hoeven er van het merendeel van de respondenten niet georganiseerd te worden, alhoewel nog steeds 30% het met deze stelling eens is.

Ongeveer dezelfde antwoorden worden gegeven op de vraag of er meer thematische bijeenkomsten moeten komen. Een meerdaagse conferentie vinden de leden niet noodzakelijk, terwijl meer cursussen door iets meer dan de helft van de respondenten op prijs zouden worden gesteld. De belangrijkste redenen om bijeenkomsten niet te bezoeken, liggen eenvoudig in tijdgebrek of omdat men er geen behoefte aan heeft. De keuze tussen een doordeweekse dag of de zaterdag blijft moeilijk. Met geen van beide opties kunnen wij al onze leden tevreden stellen.

De laatste conclusies worden bevestigd door de open vragen. Redenen om naar een bijeenkomst te komen zijn: het onderhouden van kontakten met gelijkgezinden, het opdoen van ideeën, het vergaren van informatie en het ontmoeten van guru's om vragen aan te stellen. Ook komt

men voor de gezelligheid en wil men het laatste nieuws horen. De mogelijkheid boeken te kopen wordt ook genoemd. Men laat zich lokken door thema's en vindt bijeenkomsten leerzaam en inspirerend.

Anderen zien in de thema's juist een reden niet te komen. Het merendeel van de niet–komers zou echter graag komen, maar is verhinderd door omstandigheden, afspraken of kan niet vrij krijgen. Relatief vaak wordt als argument de afstand genoemd, een reeël argument voor onze Belgische leden. Een enkeling noemt de kosten en relatief veel respondenten hebben problemen met het tijdstip en de dag. Er zijn er ook die de bijeenkomsten te moeilijk vinden. Daarnaast worden ook de argumenten gehanteerd dat men bijeenkomsten zowiezo niet interessant vindt en/of dat men TEX gewoon als tool ziet en geen behoefte heeft aan kontakten met andere gebruikers.

TEX zelf

Natuurlijk waren we niet alleen geïnteresseerd in meningen over de NTG, maar ook in de vraag waar onze leden TEX voor gebruiken, welke implementaties worden gebruikt en hoe tevreden men over TEX zelf is.

De meeste respondenten gebruiken TEX op het werk en daarnaast privé. Wat ons eerlijk gezegd verbaasde is dat ongeveer 15% TEX alleen privé gebruikt. Meestal is men dan ook via het werk of de studie in aanraking gekomen met TEX. Een enkeling kwam bij TEX via het internet, een Linux distributie, een uitgever of een advertentie. 91% is zo tevreden over TEX dat men niet op zoek is naar een alternatief. Iets meer dan de helft van de respondenten gebruikt naast TEX nog wel andere zet- en opmaakpakketten. Met name WordPerfect en Word worden genoemd, alhoewel de vermelding gepaard gaat van het nodige verbale gezucht, gesteun en getier (leden die deze klachten achterwege hebben gelaten staan vanzelfsprekend op de nominatie om geroyeerd te worden).

Aanzienlijk uitgebreider is de lijst met (teken)programma's:[2] Harvard Graphics, GnuPlot (zeer vaak), Excel, Lotus, Applix, TEXCad, BMPtoTEX, Photoshop, Picture Publisher, MatLab, Illustrator, CorelDraw, Atari, WordPlus, MicroGraphix, PSTricks, XFig, Freelance, Maple, PowerPoint, Visio, X*whatever*, AutoCad, MS*whoknows*, Paint Shop Pro, Mathematica en FrameMaker. Kortom: er zijn blijkbaar genoeg manieren om de grafische tekortkomingen van TEX te compenseren. In de meeste gevallen wordt overigens `.eps` gebruikt als uitvoer formaat, bitmap–formaten zoals `.pcx`, `.bmp`, `.gif` en `.tif` beslaan samen het grootste deel van de restcategorie.

TEX wordt door de leden vooral gebruikt voor artikelen, brieven en rapporten en in mindere mate voor het zetten van artikelen en boeken.[3] Ook wordt TEX gebruikt voor het maken van WWW pagina's, facturen, bibliografieën, muziek en speciaal zetwerk.

Mocht iemand u vragen waarom hij of zij TEX zou moeten gebruiken, dan heeft u hier enkele argumenten: TEX biedt een constante kwaliteit, is zeer stabiel en bovendien erg consistent en uitbreidbaar. TEX stelt nauwelijks eisen aan de computer, kan fantastisch formules zetten en is erg flexibel en programmeerbaar. De vormgeving is prima en men heeft bovendien alles in de hand. TEX stimuleert een logische ordening en scheiding van inhoud en vorm. Een enkeling vindt TEX helder en eenvoudig, en velen roemen de mogelijkheden. TEX kan grote jobs aan. Het afdwingen van huisstijlen is geen probleem. TEX is op vele computers beschikbaar, is bugvrij en is platformonafhankelijk. Verwijzingen, indices, bibliografiën, nummeringen, figuren, geen probleem voor TEX. Een voordeel is dat de invoer ASCII is. TEX is gratis, ruim beschikbaar en is geschikt voor iedere taal. Het font–concept is sterk, TEX kent ligaturen en weet op de juiste manier te spatiëren, en de macropakketten worden steeds beter.

Kleven er dan geen nadelen TEX? Vrijwel iedereen weet naast de sterke kanten ook zwakke punten te noemen, zoals: TEX is niet WYSIWYG en het aanmaken van formules, tabellen en figuren is dan ook niet eenvoudig. De interface is primitief en complex en de mogelijkheden zijn onoverzichtelijk. Sommigen vinden de beschikbare handleidingen matig, en velen vinden TEX geen gemakkelijk systeem voor beginners. Wellicht dat daardoor TEX relatief onbekend is in de PC wereld. Alles wat met fonts te maken heeft, is lastig, en TEX zal nooit een routine tool worden. Het vergt nogal wat deskundigheid om een eigen vormgeving te definiëren en de grafische mogelijkheden zijn beperkt. Sommigen missen lokale en commerciële support. TEX is traag en past niet in het rijtje van standaard tekstverwerkers (wat niet verwonderlijk is omdat TEX geen tekstverwerker is). Het kost tijd om inzicht te krijgen in de mogelijkheden en zelfs na langdurig gebruik blijft TEX lastig. LATEX verandert te vaak en is niet te onderhouden en de vormgeving van de standaard LATEX–uitvoer is sub–optimaal. In de Windows omgeving is TEX gebruik lastig. Sommigen missen kleur.

We zien is deze twee groepen antwoorden nogal wat tegenstrijdigheden. Kijkend naar de ingevulde lijsten, valt op dat men tegelijk zowel zeer positief als negatief kan zijn. Vooralsnog wegen de voordelen tegen de nadelen op en blijft men TEX gebruiken. Voor de ontwikkeling van opvolgers van TEX is bovenstaand lijstje echter van groot belang. De tekortkomingen moeten worden weggewerkt, dat

2 Wellicht zijn sommige namen verkeerd gespeld.
3 Voor alle volledigheid: sommige uitgevers zetten wel degelijk tijdschriften en boeken met TEX.

systeem	%	aantal
Atari	10.4	𝍸 𝍸 I
MSDOS	49.1	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 II
OS/2	9.4	𝍸 𝍸
MacOS	5.7	𝍸 I
UNIX	51.9	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸
WINDOWS	64.2	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 III

Tabel 2 Besturingssystemen (n=106).

is duidelijk. Tegelijkertijd dient de opvolger van TEX echter de voordelen die nu genoemd worden te respecteren: de nieuwe TEX zal zonder problemen moeten kunnen doen wat de oude TEX kan, stabiliteit wordt immers met nadruk genoemd als een voordeel van TEX.

TEX–distributies

Met welke TEX–distributies werken de NTG–leden? Natuurlijk is dit een vraag die mede afhangt van het besturingssysteem (operating–system) dat wordt gebruikt. Het is dus handig om eerst op het OS in te gaan. TEX–gebruikers, of misschien is het beter te zeggen onze leden, zijn in dit opzicht beslist niet eenkennig.

Zo'n 35% van de leden gebruikt één besturingssysteem. In de meeste gevallen Windows. De overige leden maken gebruik van 2 of meer systemen. Eén lid, maar dit lid rapporteert voor zijn organisatie, gebruikt zelfs vijf verschillende besturingssystemen. Atari–gebruikers, een opvallende 10% van de respondenten, gebruiken vaak tevens Unix of Windows. Circa de helft van de respondenten gebruikt MS–DOS en daarnaast Windows of Unix. MacIntosh, Acorn en OS/2 worden door minder dan 10% van de respondenten gebruikt. Unix door meer dan de helft, vaak in combinatie met MS–DOS of Windows. Windows zelf is de absolute topscoorder met 65%, maar kennelijk niet tot volle tevredenheid: de helft van de Windows–gebruikers gebruikt ook nog eens Unix.

Dan nu naar de kern van de zaak: de gebruikte TEX–implementatie. Ook hier blijken de respondenten niet eenkennig. Het gemiddeld aantal gebruikte implementaties bedraagt 1,8. Dit wordt overigens mede veroorzaakt door een groot aantal (21) gebruikers dat ‚zowel' EMTEX als 4allTEX gebruikt. De meest populaire implementatie is onze ‚eigen' 4allTEX, dat door bijna 60% van de respondenten wordt gebruikt. Deze populariteit wordt met name veroorzaakt door de MS–DOS– en Windows–gebruikers. emTEX, dat het van hetzelfde besturingssysteem moet hebben, is een goede tweede met bijna 40%, maar lift natuurlijk deels mee met 4allTEX.

Het verspreiden van CDROM's door de NTG blijkt een goede strategie om distributies populair te maken: TEXLive bezet de vierde plaats met bijna 30%. Opvallend is dat deze distributie het niet alleen van Unix–, maar ook van Windows–gebruikers moet hebben bij het behalen van zijn score. Ook opvallend is dat het de niet door ons verspreide teTEX distributie voor zich moet dulden. Deze distributie wordt door iets meer dan 30% van de respondenten gebruikt en is onder Unix gebruiker bijna twee keer zo populair als TEXLive. Hierbij dient wel gemeld te worden dat teTEX veelal standaard met Linux wordt meegeleverd. Verder dienen nog te worden genoemd StrunkTEX, als favoriet van de Atari–gebruikers en ozTEX en TEXtures die beiden onder Apple–gebruikers even populair zijn. Overigens, er zijn ook VMS gebruikers.

Kijken we naar de onderhoudbaarheid van de applicaties, dan blijkt dat 6% van de respondenten niet in staat is zelf de installatie te onderhouden, 13% kan dit alleen met hulp, 50% kan het meestal en 31% kan het altijd. De verschillen per implementatie zijn aanzienlijk. teTEX gebruikers blijken (er is niet getest op significantie) zichzelf het best in staat te achten de eigen installatie te onderhouden. Dit komt vermoedelijk door de goede integratie met (de distributie van) het besturingssysteem Linux. Het merendeel van de problemen wordt veroorzaakt door de MS–DOS/Windows–gebruikers en hun implementaties. De integratie tussen dit besturingssysteem en TEX is nog kennelijk nog steeds niet optimaal. Daarnaast kan natuurlijk meespelen dat Unix–gebruikers over het algemeen meer computerervaring hebben.

Niet verbazingwekkend was het te constateren dat het meest gebruikte macropakket LATEX2_ε is (83%). Wat wel opviel is dat er nog steeds 18 respondenten zijn die gebruik maken van het in de tussentijd verouderde LATEX2.09 zonder tevens LATEX2_ε–gebruiker te zijn. Natuurlijk is er nog een groep PLAIN TEX–gebruikers (26.4%). Overige pakketten die door meer dan een persoon worden gebruikt zijn EDMAC, CONTEXT en $\overline{\text{o}}$-TEX. Daarnaast zijn er nog vier gebruikers van het wat moeilijker te classificeren Scientific Word/Scientific Workplace.

printer	%	aantal
InktJet	45.3	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 ///
Matrix	8.5	𝍸 ////
LaserJet	66.0	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸
PostScript	67.0	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 /

Tabel 3 Printers (n=106).

distributie	%	aantal
4AllTEX	55.7	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 ////
TEXLive	28.3	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸
TEXtures	2.8	///
emTEX	36.8	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 ////
Web2C	10.4	𝍸 𝍸 /
ozTEX	2.8	///
teTEX	29.2	𝍸 𝍸 𝍸 𝍸 𝍸 𝍸 /
Y&YTEX	0.9	/
mikTEX	4.7	𝍸
strunkTEX	2.8	///
pcTEX	1.9	//

Tabel 4 Distributies (n=106).

WWW

Opvallend is dat ongeveer 40% van de respondenten nog nooit de WWW pagina's van de NTG heeft gezien. De overige invullers bezoeken gemiddeld een maal per maand de NTG site. Zo'n 10% van de respondenten bezoekt nog regelmatig het FGBBS.

Conclusie

De NTG doet het niet slecht, maar we kunnen best nog wel wat dingen verbeteren. We moeten als bestuur goed opletten dat de NTG bijeenkomsten en de MAPS ook voor nieuwe leden en beginners toegankelijk blijven. Daarbij moeten we natuurlijk onze bestaande leden wel tevreden houden. Ook ligt er een duidelijke taak op het gebied van de promotie van TEX. In lijn met dit argument moet vermoedelijk het meest frequent gebruikte antwoord op de vraag naar het aantal TEX–gebruikers in de organisatie (,te weinig') worden gelezen. Niets is frusterender dan, terwijl je weet dat het met TEX allemaal veel handiger kan, aan te moeten modderen in Word, omdat je collegae nog nooit van TEX hebben gehoord. Daarnaast worden in de enquêtes met zekere regelmaat suggesties gedaan voor nieuwe zaken die al lang bestaan. Hier kan de voorlichting van onze kant duidelijk beter. Zowel voor beginners als niet–gebruikers is de NTG PR–set een belangrijke informatiebron. De redactie van de MAPS is met de opmerkingen uit de enquête in het achterhoofd druk bezig deze te herzien en we hopen de verbeterde versie binnenkort aan alle leden toe te kunnen sturen.

Een geheel ander punt van aandacht is de relatief kleine kern van actieve leden. Als bestuur zijn we ons er terdege van bewust dat dit de vereniging kwetsbaar maakt. Ook een groot aantal respondenten blijkt dit op te zijn gevallen. Hopelijk mogen we hieruit af leiden dat deze mensen ook bereid zijn, binnen zekere grenzen, zelf actief te worden. Wie concrete ideeën heeft over een activiteit waarmee hij/zij de NTG kan helpen, of gewoon in het algemeen zijn/haar bereidheid een bijdrage te leveren aan de vereniging kenbaar wil maken, kan het bestuur direct bereiken op het boven dit artikel genoemde email–adres[4]. Natuurlijk is het bestuur ook voor al uw andere opmerkingen bereikbaar via bovenstaand adres. De titel van dit artikel is niet voor niets: NTG, wat vinden de leden?

4 In de enquête geeft een aantal respondenten aan iets te willen doen. Helaas ontbreekt veelal de naam, zodat wij zelf geen contact met u op kunnen nemen.

Het nieuwe NTG logo

Siep Kroonenberg
Faculteit der Economische Wetenschappen
Rijksuniversiteit Groningen
n.s.kroonenberg@eco.rug.nl

abstract
Introductie van en toelichting bij het nieuwe NTG-logo

keywords
logo, Encapsulated PostScript

In het vorige nummer van de MAPS heeft u kunnen lezen over de ontwerpwedstrijd voor een nieuw logo. En hier is dan het nieuw logo. U ziet dat in geen enkele versie de tekst 'Nederlandstalige TEX gebruikersgroep voluit voorkomt. Het leek me beter om per toepassing daar een keus voor te maken.

Ik heb met een aantal ideeen gespeeld:

1. iets dat lijkt op het TEX logo, met de 'T' naar boven gewipt
2. een monogram met meer dan 1 font: een strakke, geometrische 'T', en 'N' en 'G' uit een minder schematisch font.
3. elke letter in zijn eigen rechthoek plaatsen; dit als verwijzing naar de manier waarop TEX tekst zet. Hier is het uitkijken geblazen dat het logo er niet warrig uit gaat zien.

Deze ideeën leidden tot een aantal logo's; een combinatie van idee 2 en 3, met font cmb10:

Het is niet strikt noodzakelijk om TEX-fonts te gebruiken; het volgende logo, een implementatie van idee 1, gebruikt b.v. Bitstream's versie van Kabel. Een citaat uit de Adobe Type On Call CD:

> *Designed by Rudolf Koch and released in 1927 by the Klingspor foundry in Germany, Kabel is named in honor of the laying of the first trans-Atlantic telephone cable.*

De dwarsbalk van de 'T' is iets naar links verlengd. Een tweede versie is wit uitgespaard tegen een donker vlak. Hierbij is de afstand tussen de letters een fractie vergroot om optische effecten te compenseren.

Het winnende logo combineert idee 1 en 3. Ik was gecharmeerd door het stijlcontrast tussen de zeer simpele ronde lettervormen en de dunne strakke rechthoekjes. Omdat echter die rechthoekjes bij verkleining niet meer goed tot hun recht komen, heb ik ook een versie zonder rechthoekjes gemaakt:

Het font is Formal Script 421, Bitstream's versie van Ondine. Wederom een citaat uit Type On Call:

> *Released by the French type foundry Deberny & Peignot in 1954, this bold upright script was an early design of Adrian Frutiger and the only script face he ever created.*

Toch eens kijken of een soortgelijk effect niet met een TEX font te bereiken valt. Euler (`eusb10`) ligt het dichtst in de buurt:

Maar dit wil echt niet.

Mijn eigen voorkeuren waren het Kabel- en het Ondine logo, en het bestuur koos voor het laatste.

De kleur van het logo – als kleur werd gebruikt, b.v. op voorbedrukt briefpapier en voor de web-pagina's – zou identiek moeten zijn aan de MAPS-kleur. Omdat beeldschermen geen heldere blauw-groene kleuren kunnen produceren, werd de web-versie een veel stemmiger blauwgroen. Toen men die kleur zag werd besloten die ook voor gedrukte toepassingen te gebruiken. Voor de prepress-kenners: het Pantone-nummer is 549.

Op het moment van schrijven heeft het bestuur zich nog niet uitgesproken over een brief-stijl, maar als u dit leest heeft u waarschijnlijk al kunnen zien wat daar uit is gekomen.

Bij een logo kunnen letters het beste naar contouren worden omgezet. Daardoor hoeft het font niet beschikbaar te zijn bij het afdrukken van het logo. Veel tekenprogramma's kunnen dit en een logo in eps- (Encapsulated PostScript) formaat opslaan, en dat is wat ik heb gedaan. Voor niet-PostScript printers zal het logo voor een aantal resoluties worden omgezet naar bitmapped formaten.

Why `\expandafter` is sometimes needed by common users too.

1. When dealing with textual input, that is reading lines of text, TEX is nearly always expanding whatever it encounters. There are two primitives that can influence this proces: `\noexpand` and `\expandafter`. The first primitive will probably never surface in user input, but the latter may! Let's give an example. In most cases one will call for a new chapter by saying something like:

```
\Chapter{This Or That}
```

or

```
\Chapter{\ThisOrThat}
```

When `\ThisOrThat` has only one meaning and is never changed, this goes all right. The title is typeset and when called upon, it appears in the table of contents too.

2. Macro packages use auxiliary files to save entries of for instance tables of contents and indexes. When writing a chapter title to such a file, we can:

- expand the macros that are part of the title
- selectively expand those macros
- copy the title as is without expansion

The first alternative is the most simple. Hoewever, certain macros can expand into long sequences of tokens, that in the worst case are only partially expanded. Font switches are an example of such fragile commands.

A solution to this problem is partial expansion. This can be accomplished by preventing certain macros from expanding. This can be done quite easily at the macro programming level by preceding such commands by a `\noexpand` when the moment is there. But which macros need such precautions and which ones don't? And how is the user supposed to know this? Even worse, we expect users to explictly prohibit expansion for their own macros if needed, and in practice they end up with weird sequences like:

```
\Chapter{\dontexpand\ThisOrThat}
```

This leaves option three as the most save one. This method has one disadvantage. By copying the title verbatim, we got problems when `\ThisOrThat` is used in more chapter titles with different meanings. In such cases we have to expand `\ThisOrThat` on forehand:

```
\expandafter\Chapter\expandafter{\ThisOrThat}
```

The first `\expandafter` reaches over `\Chapter` and expands the next token, `\expandafter`. This second `\expandafter` reaches over the `{` and expands `\ThisOrThat`.

When we have a bit longer title, like the next one, we end up with a lot of `\expandafter`'s:

```
\Chapter{About \ThisOrThat}
```

Out of convenience, we prefer something like:

```
\expanded{\Chapter{About \ThisOrThat}}
```

Here `\expanded` is a macro defined as:

```
\def\expanded#1%
  {\edef\expandedsequence{\noexpand#1}%
                    \expandedsequence}
```

Hans Hagen

Mijn eigen voorkeuren waren het Kabel- en het Ondine logo, en het bestuur koos voor het laatste.

De kleur van het logo – als kleur werd gebruikt, b.v. op voorbedrukt briefpapier en voor de web-pagina's – zou identiek moeten zijn aan de MAPS-kleur. Omdat beeldschermen geen heldere blauw-groene kleuren kunnen produceren, werd de web-versie een veel stemmiger blauwgroen. Toen men die kleur zag werd besloten die ook voor gedrukte toepassingen te gebruiken. Voor de prepress-kenners: het Pantone-nummer is 549.

Op het moment van schrijven heeft het bestuur zich nog niet uitgesproken over een brief-stijl, maar als u dit leest heeft u waarschijnlijk al kunnen zien wat daar uit is gekomen.

Bij een logo kunnen letters het beste naar contouren worden omgezet. Daardoor hoeft het font niet beschikbaar te zijn bij het afdrukken van het logo. Veel tekenprogramma's kunnen dit en een logo in eps- (Encapsulated PostScript) formaat opslaan, en dat is wat ik heb gedaan. Voor niet-PostScript printers zal het logo voor een aantal resoluties worden omgezet naar bitmapped formaten.

Why `\expandafter` is sometimes needed by common users too.

1. When dealing with textual input, that is reading lines of text, TEX is nearly always expanding whatever it encounters. There are two primitives that can influence this proces: `\noexpand` and `\expandafter`. The first primitive will probably never surface in user input, but the latter may! Let's give an example. In most cases one will call for a new chapter by saying something like:

```
\Chapter{This Or That}
```

or

```
\Chapter{\ThisOrThat}
```

When `\ThisOrThat` has only one meaning and is never changed, this goes all right. The title is typeset and when called upon, it appears in the table of contents too.

2. Macro packages use auxiliary files to save entries of for instance tables of contents and indexes. When writing a chapter title to such a file, we can:

- expand the macros that are part of the title
- selectively expand those macros
- copy the title as is without expansion

The first alternative is the most simple. Hoewever, certain macros can expand into long sequences of tokens, that in the worst case are only partially expanded. Font switches are an example of such fragile commands.

A solution to this problem is partial expansion. This can be accomplished by preventing certain macros from expanding. This can be done quite easily at the macro programming level by preceding such commands by a `\noexpand` when the moment is there. But which macros need such precautions and which ones don't? And how is the user supposed to know this? Even worse, we expect users to explictly prohibit expansion for their own macros if needed, and in practice they end up with weird sequences like:

```
\Chapter{\dontexpand\ThisOrThat}
```

This leaves option three as the most save one. This method has one disadvantage. By copying the title verbatim, we got problems when `\ThisOrThat` is used in more chapter titles with different meanings. In such cases we have to expand `\ThisOrThat` on forehand:

```
\expandafter\Chapter\expandafter{\ThisOrThat}
```

The first `\expandafter` reaches over `\Chapter` and expands the next token, `\expandafter`. This second `\expandafter` reaches over the `{` and expands `\ThisOrThat`.

When we have a bit longer title, like the next one, we end up with a lot of `\expandafter`'s:

```
\Chapter{About \ThisOrThat}
```

Out of convenience, we prefer something like:

```
\expanded{\Chapter{About \ThisOrThat}}
```

Here `\expanded` is a macro defined as:

```
\def\expanded#1%
  {\edef\expandedsequence{\noexpand#1}%
                    \expandedsequence}
```

Hans Hagen

Bijlage 1
10 jaar NTG

Erik Frambach
Rijksuniversiteit Groningen
email: E.H.M.Frambach@eco.rug.nl

abstract
Een terugblik op tien jaar NTG, met een vooruitblik naar de komende jaren.

keywords
NTG, gebuikersgroep, lustrum

Inleiding

Tien jaar NTG, dat vraagt om een terugblik op wat er in die tijd allemaal gebeurd is, en een vooruitblik op wat we in de toekomst kunnen verwachten en waar we naartoe willen met de NTG.

Het is allemaal begonnen in juni 1988 in Groningen, toen Kees van der Laan, Gerard van Nes en Gerard Draaijer bij elkaar kwamen op de kamer van Kees in het Rekencentrum van de Rijksuniversiteit Groningen.

In het najaar van datzelfde jaar maakte de NTG zichzelf bekend aan de hele wereld. De eerste NTG-bijeenkomst was een feit en in 1991 konden de eerste officiële NTG-bestuursverkiezingen worden gehouden.

De doelstelling van de vereniging is vastgelegd in de statuten en luidt tot op heden onveranderd:

"*De vereniging heeft ten doel het bevorderen van de kennis en het gebruik van TEX, een internationale standaard voor het coderen van documenten die met behulp van computers worden samengesteld.*

De vereniging tracht haar doel te bereiken onder meer door:

- *de uitwisseling van informatie met betrekking tot de standaard als in het doel omschreven, door middel van woord, geschrift en met behulp van elektronische hulpmiddelen, zowel binnen de vereniging als met andere organisaties die een soortgelijk doel nastreven;*
- *organiseren casu quo stimuleren van congressen, tentoonstellingen en symposia, met betrekking tot bovengenoemde standaard;*
- *het onderzoeken van verbanden tussen bovengenoemde standaard en andere standaards op hetzelfde gebied;*
- *al hetgeen verder dienstig is aan de verwezenlijking van het doel.*"

Wat wel al snel veranderd is, is de naam van de vereniging. In eerste instantie was dat 'Nederlandse TEX-gebruikersgroep', maar al snel zag men in dat 'Nederlandstalige TEX-gebruikersgroep' veel beter past, omdat zo ook niet-Nederlanders toegang krijgen, met name Belgen natuurlijk. In 1993 leidde dat tot de benoeming van Philippe Vanoverbeeke tot België-commissaris.

Activiteiten

In de filosofie van de NTG heeft steeds centraal gestaan: het organiseren van bijeenkomsten en het uitgeven van het verenigingsblad MAPS, en dat alles op een low budget manier.

Daarnaast zijn regelmatig speciale bijeenkomsten georganiseerd en speciale MAPS-edities uitgegeven. Speciale bijeenkomsten waren bij voorbeeld cursussen, zoals die van David Salomon in 1992 die erg veel belangstelling trok. Maar ook de bijeenkomst met Donald Knuth als speciale gast in 1996 is het vermelden waard. MAPS-specials waren onder andere de hele dikke cursus-syllabus van Salomon, en de Frequently Asked Questions van onze zustervereniging UKTUG. En dat NTG ook internationaal een deuntje meeblaast mag blijken uit de organisatie van de Europese TEX-conferentie in 1995.

De internationale contacten die de NTG onderhoudt met vele andere TEX-gebruikersgroepen en personen leidt ook regelmatig tot uitwisseling van artikelen van en voor de MAPS. Uiteraard spelen de verschillende TEX-conferenties die her en der worden georganiseerd daarbij een belangrijke rol. Immers, persoonlijk contact is toch heel wat anders dan email of telefonisch contact. Vandaar dat de NTG altijd te vinden is op EuroTEX-conferenties, TUG-conferenties, maar soms ook op bijeenkomsten van Dante (Duitsland), UKTUG (Engeland), Gust (Polen) of CyrTUG (Rusland).

Ook mag niet onvermeld blijven dat in 1994 Kees van der Laan benoemd is tot erelid van de NTG. In 1996 is Johannes Braams eveneens erelid geworden, en in 1997 Piet van Oostrum, allen wegens bijzondere verdiensten voor de NTG en de hele TEX-wereld.

Tijdlijnen

In de afgelopen jaren hebben uiteraard erg veel mensen bijgedragen aan het succes van de NTG. In tabel 1 heb ik wat gegevens over NTG's eerste decennium op een rij gezet.

jaar	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997
ledental:	84	90	117	159	193	232	265	286	285	292
waarvan inst.leden:		15	21	28	30	33	36	33	32	33
MAPS-dikte:	40	120	264	278	342	478	410	682	332	467
bestuursleden:										
Kees van der Laan	v	v	v	v	v	v				
Johannes Braams	p	p	p	p	p	p	v	l		
Gerard van Nes	s	s	s	s	s	s	s	s	s	s
Huub Mulders	l	l								
Theo de Klerk		l								
Jos Winnink				l	l	l				
Theo Jurriens					l					
Erik Frambach						l	l	v	v	v
Wietse Dol							p	p	p	
Frans Goddijn							l	l	l	l
Hans Hagen									l	l
Taco Hoekwater									l	l
Wybo Dekker										p

v = voorzitter; s = secretaris; p = penningmeester; l = gewoon lid.

Daaruit is af te lezen hoe het ledental is verlopen, hoe hoeveel pagina's MAPS er geproduceerd zijn, en wie allemaal welke functies in het NTG-bestuur hebben vervuld.

Tabel 2 bevat een overzicht van alle NTG-bijeenkomsten, waar ze gehouden zijn en hoeveel deelnemers er waren.

Maar laten we niet te veel navelstaren. Om ons heen heeft de wereld ook niet stil gestaan. Gelukkig heeft de TEX-wereld daar steeds goed op ingespeeld.

Mijlpalen

In de afgelopen tien jaar zijn een aantal mijlpalen te onderscheiden die van wezenlijk belang zijn (geweest) voor de NTG en/of de hele TEX-wereld, of zelfs de hele computerwereld.

Voor de TEX-wereld was natuurlijk het feit dat Don Knuth in 1990 stopte met de ontwikkeling van TEX van groot belang. Daarmee was TEX feitelijk bevroren en dus stabiel.

Als belangrijkste (maar niet de enige) mijlpalen zou ik verder willen noemen SGML, LATEX 2.09/LATEX 2_ε, LATEX2html, BLUe, ConTEXt, PostScript, 4allTEX, emTEX, NTS, Babel, ϵ-TEX, Internet, PDF, PDFTEX, TDS, CTAN, NTG-stijlen, Nederlandse afbreekpatronen, FTP-servers, TEX-NL, TEX Live, FGBBS, WWW-servers, Omega en MetaPost. Bij veel van deze mijlpalen is de NTG of zijn NTG-leden direct betrokken (geweest).

Het is een hele waslijst, in tamelijk willekeurige volgorde. Wellicht weet niet iedereen precies wat al die kreten ze inhouden. Daarom geef ik hier een korte toelichting en probeer ze in verband te brengen met verschillende ontwikkelingen en trends in de laatste tien jaar.

SGML & Internet

SGML staat voor Structured Generalized Markup Language. Het is een internationale standaard voor het gestructureerd opmaken van documenten, waarbij 'opmaken' niet betekent vormgeven, maar juist aangeven wat de *betekenis* is van een bepaald stuk tekst, zoals titel, auteurs, samenvatting en secties. LATEX kent een vergelijkbare opbouw, maar SGML gaat in het structureren nog veel verder, terwijl LATEX zich ook bezig houdt met vormgeving. SGML is dus 'enkel' een beschrijvende taal. Voor het uiteindelijke zetwerk blijft een typesetting programma nodig. Dat kan heel goed TEX zijn, omdat TEX de noodzakelijke flexibiliteit en programmeerbaarheid heeft om zo'n klus te klaren.

HTML is een afgeleide van SGML, zeg maar het kleine broertje van SGML. Het wordt op heel grote schaal gebruikt op het World Wide Web (WWW), wat weer een onderdeel is van het Internet. Het Internet heeft de laatste jaren een stormachtige groei doorgemaakt wat ertoe heeft geleid dat iedere zichzelf respecterende orga-

jaar	aantal deelnemers	plaats
1988	26	Groningen, RUG
	55	Petten, ECN
1989	37	Leidschendam, PTT Neher Lab
	34	Tilburg, KUB
1990	42	Nijmegen, KUN
	38	Utrecht, Digital
1991	40	Amsterdam, Elsevier
	32	Eindhoven, TUE
1992	32	Amsterdam, CWI
	59	Meppel, Kon. Boom Pers
1993	55	De Bilt, KNMI
	72	Den Bosch, Océ
1994	40	Groningen, RUG
	55	Antwerpen, UIA (België)
1995	56	Twente, UT
	33	Arnhem, Papendal
1996	53	Amsterdam, Hogesch. A'dam
	46	Utrecht, RUU
1997	51	Delft, TUD
	40	Amsterdam, AMC

nisatie een WWW-site heeft opgezet. Zo ook de NTG: `http://www.ntg.nl` is het officiële adres van de NTG WWW-server. Al wat langer had de NTG een FTP-server draaiend. Dat is een eveneens via Internet bereikbare server van waaraf alle mogelijke TEX-gerelateerde software is te bemachtigen. Dat enorme archief heet CTAN, Comprehensive TEX Archive Network, een netwerk van servers over de hele wereld, zodat je waar ook ter wereld snel toegang hebt tot alle *public domain* TEX-software. Dat archief wordt door door een groep vrijwilligers uit verschillende landen onderhouden. De Duitse TEX-gebruikersgroep geeft het hele archief tegenwoordig ook uit op cdrom.

Een andere veel gebruikte vorm van Internet is email. Al vele jaren is in Nederland de discussielijst TEX-NL actief, waarop TEX-gebruikers hun ervaringen kunnen uitwisselen, problemen kunnen voorleggen en daar oplossingen voor bedenken. Naast TEX-NL zijn in Nederland thans verschillende andere lijsten geactiveerd. Internationaal zijn er nog veel meer. Ook 'Usenet' (`comp.text.tex`) mag niet onvermeld blijven als communicatiekanaal voor TEX-gebruikers.

En voor wie nog geen Internet-aansluiting heeft is het bulletin board FGBBS een uitkomst. Daar zijn voor gewone modem-gebruikers allerlei TEX-spullen te halen.

Internet & PDF

LATEX (zowel in de vorm van LATEX 2.09 als LATEX 2_ε) is wellicht het meest populaire macro-pakket voor TEX-gebruikers. LATEX2html is een programma dat het mogelijk maakt LATEX-documenten om te zetten naar HTML, wat noodzakelijk is om op WWW te publiceren. Een ontwikkeling die daarbij aansluit, maar vanuit een heel andere optiek is PDF. Dat staat voor 'Portable Document Format' en is bedacht door de firma Adobe. Documenten in PDF zijn eenvoudig in te zien met gratis 'viewers' (Adobe Acrobat of Ghostview) en af te drukken op bijna willekeurige printers. Het verschil met DVI is dat PDF volledig *self-contained*, dat wil zeggen alle fonts, plaatjes en wat dies meer zij zit in het bestand zelf. Dat maakt PDF tot een zeer geschikt formaat voor elektronisch publiceren. Ook daar is de TEX-wereld onmiddellijk op ingesprongen. PDFTEX is een variant op TEX die in plaats van DVI direct PDF kan genereren.

PDF & ConTEXt

Het macropakket ConTEXt maakt heftig gebruik van alle mogelijkheden die PDF biedt. Het pakket kan zo ongeveer alles wat LATEX kan, maar er ligt een heel andere filosofie aan ten grondslag. ConTEXt is in wezen monolitisch, terwijl LATEX meer modulair is. Dat wil zeggen: in ConTEXt is "alles" bij voorbaat ingebouwd, bij LATEX heb je voor "alles" een extra *style file* nodig. Met ConTEXt kunnen interactieve documenten gemaakt worden van een complexiteit waar op dit moment nog geen enkel ander pakket, binnen noch buiten de TEX-wereld, aan kan tippen.

Plain & BLUe

Voor wie al die extra toeters en bellen minder relevant zijn is er, naast Knuth's 'Plain TEX' het pakket BLUe. Plain TEX is *de* standaard zoals die door Knuth zelf is neergezet. Alle andere macropakketten bouwen min of meer door op die basis, zij het in verschillende richtingen. BLUe is een heel degelijk, doordacht en betrouwbaar systeem gebaseerd op Knuth's concepten, dat zich met opzet beperkt tot de wezenlijke zaken van tekstverwerken. Zaken die in veel andere pakketten via externe programma's gerealiseerd moeten worden kan BLUe helemaal binnen TEX afhandelen. Dat geldt onder andere voor het verwerken van bibliografieën, voor indexen en database-functies.

Nederlandstalige TEX

Uiteraard heeft de NTG zich vanuit haar doelstellingen bezig gehouden met het ontwikkelen van Nederlandse afbreekpatronen voor TEX. Met de nieuwe spellingsregels

van enkele jaren geleden is dat initiatief weer nieuw leven ingeblazen, zodat we nu kunnen beschikken over een uitmuntende set patronen voor het Nederlands. Samen met de 'NTG-stijlen' (LaTeX-stijlen voor artikelen, rapporten en boeken) is dat een ijzersterke combinatie. De NTG-stijlen blinken met name uit door soberheid en strakheid. Voor Nederlanders (en wellicht vele andere Europeanen) ziet dat er een stuk aantrekkelijker uit dan de Amerikaans georiënteerde standaard LaTeX-stijlen.

Al snel werd duidelijk dat voor het Nederlands en vele andere (Europese) talen specifieke aanpassingen nodig zijn aan LaTeX om prettig te kunnen werken. Uiteraard zijn er in iedere taal andere uitdrukkingen voor bij voorbeeld 'Hoofdstuk', maar ook subtielere verschillen zoals de manier waarop aanhalingstekens worden gezet moeten netjes en eenduidig geregeld worden. En dat is precies wat Babel doet. Via Babel kan in een LaTeX-document vloeiend geschakeld worden tussen stukken in allerlei verschillende talen. Babel zorgt dat alle taal-specifieke zaken vanzelf goed worden ingesteld.

PostScript & MetaPost

PDF heb ik al genoemd, maar ook PostScript moet genoemd worden. De printertaal PostScript is voor professionals al erg lang *de* standaard. Tot enkele jaren geleden was die voor gewone stervelingen echter onbetaalbaar. Maar dankzij steeds goedkopere printers, en ook dankzij gratis PostScript interpreters als Ghostscript is PostScript tegenwoordig 'voor de massa'. Dat maakt het plotseling veel gemakkelijker om relatief ingewikkelde typografische dingen te doen.

MetaPost maakt dankbaar gebruik daarvan. Het is een variant op Knuth's Metafont, het font-generatieprogramma. MetaPost is echter niet gericht op fonts, maar op tekeningen, die in uiterst compacte PostScript-code worden uitgevoerd. Dankzij de naadloze samenwerking met TeX zijn hiermee bijzonder fraaie resultaten te bereiken.

Opvolgers voor TeX

Ofschoon TeX zelf door Knuth bevroren is gaan de ontwikkelingen door. ϵ-TeX is een extensie van TeX die volledig compatibel is de originele TeX, maar bevat een aantal extra's die het leven voor programmeurs en typografisch veeleisende mensen een stuk aangenamer kan maken. Omega is ook een variant op TeX, maar gaat veel verder. Het werkt intern volledig met 16-bits *integers*, terwijl de originele TeX 8-bits is. 8-bits wil hier zeggen: er zijn bij voorbeeld 'slechts' 255 counters in TeX, terwijl Omega er 65535 heeft.[1] Die 16-bits van Omega maakt het programmeren een stuk eenvoudiger doordat niet meer op elke bit hoeft te worden bezuinigd, maar nog veel belangrijker is dat Omega volledig in Unicode draait. Unicode is de 16-bits standaard voor font-tabellen, waarin ruimte is gereserveerd voor alle talen ter wereld. Met Omega is het daardoor eenvoudig om desgewenst Nederlands, Russisch, Chinees en Arabisch door elkaar te gebruiken. Zelfs als de ene taal van links naar rechts loopt, de ander van rechts naar links en nog een ander van boven naar onder.

NTS is een parallelle ontwikkeling van een opvolger voor TeX. De afkorting staat voor New Typesetting System en is op dit moment nog in de ontwerpfase. Maar er zijn plannen om dit jaar met een eerste implementatie te beginnen. NTS zal in de gloednieuwe en veelbelovende programmeertaal Java geschreven worden. Een belangrijk kenmerk van Java is (of zou moeten zijn) dat het platformonafhankelijk is. Dat wil zeggen: Java-TeX zou zonder aanpassingen meteen op elk type computer kunnen draaien, mits daarop Java draait. Dat is iets waar Knuth destijds bijzonder veel moeite voor heeft moeten doen. Ik ben benieuwd wat er van de belofte van Java uitkomt.

Installatie & distributie

TeX heeft de laatste jaren een belangrijke ontwikkeling doorgemaakt op het gebied van installatieprocedures. Voorheen was het een zaak voor zeer deskundige systeembeheerders, dan wel geduldige en frustratiebestendige gebruikers, om een TeX-systeem te installeren. Dankzij de opkomst van de cdrom is dat veranderd. NTG's 4allTeX cdroms voor MS-DOS en MS-Windows-gebruikers heeft de wereld laten zien dat het mogelijk is TeX in een minuut te installeren, bij wijze van spreke met een druk op de knop. Dat heeft beslist geholpen bij het populariseren van TeX. Immers, TeX is hiermee voor een deel van zijn elitaire, academische, ivoren-toren-reputatie afgeholpen. De 4allTeX cdroms hebben dankbaar gebruik gemaakt van allerlei *public domain* software, waarvan emTeX wellicht de belangrijkste is. Deze zeer goede gratis TeX-implementatie voor MS-DOS en OS/2 heeft TeX een enorme duw in de rug gegeven.

Enkele jaren later volgde ook de Unix-wereld met de TeX Live cdrom. De Unix- en Microsoft-wereld zijn op TeX-gebied nu dichter bij elkaar gekomen dan ooit: ze kunnen nu dezelfde TeX-implementatie draaien (Web2c) met dezelfde format-files, en dezelfde sources, mede dankzij het TDS-initiatief. TDS staat voor TeX Directory Structure. Het definieert een standaardmanier om alle bestanden die in een TeX-installatie voorkomen in directories op te slaan. Zo'n TDS maakt het dus relatief gemakkelijk om

1. Niet verwarren met 16-bits of 32-bits applicaties: van zowel TeX als Omega zijn 16-bits, 32-bits en 64-bits applicaties beschikbaar, maar dat heeft alleen te maken met het besturingssysteem waarop ze draaien.

(delen van) een TEX-installatie toe te voegen, te verwijderen, te verversen of uit te wisselen met anderen.

Uit het zicht

Niet alle nieuwe ideeën, hoe goed ze ook (bedoeld) waren, zijn uitgegroeid tot belangrijke geaccepteerde standaards. Ook in de TEX-wereld gaat wel eens iets de mist in.

Knuth's Metafont heeft het als font-ontwerpprogramma nooit echt gemaakt. Waarschijnlijk zijn fontontwerpers toch meer visueel dan wiskundig ingesteld, waardoor Metafont een te grote overgang zou betekenen. Echter, in de vorm van MetaPost begint Metafont (althans de programmeertaal) een tweede jeugd als geavanceerd tekenprogramma. Het blijft specialistisch gereedschap, maar nu wellicht voor een grotere groep gebruikers.

Knuth's *literate programming* ideeën zijn ook nooit op grote schaal toegepast. De kansen daarop worden alleen maar kleiner nu veel programmeurs werken met geïntegreerde *work benches*, RAD-tools (*Rapid Application Development*), en vierde-generatie programmeertalen. Daarin is nauwelijks plaats voor literaire uitingen: die beschouwt men al gauw als overbodige luxe — geheel ten onrechte, zoals het jaar-2000-probleem weer eens aantoont.

Grote macropakketten geaccepteerd krijgen valt ook niet mee. Voorbeelden van pakketten die het nooit echt gemaakt hebben zijn LAMSTEX en Lollipop. Ik ben benieuwd hoe het ConTEXt zal vergaan. Daarvan is in ieder geval duidelijk dat het functionaliteit biedt die tot nu toe met geen enkel ander pakket redelijkerwijs te realiseren is. Maar of dat genoeg is om met name LATEX-gebruikers over te halen? LATEX-gebruikers wachten eigenlijk al jaren op LATEX 3 en de vraag is hoe lang ze daar nog op willen wachten, en of het reëel is daarop te wachten.

De wereld rondom

Parallel aan de ontwikkelingen aan het TEX-front werd natuurlijk ook druk gewerkt aan andere tekstverwerkers, waarvan we er verschillende hebben zien komen en gaan. Voor veel gebruikers is de ontwikkeling van Microsoft Windows erg belangrijk, en de ontwikkeling van grafische gebruikersomgevingen in het algemeen. Zeker de thuisgebruikers verwachten tegenwoordig dat een programma snel en eenvoudig te installeren is en meteen voor 100% werkt. Massa's MS-Windows programma's hebben laten zien dat dat kan (denk bij voorbeeld aan MS-Word en WordPerfect).

Ook niet onbelangrijk is het feit dat computers sterk zijn blijven groeien in reken- en geheugencapaciteit. De beperkingen waar Knuth bij het ontwerpen van TEX rekening mee hield zijn nu haast lachwekkend. Dat is voor bepaalde toepassingen zeker belemmerend, maar laten we niet vergeten dat TEX door zijn bevroren toestand nu waanzinnig snel draait in vergelijking met tien jaar geleden. Dat kan van programma's als WordPerfect niet gezegd worden...

Die overdaad(?) aan computerkracht kan en zal natuurlijk opgesoupeerd worden door nieuwe toepassingen. Het blijft koffiedik-kijken maar enkele trends zijn duidelijk te zien, en van daaruit kunnen enkele voorspellingen worden gedaan.

De toekomst

Het papierloze kantoor is er nog steeds niet, maar er wordt toch al steeds meer elektronisch gepubliceerd. Kennelijk hebben we eerder te maken met een evolutie dan een revolutie.

Internet/intranet en PDF hebben daar natuurlijk flink aan bijgedragen, omdat die samen een hoop papier kunnen vervangen.

Wellicht zien we in de nabije toekomst (de komende vijf jaar) dat "vluchtige" publicaties, dat wil zeggen publicaties die snel hun (nieuws-) waarde verliezen (zoals kranten en artikelen) steeds meer alleen nog elektronisch beschikbaar zijn. Voor echte boeken zal altijd een markt blijven, maar ze zullen exclusiever worden. *Publishing on demand* zal vast een belangrijkere rol gaan spelen. Via Internet en met bij voorbeeld PDF en TEX als *typesetting engine* is zoiets met relatief weinig moeite te realiseren. Echter, die markt moet nog aangeboord worden, en of die economisch rendabel is moet nog afgewacht worden.

Verspreiding van brontekst zal steeds minder plaatsvinden, domweg omdat er minder behoefte aan is. Er blijven echter situaties waarin dat wel wezenlijk is. Dat zijn denk ik met name professionele toepassingen zoals wiskundige teksten waarin linearisering van de meest complexe formules altijd mogelijk is, en noodzakelijk.

De nieuwste hype is natuurlijk 'multi-media'. Iedereen is ermee bezig, of doet alsof. Moderne computers zijn krachtig genoeg om video-filmpjes tekst en geluid simultaan weer te geven. Maar als iedereen eenmaal moe is van alle overbodige *special effects* kunnen we pas serieuze toepassingen verwachten. De meeste multi-media-presentaties die ik tot nu toe heb gezien bieden enkel meer van hetzelfde op een andere manier. Dat is geen vooruitgang.

Wat we zeker zouden willen, liefst op korte termijn, is de mogelijkheid om eenvoudig en betrouwbaar met stemgeluid tekst in te voeren. En het zou natuurlijk ook erg mooi zijn als de computer een tekst goed kan voorlezen.

Aan de uitvoerkant kan trouwens nog veel meer gebeuren. Met *virtual reality*-technieken zou het mogelijk moeten zijn door virtuele bibliotheken te lopen en op lichtsnelheid documenten te zoeken.

Maar ook in de reële wereld zullen we nog belangrijke ontwikkelingen meemaken in de komende tien jaar. Wat

dacht u van een computerscherm (als ik dat zo mag noemen) op A4-formaat, 1 cm dik, 300 gram, en met een resolutie van 300 dpi? Zo wil ik best in mijn luie stoel een "boek" lezen. Uiteraard staat dat scherm direct (draadloos) in verbinding met de rest van de wereld, zodat ik op afroep een artikel uit de MAPS te voorschijn kan toveren.

Uiteraard moeten nog vele bronnen ontsloten worden voordat we echt alles via computerverbindingen kunnen bereiken. Ook zal er nog veel meer dan voorheen elektronisch opgeslagen moeten worden. Opslagcapaciteit wordt gelukkig steeds goedkoper zodat daar niet gauw problemen te verwachten zijn. De rekenkracht van computer neemt ook nog steeds sterk toe, en is dat is ook hard nodig om al die bronnen te kunnen beheren en doorzoeken.

Zoals eerder gezegd, *publishing on demand* kan een hoge vlucht gaan nemen, mede dankzij veel krachtigere hardware. Die hardware maakt het ook mogelijk dat TEX-achtige systemen min of meer WYSIWYG (*What You See Is What You Get*) worden. Immers, het formatteren van een heel boek kost straks nog maar een fractie van een seconde. Dat opent perspectieven voor initiatieven als NTS. Waar TEX zich nu beperkt tot optimalisatie van alinea's en (tot op zekere hoogte) pagina's, zou NTS *spreads* (tegenover elkaar liggende pagina's) als geheel kunnen optimaliseren. Nog beter: alle pagina's zouden ten opzichte van elkaar geoptimaliseerd kunnen worden. Waarbij wellicht rekening moet worden gehouden met het uiteindelijke medium: op mijn A4-schermpje komt het waarschijnlijk niet zo nauw, maar voor een fraai gezet 'echt' boek wel degelijk. Vakmanschap blijft altijd noodzakelijk.

De rol van de NTG

En welke rol speelt de NTG in deze ontwikkelingen? Uiteraard blijft de NTG alle TEX-gebruikers ondersteunen, of ze nou TEX gebruiken als hun persoonlijke tekstverwerker of als hun professionele *back end typesetter*. De NTG houdt de vinger aan de pols en probeert interessante nieuwe ontwikkelingen te stimuleren. Dat kan op verschillende manieren. Sommige projecten hebben geld nodig (waarvan de NTG natuurlijk niet echt veel heeft), andere zijn juist meer gebaat met promotie en energie van meerdere enthousiastelingen. De NTG kan ook voor faciliteiten zorgen, zoals Internet discussielijsten, snelle toegang tot CTAN en WWW. Initiatiefnemers zijn natuurlijk altijd van harte welkom als ze een artikel over hun bezigheden willen publiceren in de MAPS. Het spreekt vanzelf dat de NTG bijeenkomsten blijft organiseren waarop alle geïnteresseerden actuele onderwerpen kunnen bespreken en/of interessante toepassingen demonstreren. Ook blijft de NTG actief in het aanbieden van belangrijke software aan haar leden (onder andere 4allTEX, TEX Live en CTAN cdroms).

Verder kan de NTG proberen nieuwe doelgroepen aan te spreken. Als TEX dan nu eindelijk geschikt is voor een breder publiek, dan moeten we daar ook op inspringen. Daarbij kunnen we denken aan:

- Middelbare scholen: die hebben typisch een vreselijk klein budget voor computerfaciliteiten, dus een gratis programma als TEX zou erg moeten aanspreken.
- Hogere scholen: het zou mooi zijn als TEX in het curriculum wordt opgenomen, als zijnde een stuk basisgereedschap waar je je leven lang plezier van kunt hebben.
- Taleninstituten: TEX (en Omega nog meer) is erg sterk in het verweven van teksten in meerdere talen in één document.
- Uitgevers: voor *publishing on demand* is TEX een erg krachtig stuk gereedschap, wellicht de enige reële oplossing. Ook als *back end* voor SGML zou TEX een veel grotere (zij het minder zichtbare) rol kunnen spelen.
- Computer-programmeurs: zij zouden op de hoogte moeten worden gebracht van de kracht van *literate programming*. Dat TEX hun zal aanspreken is zowiezo erg waarschijnlijk.

Veel energie zal moeten worden gestoken in promotie. TEX is nog steeds relatief onbekend (zeker buiten academische en vooral natuurwetenschappelijke kringen), dus moeten mensen eerst ervan overtuigd worden dat het zinvol is om in TEX te investeren. Dat kan onder andere door in de praktijk te laten zien welke voordelen ermee te behalen zijn. En als mensen overtuigd zijn zullen we ze de noodzakelijke ondersteuning moeten bieden. Met goede cursussen, toegesneden op de behoeften van de gebruikers, moet dat mogelijk zijn. Kortom, de NTG heeft ook de komende tien jaar genoeg werk te doen. Het gonst binnen NTG-kringen nog steeds van enthousiasme en activiteiten, en dat geeft veel vertrouwen in de toekomst.

Tot slot wil ik iedereen bedanken die zich voor de NTG op wat voor manier dan ook heeft ingezet. Zonder die inzet waren we nooit zo ver gekomen. Laten we proberen de volgende tien jaar net zo succesvol te volbrengen!

Bijlage 2

EuroTEX'98 in Saint-Malo, France

Erik Frambach,
Hans Hagen,
Taco Hoekwater,
Siep Kroonenberg

abstract

This year's EuroTEX conference was held in Saint-Malo. On behalf of the NTG Erik Frambach, Hans Hagen, Taco Hoekwater and Siep Kroonenberg represented the Dutch TEX-community. In this report they discuss the social and TEXnical aspects of the conference.

keywords

EuroTEX, conference, Saint-Malo

At the EuroTEX conference, the Netherlands was represented by Erik Frambach, Hans Hagen, Taco Hoekwater and Siep Kroonenberg. There was also a rather large Belgian delegation, consisting of Marc Baert, Gontran Ervynck, Erik van Eynde en Bar van Maele.

The trip to Saint-Malo

We traveled by train via Rotterdam, Brussels and Paris. From Dordrecht on, the party was complete. In 'Bruxelles Midi' we were to change to the Thalys, the new French high-speed train. Assuming that Bruxelles Midi meant Brussels Central, we got off the train there, only to find out that Bruxelles Midi meant Brussels South. This meant that we would miss our connection. Fortunately we could afford to lose one or two hours so we didn't loose hope. However, upon arrival at Brussels South it turned out that it was not possible to simply board the next Thalys. In fact, we had to buy entirely new tickets: our tickets had been valid only for the particular train we originally intended to take. By the time we were in the possession of new tickets, we had missed another train, and the next one would not leave for another two hours. In Paris, we had to travel by subway from Gare du Nord to Gare Mont Parnasse where we needed to buy new tickets once more, and had another lengthy wait.

What happened at the Rennes station

On arrival at Rennes, the station looked deserted. However, there were lots of vending machines for tickets. We all tried our credit cards on various machines, with little luck, until we finally found a machine which accepted Hans's card. It was a slow business: each ticket had to be purchased separately, and the machine took ten minutes per ticket. At the fourth and last ticket, the machine failed, and Hans didn't get his card back. On the floor above, a window turned out to be open, but it was not easy to explain our plight. 'La machine aux billets a mangé notre carte de crédit' was eventually understood, and the credit card was eventually retrieved from the machine.

In Rennes, we ran into a Norwegian colleague, Dag Langmyr. He had tickets which were probably more expensive than ours, but whose validity was not restricted to one particular train. Meanwhile, it had turned so late that we needed to call our hotels about our delayed arrival. Again, after trying lots of phone boots, none of which would accept either coins or our credit cards. We finally make our phone calls at a snack-bar, where we enjoyed a simple but tasty meal of French bread sandwiches.

Saint-Malo is a seaside resort, but this early in the season it is still quiet enough. The city center is called Intra Muros. Most of it has been destroyed at one time or another, and rebuilt later; according to Taco with some romantic embellishments which weren't in the original plan. The ramparts and the beach were perfect for walks during free hours. But all this we were to discover later.

The courses

On Sunday April 29 EuroTEX participants could attend one of three courses. There was a course called 'Comment débuter avec LATEX' by Michèle Jouhet, a course called 'Approfondir LATEX' by Éric Picheral, and a course called 'Producing advanced PDF documents with TEX' by Hans Hagen.

We all went to Hans's course, simply because our knowledge of French is insufficient for the other courses, and because it seemed to be more suitable given our current expertise. The other advanced course, by Philip Taylor, had been cancelled at the last moment, so Hans enjoyed the participation of a rather big audience.

Hans showed some impressive things you can do with PDF. His 'ConTEXt' format makes it quite easy (or so it seemed) to produce very complex documents with hyperlinks all over the place, multiple indexes, navigation buttons and other fancy gadgets.

The program

On Monday the conference was officially opened by GUTenberg's president Michel Goossens. He informed us that there were 112 participants from no less than 17 countries. Even people from countries as far as Australia and Japan were present.

The first morning's theme was 'Fonts, maths & encoding'. The first lecture was by the typographer Richard Southall, who explained to us how he designed a telephone-directory using TEX & Metafont. His challenge was to save ink and paper, so the phone book uses a five-point typeface, which is still surprisingly legible. He also explained the rationale behind the selection and organization of the data.

It was funny to see how Southall designed a font on a grid and next would approximate this design by Metafont code. Naturally a few pixels would be off, but after a few rounds of corrections the typeface would be finished.

Next the Euromath system, a structured editor for mathematicians, was introduced by Janka Chlebíková. That system makes it easier for mathematicians to input their mathematical articles and reports. Unfortunately the system only runs on Sun systems.

M. Clasen and Ulrik Vieth introduced a new 8-bit math font encoding scheme. The desirability of this new math encoding was convincingly demonstrated. Just to mention a couple of idiosyncrasies of the old encoding: the equal-sign = doubles as component for double arrows, and whereas the math italic encoding contains a full set of lower case Greek characters, it only contains a partial set of uppercase Greek characters. Elsewhere in this MAPS issue you can read more about this subject.

Two other lectures were organized this morning. Sasha Berdnikov presented an encoding paradigm for LATEX and the projected X2 encoding which he hopes will replace the abundance of encodings for Russian (Cyrillic) characters now in use. In the final lecture Hirotsugu Kakugawa introduced VFlib, a general programming library that supports multiple font formats.

Parallel to EuroTEX the RIDT'98 conference was organized in the same building. This provided us with the opportunity to meet John Hobby, the author of MetaPost. A nice surprise which gave some of us a chance to talk about future developments in MetaPost.

After a delicious lunch we were ready for the afternoon sessions. The first part this session was devoted to 'PDF and tools'. Hàn Th´ê Thành showed us new developments regarding his PDFTEX program. Sergey Lesenko, the next speaker, showed us another approach for generating PDF: his program DVIPDF is somewhat similar to DVIPS. It takes a DVI file as input and converts it to PDF. Unfortunately, we still haven't seen any executables—it seems that this program might spend the larger portion of it's life as 'alpha software'.

Next, Thomas Esser discussed the concepts of installation, configuration and maintenance of the teTEX system. Interesting news from Thomas was that the next version of teTEX (0.9) will include the win32 executables as an integral part of the system. Surely this will make the Web2c distribution for win32 both simpler to install and easier to configure.

The next presentation was about Xindy. Roger Kehr held an inspiring talk about this brand new program for generat-

ing indexes. It's suite similar to MakeIndex, but a lot more powerful and flexible. It handles different input encodings and sorting algorithms much better than MakeIndex. It also comes with a couple of tools that hide all of the new extensions behind a traditional MakeIndex interface, so switching to the new program should be no problem at all.

After the tea break we focused on 'TEXnics'. The first lecture was given by Sasha Berdnikow. He discussed some problems with accents in TEX: letters with multiple accents and accents varying for uppercase/lowercase letters.

Next a presentation about fancyvrb was scheduled. This LATEX-package provides an improved version of the traditional `verbatim`-environment. Sebastian Rahtz demonstrated some of the fancy tricks it can do.

The remaining sessions in this afternoon were mainly of interest to the French members of the audience. Bernard Gaulle talked about the french extension in LATEX, and how one can personalize it and a new French dictionary for ISPELL was presented by Christophe Pythoud.

Figure 1. The celebration cake. Photograph taken by Gyöngyi Bujdosó

After this long day there was a meeting of the members of GUTenberg. In the evening drinks were served because of GUTenberg's 10th anniversary, and a special dinner was prepared. During and after dinner a musical ensemble called the 'Free Donkeys' performed nice folk songs. A few members of GUTenberg and a few others entertained the audience with thoughts and stories about 10 years of GUTenberg. The desert was very special: a birthday cake of one square meter! The atmosphere became even better when we all had a glass of vodka donated by the Russian TEX users group.

After this celebration some of us went to an Irish(!) pub in town, and guess what, after half an hour or so many other TEXies joined the club. As if they could smell where the fun is. Needless to say that we had a good time, discussing TEX and many other things.

On Tuesday morning we started of with the 'Editorial chain'. Marcia Bossy introduced us to WWW-TED, an evolutionary and dynamic thesaurus for HTML and Marie-Louise Munier gave us some insight in his experience using TEX (LATEX) in the editorial chain.

Figure 2. Photograph taken by Gyöngyi Bujdosó

Petr Sojka described a digitization project that involved creating a PDF version of the Czech Otto encyclopædia, which is a 28-volume monster dating from 1888–1908. The goal was an exact replication of the page layout, including line breaks, and the addition of hypertext features. The digitization process started out with OCR, and markup was added in several, largely automated stages. An impresive project, that once again shows the almost unlimited applicability of TEX.

After the coffee break Robert Sutor showed the possibilities of IBM's Techexplorer as an online publishing tool. Next, B. Bachimont demonstrated 'PolyTEX', but since this lecture was in French we had to skip it and can't tell you what it was about. Even the title of the lecture was incomprehensible to us. The same problem occurred with Jean-Daniel Fekete discussion of something called TEI.

After one more delicious lunch people gathered in 'Birds of a feather' (BOF) sessions. Themes were: LATEX2html, Xindy, and math font encodings/symbols. As usual, the BOFs hardly functioned as expected: The LATEX2html BOF never really started because there was only one participant, the Xindy BOF was more like a 'feature request meeting', and the math fonts BOF went completely out of control (they actually talked for about 6 hours during the BOF, during the following lectures, during dinner and in the pub).

'XML' was the next topic to be discussed. Again, Michel Goossens gave a lecture, this time about XML, the extens-

ible markup language that may well be the successor of HTML, and the future of the web. The final speaker was Murray Maloney who gave us some insight on what is happening in the XML world and what we can expect from it in the (near) future.

Representatives of all BOFs summarized what was discussed in their sessions and which conclusions had been reached. After that, Michel Goossens gave a presentation on publishing scientific documents on the web. He discussed the strategies used by his employer, CERN—the birthplace of the world wide web—, for posting scientific papers on the web. They have to process massive amounts from various sources. Manual intervention should therefore be minimized. No single solution worked in all cases; partial solutions involve PDF, various types of conversion from TEX to html, and on-the-fly HTML generation. Incidentally, the presentation gave a very nice comparison between the various TEX to HTML converters that are available, showing their weak points as well as their strong points.

Then the conference was officially closed by GUTenberg's president Michel Goossens. As usual there was an election of the best lecture of the conference. The presentation on Xindy by Roger Kehr was the winner. We think he deserved it.

Food and drinks

With every daily lunch we were treated to a four-course menu consisting of a starter, main course, cheese and dessert, served with both water and two different wines. After the first lunch we understood that the water was absolutely essential to limit the amount of alcohol intake, and also that it was unwise to finish each plate if you wanted to reach the end of the meal. By the way, all of the food was very good.

The social events

You may argue that EuroTEX by itself is one big social event, but anyway, part of the conference was a guided tour to either the Mont-Saint-Michel or the library of Avranches. Both are quite famous.

Mont-Saint-Michel

The Mont-Saint-Michel is a steep offshore mountain that harbors an old cloister. While driving to the Mont by bus we crossed many picturesque villages. The Mont-Saint-Michel itself was reached by a small road that crosses a long muddy beach that floods at high tide.

The main part of the visit was spend in the cloister, where we were told repeatedly that in fact many cloisters were build upon each other. It's hard to imagine how such a large and heavy building was constructed, but the peaceful tiny gardens and silent places from which one could oversee the ocean, probably compensated for the hard labor.

Figure 3. Le Mont-Saint-Michel.

When I compare my french dictionary with the english one, I always thought that the french needed less words than the english to explain things. When however our guide switched to english, she needed about half the time she spent on french, so we probably missed some explanations. Nevertheless, the trip was worth attending.

Avranches

Wednesday mornings are usually rather ordinary mornings, but not on that day. The buses left at 08.00AM sharp (or so we were led to believe), so we had to get up at seven o'clock to make sure that there was enough time to have breakfast and pay our hotel-bills! After a bus trip of about 90 minutes we arrived at Avranches at precisely 10.15AM (yes, this implies that the bus did *not* leave at 08.00AM sharp).

Afterwards, it was more than worth the hassle. Not only is the Avranches Library one of the most impressive libraries in the world, but it also has a museum annex workshop attached to it. This gave us the opportunity to look at and compare various kinds of real (newly created) parchments, learn a bit about how the monks lived who did the 'manu-scripting', and look at the various kinds of poison that these people considered 'paint'!

Also, there was a guided tour in the library itself, but unfortunately this was completely in French. Probably this came about because our visit took place during the off-season and they couldn't find somebody that spoke English. Really a pity, but at least now we have a valid excuse to come back one day. One thing that became very clear looking at the exposition is that the au-

thors of those mediæval manuscripts could have gained a lot from using TEX: the average paragraph starred 'Overfull Boxes' on just about every second line.

Figure 4. The cloister that houses the library of Avranches.

Going home

Taco, Hans and Siep took the train back to Holland on Wednesday, and this time everything went remarkably smoothly. Erik, however, was tempted into another adventure. The Polish delegation had asked him if it were possible to stay overnight at his place, so they could split their journey back home (2000 km by car) in two equal parts. Naturally he said yes and in spite of lots of rain and fog we managed to reach Groningen at 5AM.

Conclusion

Though it was a very packed conference (only three days including the tutorials) it wasn't overfull. The organization was very well done, the accommodation was very good and the enthusiasm of the people attending the conference was high. All in all something to remember.

Expansion, what is that?

Let's start with a very plain TEX file:

```
Hi there! \end
```

When TEX reads this one line file, it typesets the words *Hi there!* and then ends the session. Here `\end` is not read as text to be typeset, but as a directive that tells TEX to do something special. By default, this sequence of characters equals the built in primitive with the meaning *end this session*.

One cannot only call for build in primitives, but also define his or her own macros. So we've got primitives and macros!

```
\def\name{Hans} Hi there \name ! \end
```

Here we define a macro `\name`, that is said to expand into `Hans`. In most cases, one will use macros for defining more complicated things, but the principles remain the same.

In this small file, after accepting the definition of `\name`, TEX inserts the two words *Hi there* plus a space. Next it meets `\name` and looks up its meaning. This macro expands into *Hans*, and after typesetting this word, TEX reads on and sees the exclamation mark. Finally `\end` ends the session.

Here expansion means as much as *insert the meaning of a macro* and when doing so, *expand all macros that are part of this macro*, and so on. Such a macro can contain text or a program, which is a sequence of TEX primitive operations, macros and/or text. Often those macros influence the typesetting process.

So in fact TEX is constantly grabbing text and expanding macros. Although there is no sharp border between those activities, one should be aware that TEX is constantly switching between typesetting text, interpreting character sequences, and executing the programs laid down in macros.

I have to admit that this is not the whole truth. Typesetting is a multi——step process in which TEX is doing quite a lot. In the previous example TEX first scans a paragraph, then determines the best line breaks, and finally adds the results to the current page, or in TEX terms: the main vertical list. Then TEX looks if and how the content of the current page should be split over more pages. If indeed a full page can be split off, TEX calls the appropriate macros to complete the page, that is, add a page number, headers and footers, flush floating bodies, add footnotes, do some housekeeping etc. etc.

(Continued on page 43)

thors of those mediæval manuscripts could have gained a lot from using TEX: the average paragraph starred 'Overfull Boxes' on just about every second line.

Figure 4. The cloister that houses the library of Avranches.

Going home

Taco, Hans and Siep took the train back to Holland on Wednesday, and this time everything went remarkably smoothly. Erik, however, was tempted into another adventure. The Polish delegation had asked him if it were possible to stay overnight at his place, so they could split their journey back home (2000 km by car) in two equal parts. Naturally he said yes and in spite of lots of rain and fog we managed to reach Groningen at 5AM.

Conclusion

Though it was a very packed conference (only three days including the tutorials) it wasn't overfull. The organization was very well done, the accommodation was very good and the enthusiasm of the people attending the conference was high. All in all something to remember.

Expansion, what is that?

Let's start with a very plain TEX file:

```
Hi there! \end
```

When TEX reads this one line file, it typesets the words *Hi there!* and then ends the session. Here `\end` is not read as text to be typeset, but as a directive that tells TEX to do something special. By default, this sequence of characters equals the built in primitive with the meaning *end this session*.

One cannot only call for build in primitives, but also define his or her own macros. So we've got primitives and macros!

```
\def\name{Hans} Hi there \name ! \end
```

Here we define a macro `\name`, that is said to expand into `Hans`. In most cases, one will use macros for defining more complicated things, but the principles remain the same.

In this small file, after accepting the definition of `\name`, TEX inserts the two words *Hi there* plus a space. Next it meets `\name` and looks up its meaning. This macro expands into *Hans*, and after typesetting this word, TEX reads on and sees the exclamation mark. Finally `\end` ends the session.

Here expansion means as much as *insert the meaning of a macro* and when doing so, *expand all macros that are part of this macro*, and so on. Such a macro can contain text or a program, which is a sequence of TEX primitive operations, macros and/or text. Often those macros influence the typesetting process.

So in fact TEX is constantly grabbing text and expanding macros. Although there is no sharp border between those activities, one should be aware that TEX is constantly switching between typesetting text, interpreting character sequences, and executing the programs laid down in macros.

I have to admit that this is not the whole truth. Typesetting is a multi——step process in which TEX is doing quite a lot. In the previous example TEX first scans a paragraph, then determines the best line breaks, and finally adds the results to the current page, or in TEX terms: the main vertical list. Then TEX looks if and how the content of the current page should be split over more pages. If indeed a full page can be split off, TEX calls the appropriate macros to complete the page, that is, add a page number, headers and footers, flush floating bodies, add footnotes, do some housekeeping etc. etc.

(Continued on page 43)

tention, some colors will have traveled from rgb to cmyk back to rgb and again to cmyk and will be much the worse for wear ;-).

Further reading

For online documents, dates and exact locations are omitted, because these are subject to too much change.

Adobe Systems, PostScript Language Reference Manual, second edition, Addison-Wesley 1990.

Adobe Systems, Portable Document Format Reference Manual. Available online from `www.adobe.com`.

Adobe Systems, PDF for Prepress Workflow and Document Delivery. Available online from `www.adobe.com`. Various other documents of interest on pdf, PostScript Level 3 and color management can be found at this site.

David Carlisle and Sebastian Rahtz, Graphics package. Available from CTAN and included in most TEX/LATEX distributions. This is in docstrip format; just run `latex graphics.dtx` to get formatted documentation.

Michel Goossens, Sebastian Rahtz and Frank Mittelbach, The LATEX Graphics Companion, Addison Wesley, 1997.

Thomas Rokicky, Dvips manual. Included in the dvips distribution; available from CTAN and included in most TEX distributions.

Postprocessing software vendors, see e.g. `www.lantanarips.com` for Crackerjack, and `www.imation.com` for Trapwise and Presswise.

Expansion, what is that? (continued)

Although from the users point of view, TEX expands whatever macro it meets while reading the paragraph, in practice expansion can be postponed of prohibited. References for instance can only get their meaning when the content is placed on the page, and this can be many paragraphs later. Postponing is needed to keep the references in tune with the page numbers. When you hear macro writers talk of whatsits, they are probably talking about such postponed expansion and hidden things.

The story is still not completed. Hidden for the user, and depending on the macro package in use, TEX also inserts things like color directives, indentation, skips and when asked for all kind of frills, like paragraph numbers. This means that what you type is not per se what you will get, in fact, what you see is what TEX made of your input. Some things are quite complicated in TEX. Take for instance multi-column typesetting. Splitting columns is to be programmed and is not part of TEX. The same goes for adding line numbers. Such at first sight simple things ask for recalculating page breaks and/or reading back already typeset lines and post processing them.

Hans Hagen

Bijlage 3

Color in professional print production

Siep Kroonenberg
Faculteit der Economische Wetenschappen
Rijksuniversiteit Groningen
n.s.kroonenberg@eco.rug.nl

abstract

This paper takes a look at issues arising in color printing, such as color models, color conversion and color separation. Increasingly, it is feasible to perform these functions on existing PostScript files, independent of the authoring software. The pdf format plays a key role in this trend.

keywords

Color printing, color model, device-independent color, color conversion, color separation, pdf

Can TEX produce professional-level color output? We shall see that, as to color support, TEX lags far behind commercial desktop-publishing software. But we shall also see that there are some viable options, and that recent developments tend to move advanced color support back to where it belongs, viz. to the printer.

Although TEX doesn't have built-in support for color, the `\special`-mechanism provides hooks for adding driver-dependent color support[1].

Color perception

Our retina contains three types of light-sensitive cone. Color vision depends on these types of cone being especially sensitive to resp. the red, green and blue part of the visible spectrum. If light is composed of wavelengths distributed evenly across the visible spectrum, we perceive it as white. In other words, red, green and blue (1,1,1) add up to white. The absence of all three (0,0,0) means black, or no light. Rgb is an *additive* color model.

In conventional printing, on the other hand, we start out with white, or no ink. By adding inks, more and more light gets absorbed. The cmy (cyan, magenta, yellow) color model is the opposite of the rgb model: cyan absorbes red, magenta green and yellow blue. Cyan plus magenta plus yellow absorbes approximately everything, therefore is approximately black. In the cmyk model, black ink is added to get a better black than is possible with the other three colors alone, and because replacing equal amounts of cyan, magenta and yellow with black saves ink. Cmy and cmyk are subtractive color models.

Defining colors in TEX and LATEX

In LATEX, color support is standardized through the `color` package:

```
\usepackage[dvips]{color}
```

The color package supports various color models, which can be freely mixed within one document. We might use rgb and cmyk colors as follows:

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

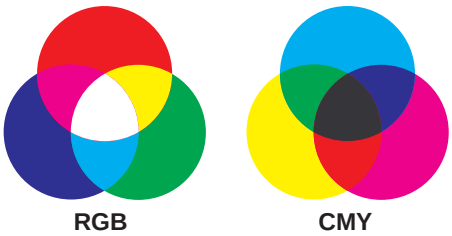


Figure 1. Mixing colors in rgb and cmyk

In the gray color model, 0 is black and 1 is white:

```
\definecolor{mygray}{gray}{0.5}
\textcolor{mygray}{gray}
```

The *named* color model allows one to use named colors:

```
{\color[named]{Apricot} Apricot}
```

In the case of dvips, a set of named colors, including 'Apricot', is predefined as cmyk colors.

A full description of the syntax for defining and using colors can be found in the LATEX Graphics Companion and

1. *Specials* are driver-dependent codes which TEX can place in a dvi file. Implementing color in TEX requires on the one hand macros which ask TEX to write specials to the dvi file, and on the other a dvi driver which interprets those specials. Dvi drivers are at liberty to ignore specials which they don't understand, guaranteeing that dvi files with specials still are device-independent in the sense that they can be processed by arbitrary dvi drivers.

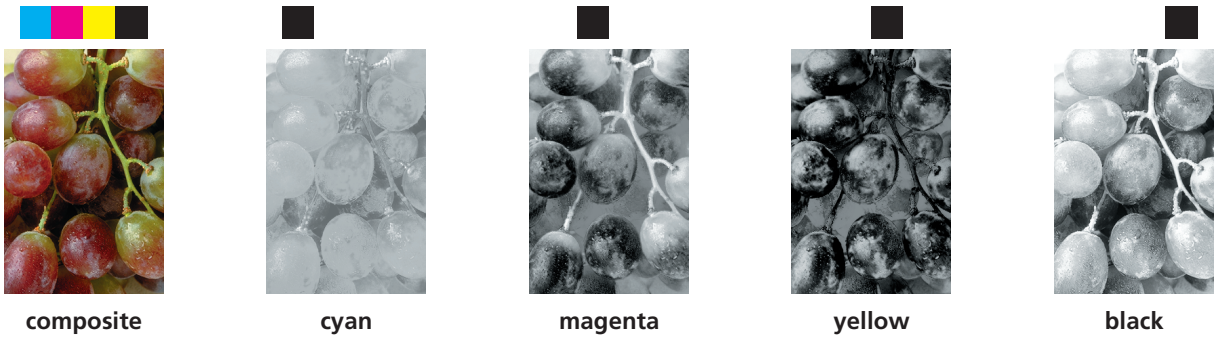


Figure 2. Separations

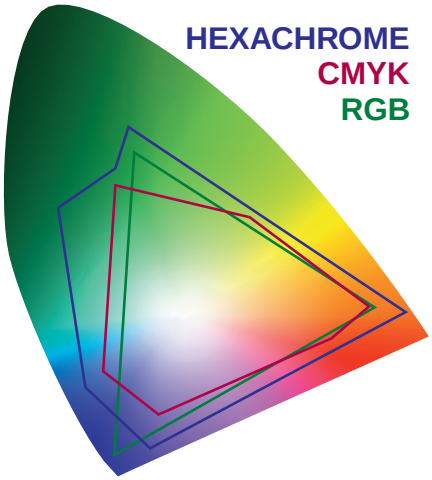


Figure 3. Color gamuts. The colored shape represents the gamut of visible colors; the inscribed polygons represent the gamuts of several device-based color models. The hexachrome model is a print- based model in which green and orange inks are added to slightly modified cmyk inks.

in the documentation accompanying the graphics and color packages.

Users of plain TEX can make use of colordvi.tex, which offers color support for dvips and is distributed with this program.

Device-independent color

The rgb color model does not actually refer to the red, green and blue cones in our eyes, but to the red, green and blue components of devices that generate color images (monitors) or record them (film, scanners). The same rgb value on different devices does not necessarily correspond to the same color. In general, rgb devices are not able to represent all visible colors. The rgb *gamut* (for a typical rgb device) is smaller than the gamut of visible colors.

We already saw that on cmyk devices, i.e. printers, colors are generated in a radically different way than on monitors, so it is only natural that the cmyk gamut differs from the rgb gamut. The cmyk gamut falls even farther short of the gamut of visible colors.

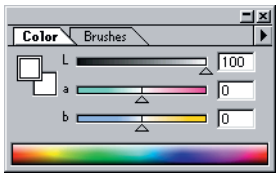


Figure 4. Specifying L*a*b colors in Photoshop

Even for someone not interested in color fidelity, the differences between rgb and cmyk are too gross to ignore. If you are used to choosing colors on screen, you'll find that vibrant colors turn dull and flat on paper; if on the other hand you select your colors from printed swatches you'll find that your screen just can't reproduce some cmyk print colors such as pure cyan.

You can specify color in vector graphics and within TEX as cmyk colors and you may never have to deal with color conversions. Photographic images, however, start out life in rgb mode, and if they are to be printed in conventional process color, the differences in rgb- and cmyk gamuts must be dealt with. The challenge here consists of contracting the source gamut within the target gamut without disturbing color relations within the image.

Note that we quietly ignore differences between devices with the same color model: not all monitors are the same, and the same cmyk color specification will look very different on newsprint and on glossy art paper. Professional designers, who really need accurate color, calibrate their monitors and carefully target their output for specific print setups. You might even have to pay money for color profiles of graphics devices.

To resolve these color conversion issues, the CIE (commission Internationale de l'Éclairage) created in 1931 a standard for color specification which allowed device-independent color and is based on color perception. The

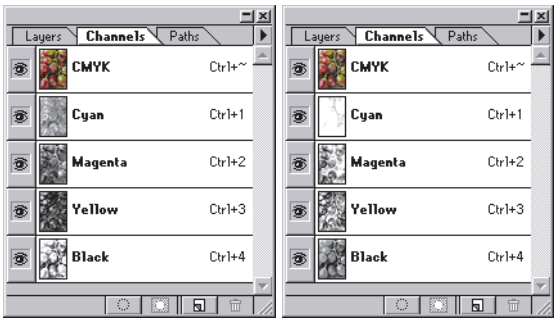


Figure 5. Converting rgb to cmyk: black generation. This illustration shows the Photoshop Channels palette for two cmyk versions of one rgb original (the same original as in the separation example). In Photoshop speak, the various color components of an image are channels. In the left palette, black generation is low, i.e. lesser amounts of cyan, magenta and black are replaced with black. In the other one, black generation is set to the maximum: equal amounts of cmy are replaced by the equivalent amount of black. This results in lighter cmy channels and a darker black channel. The advantage of a low black generation setting is more shadow detail but may result in more ink buildup than the paper can hold.

best-known CIE color model is L*a*b, whichcolor space has one lightness dimension *L* and two color dimensions *a* and *b*.

Ideally, images should be stored in a device-independent color model, and only at output time be converted to the appropriate device color space. This would facilitate designing simultaneously for different media, which is not unimportant in this age of the Internet.

Spot color

At the other end of the scale there is spot color, which is mostly relevant for printing. Pantone publishes books of color swatches which serve as a reference. If you want a second color for your publication then you cite the Pantone number or -name to your printer.

Black and one or two Pantone colors is cheaper than full-color: fewer print plates are required, and a spot color design usually doesn't require quite the same care in registration and color matching as a full-color job. However, spot color is not used exclusively for economy: a corporate identity might involve the use of spot color for the company logo and for letterheads, often in addition to the four process colors. A cmyk approximation of a Pantone color often isn't close enough, and fine details always suffer if they are rendered with a halftone screen or multiple inks.

Spot varnishes and metallic inks also require separate printing plates, often in addition to plates for process color.

Color separation

Color printing is still mostly done with conventional presses, with a separate plate for each ink; in other words, conventional color printing requires color separation.

In very simple cases this may be done manually: if a page contains black and red elements, one can make all red objects white and then print the black objects; for the red plate, one can make black objects white and red objects black.

This is in fact not so very different from what separation programs do: they would print such a page twice with appropriate redefinitions of the colors.

Professional graphics applications have this capability built in. TEX users can create cmyk separations with dvips using a command-line such as

```
dvips -h colorsep.pro -b 4 test.dvi
```

the parameter `-b 4` specifies that each page should be printed four times, and the parameter `-h colorsep.pro` loads a header file colorsep.pro which takes care of color redefinitions. It also adds cropmarks, which indicate the paper edge but also assist the printer in aligning the plates on press. With some tinkering it is not too difficult to modify this procedure for spot color separation. However, I am not aware of any ready-made solutions for using spot color *in addition to* process color. Another problem with colorsep is that bitmaps can't be separated this way, unless they contain cmyk data[2].

Another PostScript program, aurora.pro, does offer some support for spot color, but it recognizes a spot color by its cmyk composition instead of by its name. This program can handle more general color models in bitmapped images. A drawback of Aurora is that it generates separations per ink instead of per page. Film is subject to some stretch and shrink, therefore good registration requires that all films for one page are run at one go. Aurora does not add cropmarks.

When you use colorsep or aurora, you find yourself very much on the bleeding edge, so be prepared to run tests before committing yourself on a large job. Better, if at all possible you should leave separation to your printer or service provider. He may have postprocessing software to separate existing PostScript print files, or his imagesetter may have separation capabilities built-in.

Overprint, knockout and trap

Registration of print plates is never perfect. Even slight misregistration may be very noticeable. One remedy is to

2. The most economical way to turn a grayscale image into cmyk data from is to convert it to a black *monotone*, i.e. a duotone with just one color, which is black.

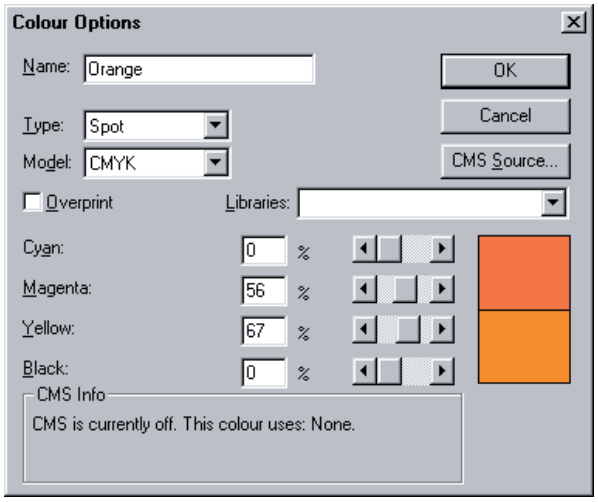


Figure 6. Specifying colors in PageMaker. Here, a spot color is being defined, with a cmyk equivalent. At separation time, one will have the choice whether to print 'Orange' on a plate of its own or whether to use the cmyk equivalent. Note the Overprint setting. If it had been checked, colors on other plates would not be knocked out underneath orange objects.

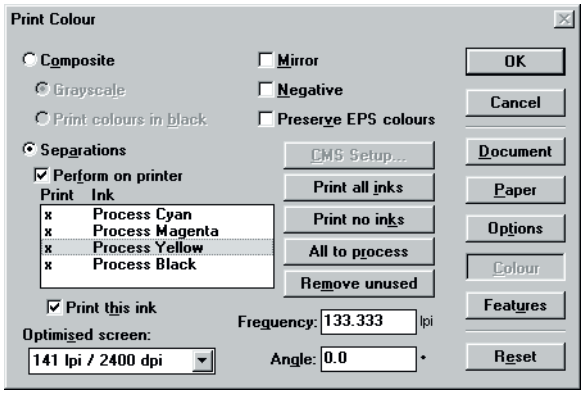
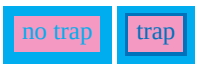


Figure 7. The Color page of PageMaker's print dialogue. Notice the 'Perform on printer' checkbox and the screen settings. Professional graphics applications consult PPD's or PostScript printer descriptions as to the capabilities of a printer. The PPD, which was selected in the first page of the print dialog, is for a PostScript level 2 imagesetter with built-in separation capabilities.

create a slight trap or overlap where colors touch. I hope that the printing of the MAPS won't be too perfect, otherwise the following illustration won't show my point (the trap is greatly exaggerated):



If one color is darker than the other, then the lighter color

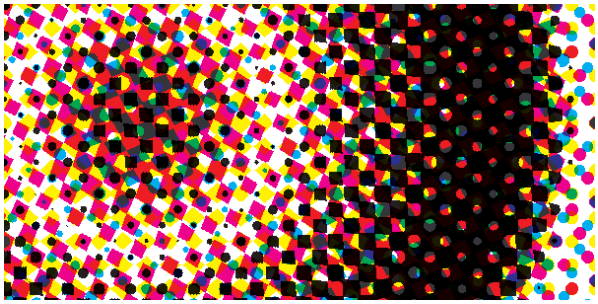


Figure 8. A greatly magnified process-color halftone. This example displays some moire.

should spread into the darker, in order that the integrity of shapes is maintained.

Trapping introduces device-dependence: the required amount of trap depends on press conditions. Trapping may be done by a postprocessing program or even by the imagesetter; ask your printer. Inexperienced users probably do well not to do their own trapping. Instead, avoid trapping by selecting adjacent colors with enough ink in common, so that misregistration won't be conspicuous, or have trapping done by the printer or service bureau, or even ignore the problem.

A simpler remedy against misregistration is overprinting. If e.g. a black shape sits on top of a magenta background, then with overprinting the magenta background continues underneath the black object:

	black plate	magenta plate	final printout
knockout	**text**	**text**	**text**
overprint	**text**	**text**	**text**

For small type and fine detail, this is the only reasonable solution. In TEX, this might be accomplished in the above example by adding magenta to the text color:

```
\colorbox{magenta{\color[cmyk]{0,1,0,1}text}}
```

This trick is also useful when printing to color printers, which ignore overprint specifications.

Professional draw- and desktop-publishing applications allow one to set a color simply to overprint instead of knockout. With TEX and dvips, one would have to insert raw PostScript. The workaround described above, viz. adding background color to the foreground object, breaks down if the background does not consist of a single color.

Screens

Tints of a color are achieved by printing dot patterns. To this end, the imageable area is divided into grid cells, with

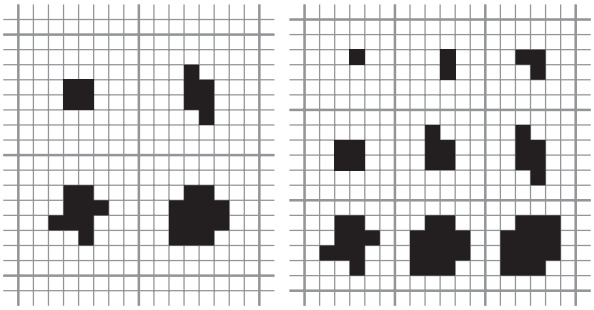


Figure 9. Left a 0° 8×8 grid with 65 graylevels, right a 0° 6×6 grid with 37 graylevels.

each cell being white, black or containing a cluster of black dots. Since the grid cells must consist of an integral number of dots, only a discrete series of screen frequencies is available for a given resolution.

Finer screens allow for rendering of finer detail, but because a grid cell consists of fewer printer dots, fewer gray levels are available. Also, an excessively fine screen might clog up on press or distort gray values, since dots tend to spread out somewhat on press. This latter phenomenon is termed *dot gain*.

With color printing, there is an additional twist: screen angles and -frequencies of the different plates must be coordinated with some care in order to avoid interference or *moire* patterns.

A recent development is FM, or frequency-modulated screening. This does away with moire problems, but aggravates dot gain, and is computationally very expensive.

This is yet another area where TEX offers no control except to PostScript hackers. It may not be smart to rely on default settings of an imagesetter, since these may be very conservative, resulting in unnecessarily coarse screens. If your printer is willing to handle screen settings for you, so much the better.

Color printers

For shorter print runs, digital printing deserves serious consideration. I am not going to quote a number, since the break-even point is moving up all the time, and quality is getting better too. These printers often employ custom screening technologies or *contone* or continuous tone, which is achieved through variable dot size. Contone technology won't help type or line art, but does offer very sharp and detailed rendering of photographic images at relatively low resolutions.

This has consequences for the resolution needed for photographic images: whereas for halftoned images a resolution of more than twice the screen frequency is pointless (say, 300 dpi for a 150 lpi screen), contone devices can make use of image resolutions up to the device resolution, which will be 400 and up.

The printer may use additional toners besides the traditional cmyk colors in order to improve color fidelity. Obviously, the printer must do some translating internally in order to use these toners, since it is likely to get cmyk or rgb instead of custom color data. If you have rgb images, it might be best to leave them in that format; ask your printer and/or arrange for some tests.

Pdf workflow

PostScript was meant to be device-independent but it turns out that we still need to know a lot about the output device. If your printer or service provider has facilities for postprocessing such as color separation, trapping and imposition, consider creating generic or device-independent PostScript and let him handle the device-specific part, including color model conversion.

Adobe's pdf (portable document) format should facilitate such a division of labor. Pdf has the same imaging model as PostScript. It should be a more tractable format for postprocessing programs since it does not support programming constructs, and page independence is built in. The current version 1.2 of the pdf format can hold the same prepress information as PostScript including CIE Lab and other CIE color specifications. PostScript 3 printers support pdf directly.

Adobe envisions a future in which documents travel the prepress workflow in pdf- instead of PostScript format, retaining device-independent color until a late stage, and in which device-specific processing such as trapping, color conversion and -separation, and imposition can be added at successive stages. TEX users will generally prefer to have most prepress information added late, i.e. by the printer, whereas many professional designers will prefer to retain control themselves, and add prepress settings early, i.e. from within their authoring software.

The prepress- and print industry seems quite enthusiastic about pdf. Pdf-based printers appear on the market, and software support is rapidly improving. One example is the Crackerjack plugin, which adds color separation with all its twists and turns to Acrobat Exchange. Soon, TEX users no longer need to be jealous of the prepress capabilities of QuarkXPress and PageMaker (not that we ever were, but we should have been), because these capabilities are increasingly becoming available outside authoring software.

Production note

The illustrations for this paper were all converted to or created with cmyk color. The publication itself will be converted by the editors from TEX to PostScript to pdf, and by the printer back to PostScript. If somebody fails to pay at-

tention, some colors will have traveled from rgb to cmyk back to rgb and again to cmyk and will be much the worse for wear ;-).

Further reading

For online documents, dates and exact locations are omitted, because these are subject to too much change.

Adobe Systems, PostScript Language Reference Manual, second edition, Addison-Wesley 1990.

Adobe Systems, Portable Document Format Reference Manual. Available online from `www.adobe.com`.

Adobe Systems, PDF for Prepress Workflow and Document Delivery. Available online from `www.adobe.com`. Various other documents of interest on pdf, PostScript Level 3 and color management can be found at this site.

David Carlisle and Sebastian Rahtz, Graphics package. Available from CTAN and included in most TEX/LATEX distributions. This is in docstrip format; just run `latex graphics.dtx` to get formatted documentation.

Michel Goossens, Sebastian Rahtz and Frank Mittelbach, The LATEX Graphics Companion, Addison Wesley, 1997.

Thomas Rokicky, Dvips manual. Included in the dvips distribution; available from CTAN and included in most TEX distributions.

Postprocessing software vendors, see e.g. `www.lantanarips.com` for Crackerjack, and `www.imation.com` for Trapwise and Presswise.

Expansion, what is that? (continued)

Although from the users point of view, TEX expands whatever macro it meets while reading the paragraph, in practice expansion can be postponed of prohibited. References for instance can only get their meaning when the content is placed on the page, and this can be many paragraphs later. Postponing is needed to keep the references in tune with the page numbers. When you hear macro writers talk of whatsits, they are probably talking about such postponed expansion and hidden things.

The story is still not completed. Hidden for the user, and depending on the macro package in use, TEX also inserts things like color directives, indentation, skips and when asked for all kind of frills, like paragraph numbers. This means that what you type is not per se what you will get, in fact, what you see is what TEX made of your input. Some things are quite complicated in TEX. Take for instance multi-column typesetting. Splitting columns is to be programmed and is not part of TEX. The same goes for adding line numbers. Such at first sight simple things ask for recalculating page breaks and/or reading back already typeset lines and post processing them.

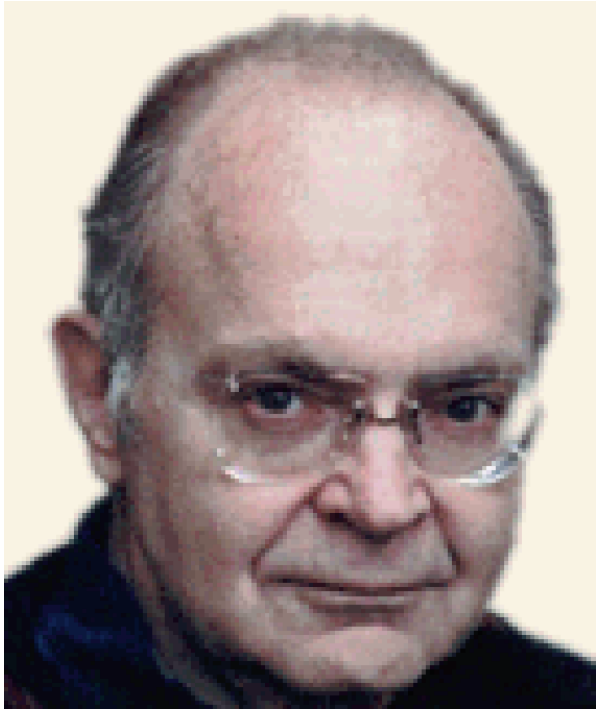
Hans Hagen

Bijlage 4

Microsoft Buys TEX, Plans New Products – Stanford Professor Reaps Windfall

PALO ALTO, CALIFORNIA, USA (CNEWS/ MSNBC) —

In a major move into the scientific publishing market, Microsoft Corporation announced today that it has purchased all rights to the computer language and document compiler known as TeX (pronounced, "tech"), and plans a major new product line based on the 20-year-old software.



Donald Knuth

Stanford Professor Donald Knuth (pronounced, "kah-nooth"), the author of the widely-used TeX software, in a joint press conference at the university campus with Microsoft Chairman Bill Gates, acknowledged that the two had been negotiating for some months. "I felt that two decades of TeX in the public domain was enough. I am reasserting the copyright to my original work in TeX. Microsoft will carry the ball now, and I can get back to my computer science research." Knuth acknowledged he was paid a "seven-figure sum" from Microsoft, which he will use to finance his work on a project he has code-named "Volume 4".

At the press conference, Microsoft chairman Bill Gates said the acquisition was "the kind of cooperation between academia and industry that builds prosperity for both." He added that TeX would "finally give Microsoft a foothold in mathematical desktop publishing" that has eluded the software giant since its founding. Drawing gasps of surprise from the college audience, Gates asserted that "TeX will soon be biggest jewel in the Microsoft crown."

Bill Gates

Apparently the jewel metaphor will include a hefty, unavoidable price tag for future TeX users. Gates outlined plans whereby all existing TeX compilers would be phased out, to be replaced by a new Microsoft master implementation written in C++. Beta versions for public testing on Windows 95 and NT platforms are expected in late 1998, issuing from a new 205-programmer project laboratory at Microsoft's Redmond campus. Microsoft TeX for other platforms, such as Unix workstations, will follow at an as-yet unspecified date. According to Gates, "the master TeX from Microsoft will ensure that the incompatibilities

across platforms are once and for all eliminated." TeX software is widely used due its portability, although variations among operating systems have been troublesome due to uncoordinated development.

Unlike the technical aspects of the project, Gates explained that pricing for Microsoft TeX has already been firmly set. The single-user retail product is expected to have a street price of about $600 and consist of three CDs. When heckled by an graduate student complaining about a high price for a formerly free product, Gates seemed startled, explaining that a "student edition at $299 is likely" and that "Microsoft will use the revenue to make TeX better."

Most current users of TeX have paid nothing for their implementations, derived from Professor Knuth's formerly-free work. Before leaving the podium, Gates made a final comment that "TeX hasn't changed in years. What kind of a product can that be?", and then handed the microphone to an assistant, introduced only as the project leader for Microsoft TeX.

The assistant displayed an overhead presentation using the current test version of Microsoft TeX. Equations and tables could be seen dissolving into each other in a morphing action between frames. "No one has ever done that with TeX," Gates announced from an audience seat at one point. "It's the kind of sizzle that can really enliven a dull paper at an academic conference." Some onlookers were not convinced, especially when the program crashed midway through the demonstration, resulting in a five-minute delay while Windows 95 was restarted. Microsoft technicians later blamed a third-party display driver.

The impact on the large base of existing TeX users was unclear. During a question-and-answer period, Gates said that the "TeX" trademark would be registered as the exclusive property of Microsoft, and could not appear in any competitive or free software. "We are granting of our own good will until the 3rd quarter of 1998, free use to any existing TeX vendors or public-domain authors. That's plenty of time for an orderly phase-out and change-over to Microsoft TeX, or no TeX at all. After that, our legal department will be contacting them."

A Microsoft attorney added that some of the project personnel would be dedicated to searching the Internet to find non-Microsoft TeX software. "Archives and collections of TeX-related programs will not be permitted. The standards must be enforced, or they become meaningless. We are rescuing a fine piece of work from being diluted into worthlessness. You would not believe the number of programs that have been based on TeX without any central, controlling authority. We will stop this infringement."

Some large organizations dependent on TeX were stunned by the announcment and had not yet formed plans for dealing with the change. At the American Mathematical Society, whose publications largely depend on TeX for typesetting, editor Barbara Beeton was incensed. "I can't believe Don [Professor Donald Knuth] sold us out like this. We should have never based a publishing enterprise of this scope on so-called public-domain software. What were we thinking?" Publication schedules for the rest of 1998 were on hold, and journal editors scrambled to reassure their authors that deadlines would not slip more than a few months.

Certain small businesses are also expected to feel the impact of the Microsoft ownership of TeX. Palo Alto restaurant owner Wu Chen appeared unhappy at the news, stating that "for ten year I print new menu every day with TeX, now I will pay big time." He displayed a crumpled, grease-spotted take-out flyer, and with tears in his eyes explained how multiple columns, exotic typefaces, and daily price changes could all be printed by TeX in a multi-lingual format. "In Wordperfect this would be a long journey."

Commercial vendors of TeX software stand to lose everything in the face of the new Microsoft monopoly. While most derivatives of TeX were freely published, several companies had made a business of publishing proprietary versions. One anonymous source from a leading TeX firm said that "publishing TeX was a gold mine while it lasted, and the Internet let us mine it deeper and deeper. Now this is a cave-in right on our heads. TeX was a monumental work of beauty and utility, freely given to the world by one of the finest and most generous minds of the 20th century. Now it belongs to a lucky dropout. We're finished."

Date of Publication 04/01/98

Nederlandse TEX Gebruikersbijeenkomst

Aan	:	Leden Nederlandse TEX Gebruikersgroep
Betreft	:	Verslag 1e bijeenkomst Nederlandse TEX Gebruikersgroep
Datum	:	23 juni 1988
Notulist	:	Gerard van Nes (ENR)
Behandeling	:	bij de volgende bijeenkomst
Aanwezig	:	R. Alma (RUG); J. Braams (PTT); C.H.J. van den Brekel (Philips); M.A.J.H. Broeren (Océ); H. Brouwer (EGD); R. de Bruin (RUG); N. Cox (URC); E.J. Evers (RUU); J. Grootenhuis (CIRCE); M. Helmig (Philips); A.R. Jooya (Open Universiteit); G. Haayer (Styx); T.A. Jurriens (RUG); J.S.M. van Knippenberg (Océ); A.P.J.M. Krijnen (ACCU); C.G. van der Laan (RUG); J.R. Luyten (RUG); A.J. de Meyer (RUU); H. Mulders (KUB); G.J.H. van Nes (ENR); C. de Ridder (Philips Nijmegen); J. Röling (Styx); J. Soutberg (Elsevier); G.J. Stemerdink (LUW); P. Tutelaers (TUE); F.J. Veldhuis (RUG).
Bericht van verhindering	:	A.P.M. Ezendam (MARIN); A.J. van der Goot (EGD); H.J. Jacobs (RUU); G. de Jong (EU); N.A.F.M. Poppelier (RUU); O. van Schalkwijk (RUG)

Agenda:

1. Opening en Mededelingen
2. Introductie deelnemers
3. Brainstorming TEX aandachtsgebieden
4. Instellen werkgroepen voor de aandachtsgebieden
5. Formele zaken
6. Rondvraag
7. Sluiting

Bijlagen:

A Werkgroepen Nederlandse TEX gebruikersgroep
B Listserver
C TEXHaX
D TEX publications information

1 Opening en Mededelingen

Voorzitter Kees van der Laan (RUG), die deze bijeenkomst samen met Gerard van Nes (ENR) heeft georganiseerd, opent om 11:00 uur de vergadering. Hij vermeldt dat naar aanleiding van deze bijeenkomst ongeveer 40-50 positieve tot zeer positieve reacties zijn ontvangen. Een aantal TEX belangstellenden waren verhinderd om op deze dag aanwezig te zijn. Tevens blijkt niet iedereen de uitnodiging voor deze dag ontvangen te hebben (waaronder de Katholieke Universiteit Nijmegen). Het opzetten van een goed adressenbestand zal één van de taken van deze gebruikersgroep zijn.

Een andere belangrijke taak van de gebruikersgroep is, om de verspreide werkers met elkaar

in contact te brengen en te stimuleren dat het gebruik van TEX en aanverwante produkten eenvoudiger wordt via het opzetten van cursussen, produkt evaluatie, uitwisseling know-how en public relation activiteiten. Een belangrijk middel hiertoe is het oprichten van werkgroepen.

Een groot aantal TEX gebruikers hebben de questionnaires ingevuld en teruggestuurd. Iedere aanwezige ontving een kopie, naast enige informatie welke was ontvangen van: de Jong (Erasmus Universiteit), Kuiken (Kuiken Consulting), Veldhuyzen van Zanten (SARA) en van der Laan (RUG).

De samenstelling van de gebruikersgroep is breed: universiteiten (vakgroepen, rekencentrum, universiteitsdrukkerij), uitgevers, zetters, softwarehouses en leveranciers van randapparatuur, m.n. (laser)printers. Een andere classificering is: makers, gebruikers en belangstellenden.

Kees van der Laan vermeldde dat er reeds contacten zijn met de Engelse TEX gebruikersgroep die graag informatie wil uitwisselen met de Nederlandse groep. Ook zijn er positieve reacties via TEXHaX vanuit de USA ontvangen.

Verschillende gebruikersactiviteiten zijn waar te nemen, waaronder:

- TUGboat, TEXHaX, TEXMaG (zie bijlagen),
- SGML activiteiten;
  (Kees van der Laan participeert in het bestuur; voorzitter is Jan Maasdam van Wolters-Kluwer),
- Relatie computer en niet-westers schrift,
- Europese gebruikersgroep (gebruikersbijeenkomst in juli a.s.).

Doel van deze bijeenkomst zou ondermeer moeten zijn het starten van een inventarisatie van wat er allemaal is, van opgedane ervaringen in het gebruik en i.h.a. het uitwisselen van informatie en know how.
M.b.t. het eerste is een boekje met de ingevulde questionnaires samengesteld; voor het tweede zie punt 2 van het verslag; voor het laatste zie het verslag en de bijlagen.

Daarentegen zou de groep wel moeten waken, geen reclame bureau te worden van allerlei aanwezige commerciële pakketten op het gebied van TEX en aanverwante produkten, noch een "vergaderclub" te worden.

2 Introductie deelnemers

De aanwezige deelnemers werden in de gelegenheid gesteld zichzelf voor te stellen en eventueel enige toelichting te geven op de door hun ingediende questionnaires.

Kees van der Laan (RUG)
Werkt ongeveer 2 jaar met LATEX op VAX-8600 systeem. Bezit een μ-TEX campuslicentie waar echter weinig gebruik van wordt gemaakt (10 à 15 participanten in licentie waaronder de Universiteits bibliotheek). Het Rekencentrum geeft twee maal per jaar een LATEX introductie cursus. Cursusmateriaal is zelf ontwikkeld (rapport met algemene overzichten én opgaven én uitwerkingen; abstract was bijgesloten bij de documentatiemap). Aan een LATEX cursus voor gevorderden en een TEX cursus wordt gedacht.
RC-RUG streeft naar integratie van μ-TEX, en TEX op VAX en koppeling met de Universiteits Drukkerij.
Er is een RUG overleggroep documentpreparatie waar de aandacht op TEX is gericht.
Hebben ervaring met Metafont. Onderzoeken WordPerfect naar LATEX conversie programmatuur. Over K-Talk is men niet zo tevreden. Een layoutprogramma uit TEXHaX om het effect van verandering van pagina-opmaak parameters van LATEX te zien, is geëvalueerd en verbeterd. Is daarnaast bezig met SGML $\leftrightarrow$ TEX en i.h.a. met documentpreparatie. Op de SUN-3

wordt geëxperimenteerd met "The Publisher".

Gerard van Nes (ENR)
ENR is sinds ongeveer een half jaar bezig met LaTeX op een VAX systeem. Het gebruik is nog beperkt tot enkele personen, doch men verwacht dat dit tijdelijk is. Men is op zoek naar een implementatie voor de CYBER mainframe (NOS/VE). Ook zoekt men naar een LN03R driver die direct van de hardware fonts gebruik kan maken.

Henk Brouwer (EGD)
Werken ongeveer 1-1.5 jaar met μ-TeX op PC. Hadden aanvankelijk moeilijkheden met het implementeren van TeX op het Unisys-1100 systeem, doch m.b.v. van een Amerikaanse site nu eindelijk gelukt. Men mist echter drivers voor de HP laserprinter.
Met LaTeX wordt ook gewerkt doch beperkt.
Binnen een interne studiegroep is men bezig met het onderzoek van het schrijven van macro's t.b.v. LaTeX.

Huub Mulders (KUB)
De Katholieke Universiteit Brabant gebruikt ongeveer 2.5 jaar hoofdzakelijk LaTeX op de VAX-8700. TeX macro's worden zelf geschreven om toegevoegd te worden aan die van LaTeX. Ongeveer 15 mensen gebruiken het produkt regelmatig. Men is geïnteresseerd in drivers die de hardwarefonts van de outputdevices aansturen.

Johan Braams (PTT)
Maken bij previewen gebruik van VT220 terminals. LN03+ wordt als outputdevice veel gebruikt. Er wordt nog niet met PostScript uitvoer gewerkt.
Bezitten ondermeer IdxTeX, GloTeX en SliTeX. Er zijn ongeveer 30-40 regelmatige gebruikers. Toonde een "Local Guide" met een overzicht van bij hun aanwezige software.

Gerrit Stemerdink (LUW)
Na ongeveer 1 jaar met LaTeX gewerkt te hebben bij de KUB, is enige maanden geleden besloten om zelf het produkt te installeren. De aanvankelijk ontvangen Decus tape bleek verouderde informatie te bevatten. De versie die zij nu gebruiken is die van Tilburg.
Missen vooral de systeem documentatie: bijvoorbeeld welke font file zijn nodig bij de installatie. Previewen met VT240; outputdevices LN03 en HP laserjet. De laatste is nog niet aan de LUW gebruikers vrijgegeven.
Hij is op dit moment zelf de enige redelijk intensieve gebruiker. In het totaal werken er ongeveer 7 mensen met LaTeX.

Jeroen Soutberg (Elsevier)
LaTeX staat bij hun nog in de kinderschoenen. Verwacht er over ongeveer een maand mee te beginnen. Willen het mogelijk als uitgangspunt voor het zetwerk gebruiken. Het systeem zal een PC zijn.
Ook SGML ligt binnen hun interesse sfeer.
Is geïntereseerd in bepaalde fonts die niet binnen LaTeX aanwezig zijn.

Jan Grootenhuis (CIRCE)
Zijn eerste contact met het produkt dateert uit 1980. Heeft er tevens mee gewerkt in het kader van een SGML onderzoek. Is in het bezit van de μ-TEX versie. Omzetting SGML naar TEX geeft geen problemen. De Addison & Wesley versie levert nog wel de oude type fonts. LATEX kent echter specifieke Amerikaanse typografische problemen. Vermeldt dat het LATEX Lamport boek geen goede aanwijzigingen voor de gebruiker geeft.

Marius Broeren (Océ)
Hebben TEX ongeveer 4 jaar in gebruik op Vax-750 systeem. Op de SUN wordt LATEX vooral gebruikt t.b.v. rapporten. Voor de μ-TEX heeft men een driver geschreven t.b.v. hun eigen laserprinter (documentatie van deze printer was beschikbaar). Metafont wordt tevens gebruikt op de μ-Vax.
Op dit moment ligt het gebruik bij ongeveer 8 medewerkers.

Jan van Knippenberg (Océ)
Verwees naar hetgeen door zijn collega reeds was vermeld.

Jan Luyten (RUG)
Redenen om het TEX te beginnen waren de mathematische tekstmogelijkgheden van het produkt. Een onderzoek is gaande m.b.t. K-Talk (conversie WP naar TEX). Een aantal onderdelen hiervan lopen nog niet volledig.
Geeft cursus LATEX (TeX).
Er blijkt vooral belangstelling voor LATEX (TEX) te zijn vanuit de taalkundige hoek.

Andre de Meijer (RUU)
Is ongeveer 2 jaar geleden begonnen met tekstverwerking op de PC, t.w. Wordstar met daar bovenop Exact. De mogelijkheden bleken niet ideaal te zijn, hetgeen ook de reden is dat men met TEX begon; aanvankelijk op de Olivetti PC doch sinds kort ook op de Atari-ST.
Hebben de mogelijkheden van LATEX wat uitgebreid.
Ondervinden de 8-bit communicatie problemen bij de terminals.
Op een vijftal secretariaten wordt nu LATEX gebruikt. Het totaal aantal actieve gebruikers is 10-15. Er is 1 AmsTEX gebruiker.
Er is behoefte aan WYSIWYG en aan een Nederlands afbreekmechanisme.

Ton Krijnen (ACCU)
Onderzocht is geworden in hoeverre het ACCU een rol kon spelen bij het onderhandelen voor een company licentie voor de TEX produkten voor de PC en Atari.
De ondersteuning wordt nog in kaart gebracht. Over enige maanden is hier meer van bekend.
Het ACCU is geïnteresseerd in algemene ondersteunings- en cursusmogelijkheden. Binnen het ACCU zijn er ongeveer 6-7 TEX / LATEX gebruikers. Andere belangrijke tekstverwerkers binnen het ACCU zijn Display Writer en WordPerfect.

Evers (RUU)
Als promovendus in aanraking gekomen met TEX. Heeft nu een functie binnen de automatiseringsafdeling van de faculteit der geneeskunde.
TEX draait momenteel op Vax stations. Er wordt nog geen gebruik gemaakt van laserprinters. Zijn wel geïnteresseerd naar ervaring hiermee. Binnen de diverse afdelingen van de RUU is er wel overleg m.b.t. de TEX programmatuur.

Evers vermeldt tevens dat Nico Poppelier verhinderd was om op de bijeenkomst aanwezig te zijn.

Geert Haayer (Styx)
Houdt zich bezig met het uitgeven van Oriëntaalse teksten. TEX biedt transcription mogelijkheden in deze. Maken gebruik van μ-TEX. Hebben diverse Beebe drivers aangeschaft. Hebben geëxperimenteerd met diverse font omzettingen.
Vermeldt dat LATEX nauwelijks flexibiliteit heeft i.t.t. TEX. Vinden LATEX ook geen goede ontwikkeling omdat dit vele mogelijkheden ontneemt.
Is van mening dat TEX niet geschikt is voor een eenvoudige brief, doch daarentegen wel voor moeilijk zetwerk.
Willen met METAFONT werken om ondermeer Hebreeuwse en Arabische fonts te genereren.
Frans Velthuis (RUG) merkte hierbij op, dat in het volgende nummer van TUGBOAT een lijst geplaatst zal worden met de beschikbare fonts binnen TEX.
Hij vermeldt de problemen bij het over twee kolommen plaatsen van teksten (tekstboxen) die met elkaar vergeleken moeten kunnen worden.
Theo Jurriens meldt daarop dat hij in het bezit is van een programma welke wisselend 2 alinea's leest van 2 files en deze binnen één TEX-pagina plaatst.

Joris Röling (Styx)
Wil fonts zelf ontwerpen. Is zeer gecharmeerd van de mogelijkheden van TEX.

Van den Brekel (Philips)
TEX/LATEX wordt op VAX systeem gebruikt. Ongeveer 300 gebruikers. Zowel rapporten als wetenschappelijke publicaties worden met het systeem aangemaakt.
De IBM groep is niet tevreden met hun huidige produkt SCRIPT. Het streven is ook om gelijke software te hebben als de VAX groep. Hebben daarom de CMS versie van TEX aangeschaft en een week geleden geïnstalleerd. Is niet direct geïnteresseerd in het Nederlands afbreekmechanisme, doch staat er wel achter.
Secretariaat is een belangrijke doelgroep.

Helmig (Philips)
Werkt bij de Audio Visuele dienst afdeling produktie en presentatie. Hebben gekozen voor Macintosh en het TEX pakket Textures.
Hebben belangstelling in fotozetters mogelijkheden. Vermeld dat de Computer Modern fonts daar niet bruikbaar zijn i.v.m. de mogelijke verschillen.

Abdy Jooya (Open Universiteit)
Is ongeveer 1.5 jaar bezig met TEX op VaX cluster systeem en op PC. Als output device wordt de Xerox-4050 gebruikt.
Hebben problemen met de Interpress driver.
Houdt zich sinds kort met het opzetten van cursusmateriaal bezig. Geeft ook eigen cursussen. Hebben een eigen TEX handleiding geschreven. Cursusmateriaal wordt ook reeds m.b.v. TEX aangemaakt. Heeft eigen macro's geschreven t.b.v. eigen layout.
Proberen TEX te implementeren op PS2 systemen. In principe hierbij geen problemen doch bij het previewen van grote files (4 à 5 blz) hangt het systeem.

Theo Jurriens (RUG)
Is 1.5 tot 2 jaar intensief met TEX bezig. Heeft contacten met uitgeverij. Ervaring met 3-koloms uitvoer inclusief plaatjes.
Software wordt gebruikt door het secretariaat na een eenvoudige scholing en enige aanpassingen. Heeft een LATEX voorbeeldenboek doen verschijnen. Heeft zicht verdiept in METAFONT op Macintosh i.v.m. het genereren van speciale symbolen. Heeft programma gemaakt om etiketten op laserprinter te genereren.
Vindt LATEX op sommige punten niet consequent in het gebruik. Heeft sterrenkaarten gemaakt m.b.v. een Picture Environment. Wil een Picture Environment maken voor de Atari systemen. Vindt dat drukkerijen en zetterijen zich wat op een afstand houden m.b.t. TEX/LATEX, ondanks het feit dat de software de werkzaamheden met ongeveer 40% kan reduceren.

Alma (RUG)
Werkt ongeveer 1.5 jaar intensief met TEX en LATEX. Speciale interesse ligt in de taalkundige aspecten van het pakket. Hij heeft onlangs een boek gemaakt met wat zonder TEX niet mogelijk geweest zou kunnen zijn.

Piet Tutelaers (TUE)
Heeft veel tijd in Troff gestoken (VAX-750; UNIX; QMS-1200 laserprinter). Vindt TEX op veel punten toch beter, naast het feit dat het op veel meer systemen beschikbaar is.
Het is hem nog niet gelukt om op de QMS-1200 met Troff of met TEX, fonts te downloaden.
Noemt de problemen bij de wit en de zwart schrijvende laserprinters. Noemt het feit dat de fontkwaliteit afhankelijk kan zijn van het type laserprinter.
Op dit moment is men in het bezit van de QMS-800, welke op PostScript gebaseerd is.
Wil op alle locale UNIX systemen zowel TEX als LATEX draaien. Is positief over de PostScript mogelijkheden. Vindt als nadeel van de Beebe drivers dat ze niet de hardware fonts van het output device ondersteunen.
Haaijer vermeldt daarop dat de ArborText drivers (geen Public Domain) dit wel doen.
LATEX gebruik groeit i.t.t. gebruik van Troff.
Overweegt een site licentie voor PC-TEX aan te schaffen, daar het blijkt dat bij 3-4 TEX gebruikers, het VaX systeem overbelast wordt. Wil de divers outputmogelijkheden binnen de TUE coördineren.

Niek Cox (URC)
Werken ongeveer 1.5 jaar met TEX. Heeft via de afdeling Natuurkunde de Stanford tape ontvangen en geïnstalleerd. Het gebruik blijkt zich nu ook uit te breiden naar afdelingen die zich met de taalwetenschappen bezig houden.
TEX/LATEX is geinstalleerd op IBM. Het wordt tevens op Atari gebruikt. Heeft nauwelijk iets aan eigen ontwikkeling gedaan. De locale ondersteuning is nog minimaal.
Heeft interesse in fonts en Nederlandstalige documentstijlen. Is zeer geïntereseerd in aanwezig cursusmateriaal over LATEX.

Cor de Ridder (Philips)
Medewerker van de software afdeling. Ongeveer 50 gebruikers werken met de software vooral bij het maken van documentatie.
Het produkt is ongeveer 2 jaar in gebruik. Draait op VAX/VMS met LN03 als output device. Wil ook de Xerox-4050 aansturen.
Is geïnteresseerd in de Nederlandse style files en in de mogelijkheden van het plaatsen van plaatjes op de juiste bladzijden.

TOT SLOT

Kees van der Laan merkte de afwezigheid op van zowel CWI als SARA. Van CWI vermeldde hij dat zij recentelijk naast ditroff, naar TEX waren overgegaan en kennelijk nog geen tijd en mogelijkheden hadden zich bij de TEX gebruikersgroep aan te sluiten.
SARA gebruikt TEX (o.a. tijdschrift Supercomputing). Een bericht van verhindering is onlangs van hun ontvangen.

Samenvattend kon worden gezegd dat er reeds veel goede software aanwezig is. Er ontbreekt echter ook nog één en ander. Genoemd werd ondermeer de SGML-TEX relatie en Picture environment/integratie.
Individuele werkzaamheden zouden ondersteund moeten worden. Voor groepswerkzaamheden is duidelijk belangstelling. Een centrale plaats zou er moeten zijn waar men kan informeren wat er is en wat er gedaan wordt.
TEXHaX is ook een goed uitwisselings medium, doch mogelijk op een te wat groot niveau. De TEX-NL Listserver zou daarentegen een goede oplossing zijn.

Cox (URC) meldde de aanwezigheid van een eerdere actieve Nederlandse TEX groep welke twee keer bij elkaar geweest zou zijn. Namen werden daarbij genoemd van Ad Gerards, Vogt, Poppelier en de commissie CVDUR.
De voorzitter had vernomen dat deze groep gestopt was mede vanwege gebrek aan tijd. Afgesproken werd om contact op te nemen met genoemde mensen. De status (eventueel opheffen) van deze groep zou duidelijk naar buiten gebracht moeten worden. Mogelijk kan de huidige TEX gebruikersgroep gebruik maken van hun adressenbestand.

Nogmaals werd opgeroepen om informatie die bij individuele personen aanwezig is en van belang kan zijn voor de groep dit via Gerard van Nes door te sluizen.

3 Brainstorming TEX aandachtsgebieden

4 Instellen werkgroepen voor de aandachtsgebieden

Een aanzet voor mogelijke aandachtsgebieden was reeds in de agenda opgenomen. De uiteindelijke lijst (let op de nieuwe nummering) samen met de belangstellenden, is als bijlage A bij dit verslag opgenomen. Bij de lijst zijn tevens de coördinatoren genoemd. Zij zijn ook de contactpersonen van de betreffende groepen.

De volgende opmerkingen werden m.b.t. de aandachtsgebieden nog gemaakt:

6. Lijst fotozetters:
   Volgens Theo Jurriens kan 'Jolly Text' in Den Haag wel TEX verwerken. Mogelijk is er ook in het buitenland ervaring in deze. Braams vermeld dat Springer een .STY file distribueert.
   Reidel (Dordrecht) schijnt zich tevens op dit vlak bezig te houden.
9. Integratie beelden en TEX:
   Genoemd werd dat dit gerealiseerd kon worden via het \special commando. Drivers van ArborText blijken dit goed te ondersteunen.
10. Op de vraag of er een standaard SGML aanwezig is, werd verwezen naar de ISO-8879.
    Vermeld werd dat de draft wel anders was dan de standaard zelf.

Macintosh gebruikers/geïntereseerden blijken ondermeer Helmig (Philips) en Cox (URC) te zijn. Een werkgroep hiervoor te starten bleek nog niet zinvol te zijn (wel zouden individuele bijdragen toegejuigd worden).

CMS perikelen werden vermeld door van den Brekel (Philips); vooral op het ASCII-EBCDIC gebied. Cox (URC) blijkt reeds één en ander opgelost te hebben. Dit mogelijke aandachtsgebied leek minder geschikt om buiten de individiuele werksfeer te halen.

N.a.v. een verzoek voor een EHBO post voor TEX en LATEX vragen/problemen, stelde Theo Jurriens (RUG) zich vrijwillig beschikbaar.
Daarnaast bestaat de mogelijkheid het op TeX-NL te zetten.

Genoemd werd de mogelijke invoering van "Traps en Tips" faciliteiten. Vooralsnog zou een ieder die in dit kader wat heeft, het kunnen indienen bij van Nes (ENR) die dan voor verdere distributie zal zorgdragen.

Informatie uitwisseling, vooral naar de nieuwe gebruikers op het TEX gebied, werd zeer belangrijk bevonden. Het zou ook één van de taken van de gebruikersgroep moeten zijn (zie ondermeer werkgroep "gebruikersdag").

Wederom kwam de TEXHaX etc. naar voren als goede informatiebron van veel TEX en aanverwante zaken.

Voor het uitwisselen van informatie werd het EARN en DECnet gekozen. De meeste van de TEX aanwezigen zijn op zo'n netwerk aangesloten, dan wel kunnen van de faciliteiten gebruik maken (via FIDO-net of via een dial-up met de dichtsbijzijnde universiteit).
Er is ook een BITNET-UUCPnet koppeling (CWI). Braams (PTT Neher Lab) is bezig met het schrijven van een handleiding over netwerken.
Eventuele mogelijkheden van de mcvax als gateway zullen ook nader bekeken worden.

Ook werd de mogelijkheid van de Listserver genoemd. Informatie werd door Braams (PTT Neher Lab) gegeven hoe men zich hierop als abonnee kan laten inschrijven en hoe informatie verstuurd kan worden (zie bijlage B). Genoemd werden de artikelen over dit onderwerp in laatste nummers van SURF-bulletin.

5 Formele zaken

a. Vaststelling bestuur:
Voor de komende twee jaar werd als bestuur voorgesteld:
C.G. van der Laan (RUG), voorzitter
G.J.H. van Nes (ENR), secretaris
J. Braams (PTT Neher Lab)
H.P.A. Mulders (KUB)
Het bestuur zal een voortrekkersrol vervullen.

b. Vergaderfrequentie:
Uitwisseling gegevens indien mogelijk zoveel mogelijk via E-mail (listserver TEX-NL).
Bijeenkomst in principe twee maal per jaar.

c. Informatie uitwisseling:
Via listserver (zie bijlage B), EARN en/of DECnet.
Zie ook onder agendapunt 3/4.

d. Ledenlijst:
Het bestuur stelt een ledenlijst op en zal deze distribueren.

6 Rondvraag

Brouwer (EGD) deeldt mede dat van 18-20 juli de Europese TEX gebruikersbijeenkomst in Engeland wordt gehouden.
Aanwezig zullen zijn Abdy Jooya, Jan van Knippenberg en Marius Broeren.

Stemerdink (LUW) zou graag informatie willen ontvangen van ondermeer TEXHaX, UK-TEX gebruikersbijeenkomst etc., mogelijk bij de volgende vergaderstukken.

Soutberg (Elsevier) wil graag meer bekendheid over de adressen van de leveranciers waar TEX en aanverwante software te verkrijgen is; wat er allemaal nodig is om met TEX te beginnen; en aanverwante zaken.

Broeren (Océ) kijkt belangstellend uit naar de rapportage van de werkgroepen.

Krijnen (ACCU) informeert bij Mulders naar het produkt DRILCOM, zoals was beschreven in het laatste nummer van het universiteitsblad CONVEX courier.
DRILCOM kan ondermeer WP, ASCII en WordMARC files converteren naar TEX. Het is in PASCAL geschreven en draait op een VAX systeem. Het produkt is nog in ontwikkeling. Condities van distributie is niet bekend. Er is een rapportje.
Van der Laan (RUG) merkt hierbij op dat het interessant is om dit pakket aan de RUG-testvoorbeelden te onderwerpen. Deze (uitbreidbare) testcollectie is gebruikt bij het beproeven van K-talk. Een (Engels) rapport over de evaluatie van K-talk is in voorbereiding.

Braams (PTT) is bezig een DSR (Digital Standard Runoff) naar LaTeX converter te testen. Mogelijk zijn er andere belangstellenden die hem willen helpen hiermee.

7 Sluiting

De volgende bijeenkomst is op:

donderdag 24 november 1988 bij ENR te Petten

(en niet op 11 november zoals eerst was aangekondigd)

BIJLAGE A

Werkgroepen Nederlandse TEX Gebruikersgroep

1. **Cursusmateriaal (algemeen)**
   N.G. Cox
   G. Haayer (Styx Publications)
   A.R. Jooya (Open Universiteit)
   C.G. van der Laan (RUG) (coördinator)
   P. Tutelaers (TUE)

2. **Richtlijnen voor publicaties/conferentie proceedings etc.**
   T.A. Jurriens (RUG; Sterrenkunde) (coördinator)
   J. Röling (Styx Publications)
   J. Soutberg (Elsevier Science Publisher)

3. **Evaluatie produkten (Ned. LATEX incl sty. files en afbreekregels; andere macrocollecties AMSTEX; converters K-talk; TEX naar ASCII; index programmatuur; dBase-TEX koppeling; adreslabels; verkrijgbaarheid etcetc.)**
   J. Braams (PTT-Neher Lab) (coördinator)
   M.A.J.H. Broeren (Océ Nederland B.V.)
   G. Haayer (Styx Publications)
   J.R. Luyten (RUG)
   H.P.A. Mulders (KUB)

4. **Fonts (gebruik van Metafont)**
   H. Brouwer (EGD)
   N.G. Cox (URC)
   G. Haayer (Styx Publications) (coördinator)
   A.J. de Meyer (RUU; Wiskunde)
   J. Röling (Styx Publications)
   P. Titelaers (TUE)
   F.J. Velthuis (RUG; Rekencentrum)

5. **Drivers, preview, printers, postscript**
   J. Braams (PTT Neher Lab)
   H. Brouwer (EGD)
   J.S.M. van Knippenberg (Océ) (coördinator)
   P. Titelaers (TUE)

6. **Lijst en link met fotozetters**
   G. Haayer (Styx Publications)
   M. Helmig (Philips)
   T.A. Jurriens (RUG; Sterrenkunde)
   F.J. Velthuis (RUG; Rekencentrum) (coördinator)

7. **PC-perikelen; campus licentie etc.**
   G. de Jong (EU)? (coördinator)

8. **Nederlandse TEX gebruikersdag (juni 1989?)**
   A.P.J.M. Krijnen (ACCU) (coördinator)

9. **Integratie beelden en TEX**
   H. Brouwer (EGD)
   N.G. Cox (URC)
   G.D. Draaijer (MARIN)?
   M. Helmig (Philips)
   T.A. Jurriens (RUG; Sterrenkunde) (coördinator)
   N.A.F.M. Poppelier

10. **SGML-TEX relatie**
    C.G. van der Laan (RUG) (coördinator)
    J. Grootenhuis (CIRCE)
    J. Bleeker & J. Soutberg (Elsevier Science Publisher)

11. **Picture environment; Preview op Atari**
    T.A. Jurriens (RUG; Sterrenkunde) (coördinator)

12. **Beheerders handleiding/documentatie**
    J. Braams (PTT Neher Lab)
    E.J. Evers (RUU; Geneeskunde) (coördinator)
    G.J. Stemerdink (LUW; Rekencentrum)

Doel van de werkgroepen is:
- het verzamelen van informatie materiaal
- onderzoek doen op het deelgebied
- fungeren als kennis databank

Resultaten worden via de coördinator doorgestuurd naar Gerard van Nes, die (voor de komende bijeenkomst) voor verdere distributie zal zorgdragen.

BIJLAGE B

Listserver

Het aansluiten bij de TEX-NL afdeling van de onder EARN vallende LISTSERVER gaat als volgt:

a. Men dient zich eerst als abonnee in te schrijven via:

```
SEND  LISTSERV@HEARN  SUBSCRIBE TEX-NL "<uw eigen naam>"
```

Dit geldt alleen voor sites die JNET geïnstalleerd hebben.

Indien men geen JNET bezit geldt het volgende:
- maak een file (TEX.TXT) met een regel:

```
SUBSCRIBE  TEX-NL  "<uw_eigen_naam>"
```

- en vervolgens:

```
$ MAIL
  SEND  TEX.TXT
  To : SARA5::INM%"LISTSERV@HEARN.EARN"
  Subject : <blanco>
  etc.
```

b. Vervolgens kan men informatie versturen via bijvoorbeeld:

```
SEND                          SEND/FILE
To: TEX-NL@HEARN              File: .....
Subject- .....                To: TEX-NL@HEARN
etc.                          Subject: .....
```

Niet JNET bezitters maken dan gebruik van de MAIL faciliteit.

Aangeraden wordt echter bij het gebruik van TEX-NL in principe alleen gebruik te maken van MAIL en niet met het versturen van files als zodanig.

Laatste Nieuws

Op het commando:

```
$ SEND LISTSERV@HEARN subscribe tex-nl "Donald Knuth"
```

reageert de listserv als volgt:

```
(HEARN)LISTSERV - * List TEX-NL is not open for automatic subscription.
(HEARN)LISTSERV - * Your request has been forwarded to the list owner(s):
(HEARN)LISTSERV - *  NIEK COX <U070007@HNYKUN11>
```

BIJLAGE C

TEXHAX

```
------------------------------

%%%
%%% Concerning subscriptions, address changes, unsubscribing:
%%%     BITNET: send a one-line mail message to LISTSERV@TAMVM1.BITNET:
%%%         SUBSCRIBE TEX-L <your name>    % to subscribe
%%%
%%%     All others: send mail to
%%%           texhax-request@score.stanford.edu
%%%     please send a valid arpanet address!!
%%%
%%%
%%% All submissions to: texhax@score.stanford.edu
%%%
%%% Back issues available for FTPing as:
%%%          machine:      directory:  filename:
%%%   [SCORE.STANFORD.EDU]<TEX.TEXHAX>TEXHAXnn.yy
%%%      nn = issue number
%%%      yy = last two digits of current year
%%%\bye
%%%

------------------------------
```

BIJLAGE D

TEX Publications Information

TeXMaG is an independantly published electronic magazine available free of charge to all interested parties reachable by electronic mail. It is published sporadicly, and the editor likes to think that its monthly so the readers humor him. Subscription requests should be sent to Don Hosek <DHOSEK@HMCVAX.BITNET> or send the following message to LISTSERV@BYUADMIN: SUBS TEXMAG-L Your_Full_Name. European subscribers may send the SUBS command to LISTSERV@DEARN, subscribers on CDNnet should send subscription requests to <list-request@ubc.csnet> (being sure to mention that they wish to subscribe to TeXMaG), and JANET subscribers should send requests to be added to the list to Peter Abbott, <ABBOTTP@UK.AC.ASTON.MAIL>. Back issues are available for anonymous FTP in the file BBD:TEXMAG.TXT on SCIENCE.UTAH.EDU. BITNET users may obtain back issues from LISTSERV@TCSVM (in an interactive message or as the first line of a mail file, send the command GET TEXMAG VvNn where v is the volume number and n is the issue number), or from UBSERVE@UBVMSC (in an interactive message, send the command SEND TEXMAG.VvvvNnnn where vvv and nnn are as above). Janet users may obtain back issues from Peter Abbott (e-mail address above) and DECNET/SPAN users may obtain them from the Decnet repository (see below). They may also be obtained from Don Hosek <DHOSEK@HMCVAX.BITNET>. Article submissions, contributions for the Toolbox, and letters to the editor are always welcome and should be sent to <DHOSEK@HMCVAX.BITNET>.

Other publications of interest to TeX users are:

TeXHAX. Arpanet mailing list for persons with questions, suggestions, etc.. about TeX, LaTeX, MetaFont and related programs. Submissions for this list should be sent to <TeXHAX@Score.Stanford.EDU>. Internet subscribers may subscribe by sending a request to <TeXHAX-REQUEST@Score.Stanford.EDU>. JANET subscribers should send subscription requests to <texhax-request@uk.ac.ucl.cs.nss>. BITNET users may subscribe by sending the following command (as an interactive message or as the first line of a mail message) to LISTSERV@TAMVM1: SUBS TEX-L your_full_name. The list is peer-linked to other listserves in the United States and Europe. Australian users should send subscription requests to <munnari!elz@uunet.uu.net> Japanese users should send subscription requests to <takagi%icot.jp@relay.cs.net>. Back issues are available by anonymous FTP from Score.Stanford.Edu and from Listserv@Tamvm1 (in an interactive message or as the first line of a mail file send the command GET TEXHAXnn yy where nn is the issue number and yy are the last two digits of the year. Issues 100 and above are named TEXHAnnn yy)

Unix-TeX. Arpanet mailing list specifically for users of TeX under the Unix operating system. Submissions for this list should be sent to <Unix-TeX@JUNE.CS.WASHINGTON.EDU>. Requests to be added or deleted from the mailing list should be sent to <Unix-TeX-Request@JUNE.CS.WASHINGTON.EDU>.

UKTeX. A U.K. version of TeXhax. To subscribe, send a note to Peter Abbott at <info-tex-request@uk.ac.aston.mail>.

TeXline. A TeX newsletter edited by Malcolm Clark. To subscribe, send a note to <texline@uk.ac.ic.cc.vaxa>.

TUGBoat. A publication by the TeX Users Group. An excellant reference for TeX users. For more information about joining TUG and subscribing to TUGBoat send (real) mail to:

TeX Users Group
c/o American Mathematical Society
P. O. Box 9506
Providence, RI 02940-9506, USA

Inquiries may be directed to Karen Butler <KLb@Seed.Ams.Com>.

LaTeX-style collection. A collection of LaTeX files is available for FTP and mail access at cayuga.cs.rochester.edu. To obtain files via FTP, login to cayuga.cs.rochester.edu (192.5.53.209) as anonymous, password guest and go to the directory public/latex-style (where the files are). The file 00index contains a brief description of current directory contents. If your site does not have FTP access, you may obtain files by mail by sending a message to latex-style@cs.rochester.edu with the subject "@file request". The first line of the body of the message should be an @. The second line should contain a mail address from rochester TO you (for example, if you are user@site.bitnet, the second line should be user%site.bitnet@cunyvm.cuny.edu). The lines that follow should be the filenames you desire and the last line should also contain only an @.

LISTSERV@DHDURZ1 has file archives of interest to TeX users. Included are the Beebe drivers and contents of the LaTeX style collection, as well as some TeX macros. Many files are available only in German.

LISTSERV@TAMVM1 also has file archives that may be of interest to TeX users on BITNET, including the files from the Score.Stanford.EDU FTP directories and back issues of TeXHAX. For a list of files available, send the following command to LISTSERV@TAMVM1: GET TeX FILELIST.

DECNET. There is a TeX file collection on DECnet accessible from DECnet and SPAN. Available files include the Beebe DVI drivers, the LaTeX style collection, and back issues of TeXhax, TeXMag, and UKTeX. For more information, contact Marisa Luvisetto (DECNET: <39947::luvisetto>, Bitnet: <LUVISETTO@IBOINFN.BITNET>) or Massimo Calvani <CALVANI@VAXFPD.INFNET> U.S. Users should contact Ed Bell <7388::Bell>

```
JANET. Peter Abbott keeps an archive of TeX-related files available
for FTP access. For more information send mail to
<Abbottp@Uk.Ac.Aston.Mail>.
```

Bijlage 6
Het zetten van wetenschappelijk werk: 1973 vs 1998
(Monotype vs TEX)

Gerrit Oomen
Kluwer Academic
Publishers, Dordrecht

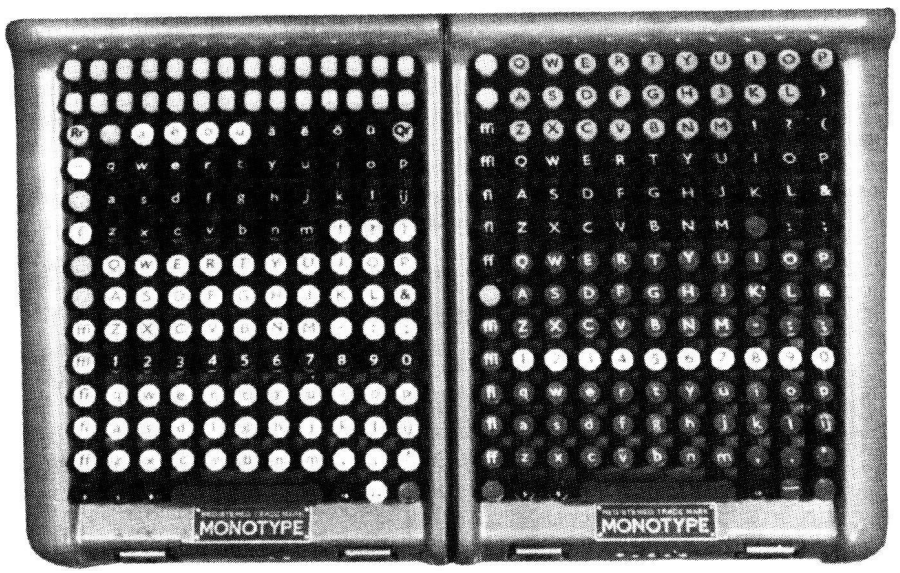
Hoe eenvoudig het zetten van wetenschappelijk werk heden ten dage is weten we inmiddels allemaal wel. Vele computersoftware is daar uitermate voor geschikt. TEX(LATEX) is daar beslist wel het meest geschikt voor. Formules, tabellen, het plaatsen van figuren alles kan een auteur al vanaf zijn bureau d.m.v. een PC de volledige vorm aan geven, zonder dat er ook maar een grafisch vakman aan te pas behoeft te komen. Hoe wetenschappelijk werk in TEX(LATEX) gezet dient te worden behoeft hierbij geen nadere uitleg.

Een kleine vijf en twintig jaar geleden (en wat is nu eigenlijk 25 jaar) was dat nog heel anders gesteld. Hoewel de fotozetmachine, waar wetenschappelijk zetwerk op verricht kon worden, al een klein beetje in opmars was. Echter van een totaal product was zeker nog geen sprake van.

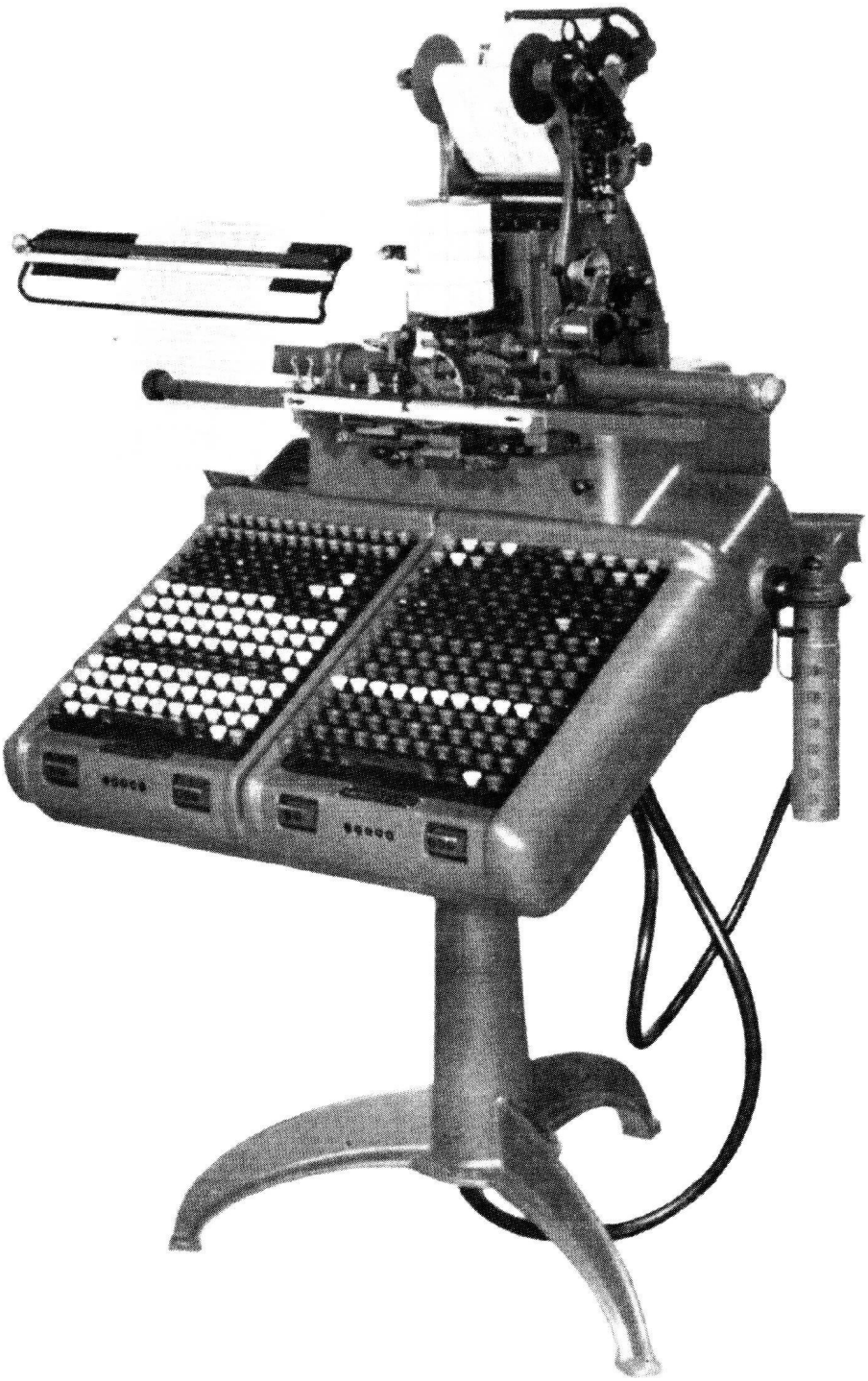
De Monotype-installatie (het zetten met lode losse letters) was uitermate geschikt voor het vervaardigen van meer ingewikkeld werk, omdat het eindproduct uit losse letters bestond. Hieronder vielen dan staat- en tabelwerk, formules e.d. Ook de materiaalvoorziening van de handzetterij speelde de Monotype-machinezetterij een belangrijke rol.

We zullen hieronder achtereenvolgens de verschillende machines onder de loep nemen.

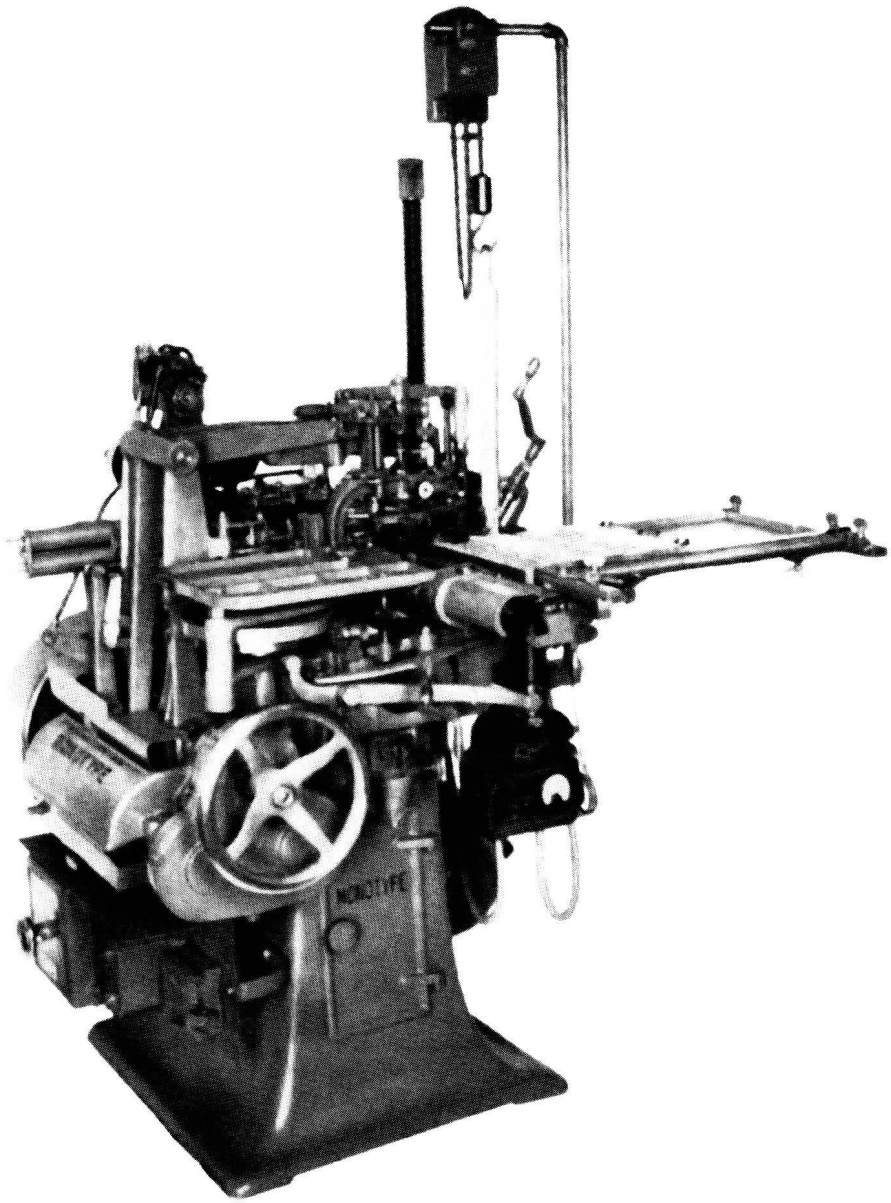
Het monotype-toetsenbord Voor het aanbrengen van de perforaties in de rol papier. Er waren twee soorten hierin, het zgn. D-toetsenbord met enkele papiertoren, en het DD-toetsenbord met een dubbele papiertoren. Het laatste was in een aantal gevallen gemakkelijk omdat men dan de werkzaamheden kon combineren. Men kon met een D-toetsenbord maar 1 lettercorps tegelijk zetten. Alle kleinere en afwijkende lettersoorten moesten op



Figuur 1. Twee losse toetsenramen.



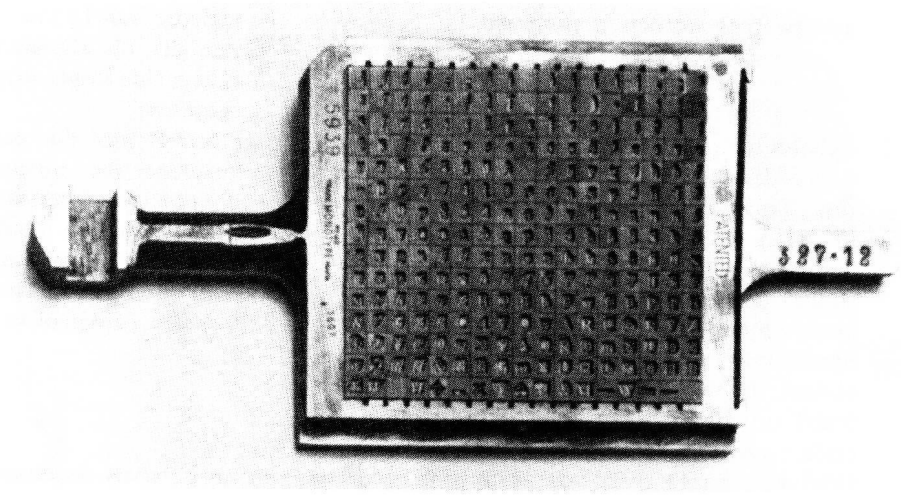
Figuur 2. Het D-toetsenbord.



Figuur 3. De zetselgietmachine.

een aparte papierrol getikt worden en op de handzetterij werd de opmaakt verricht door met de hand de verschillende lettercorpsen, zoals voetnoten, citaten, kop- en voetregels, paginacijfers met de hand toe te voegen aan de pagina-opmaak. Voor formules was het wel een zeer apart verhaal waar we later nog op terug komen.

De monotype-zetselgietmachine Op de zetselgietmachine werden de afzonderlijke letters, spaties, tekens (tot maximaal corps 14) gegoten in de volgorde zoals die door de toetsenbordbewerker d.m.v. de perforaties in de papierrol op het toetsenbord werden ingetikt.



Figuur 4. Matrijzenraam met 255 matrijzen.

Door het monteren van een grootkorpsapparaat konden de mogelijkheden van de zetselgietmachine worden uitgebreid voor het zetten van lijnen en interlinies. Men kon ook nog een Supra compleetgietmachine aanschaffen (wat eigenlijk onontbeerlijk was) voor het gieten van letter en letterwit van 5 t/m 72 punten, tabelwit, spaties en kwadraten, korte lijntjes, holwit (opvulwit), en clichéwit.

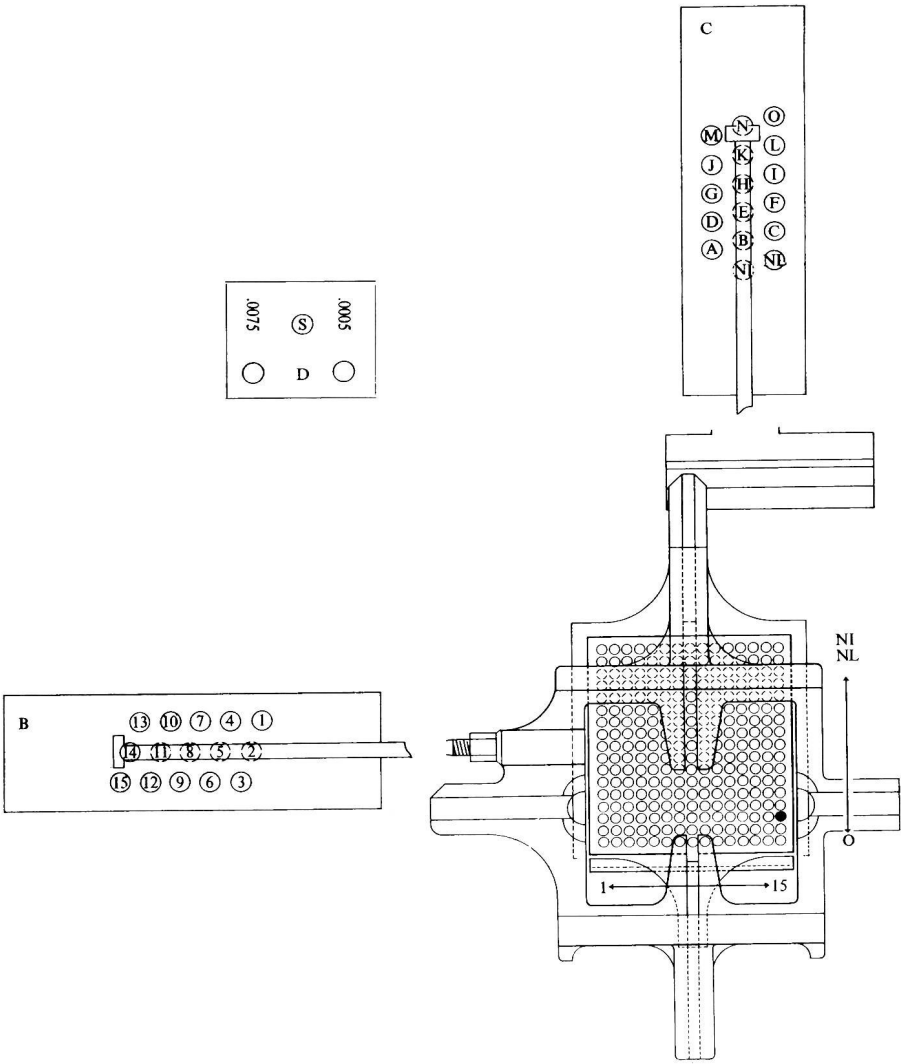
Op deze afdeling was het altijd zeer luidruchtig door het stampen van de gietmachines. De medewerkers droegen toen de tijd ook allemaal oordoppen en men kreeg ook elke dag een liter melk aangeboden van het bedrijf. De reden hiervoor moge duidelijk zijn. Lood was nu niet het allergezondste materiaal om dagelijks mee geconfronteerd te worden.

De luchtpomp De luchtpomp, die doorgaans in de gieterij stond opgesteld, leverde de samengeperste lucht, die nodig was voor het functioneren van de toetsenborden en de gietmachines.

Het matrijzenraam De wijze waarop de matrijzen in het matrijzenraam waren geplaats was, zoals uit Figuur 1 te zien is, afhankelijk van de breedte van de letters. Een normaal matrijzenraam bevatte 255 losse matrijzen, nl. 15 rijen van 17 stuks. Elke matrijs was aan de beeldzijde 0, 2 × 0, 2 inch en dus zuiver vierkant. De horizontale rijen waren genummerd van 1 t/m 15 en de verticale rijen werden aangeduid door de letters NI, NL, en de eerste vijftien letters van het alfabet (zie Figuur 2).

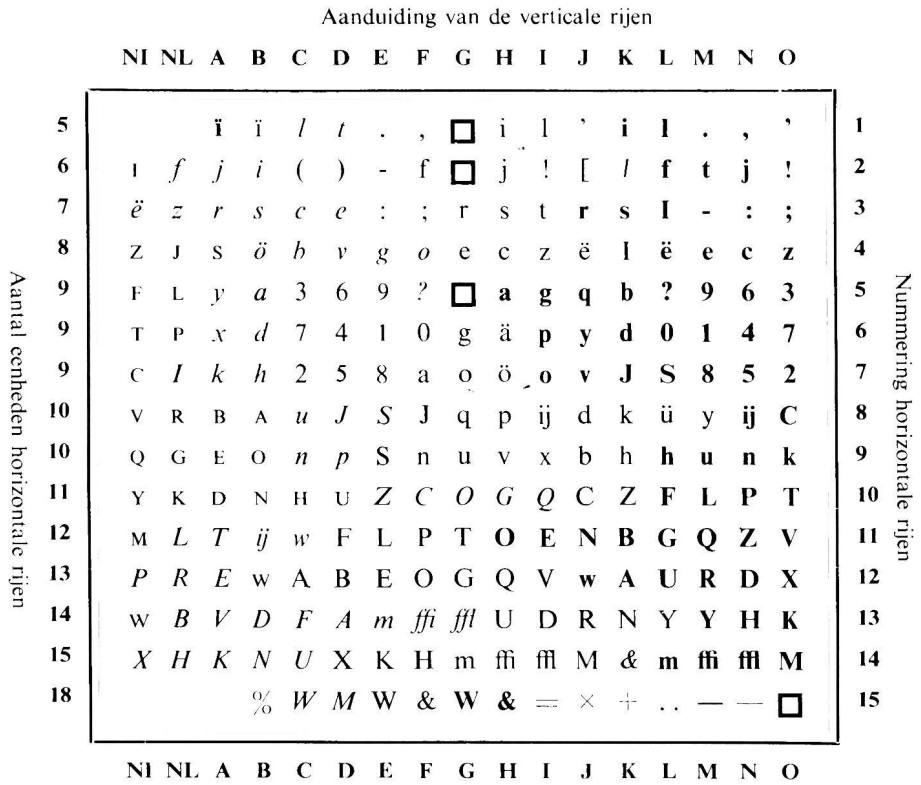
Indien men een ander teken tijdens het tikken tegenkwam, moest men opgeven welk teken uit het raam verwijderd moest worden. Ook moest men dan eerst opzoeken hoe breed het te vervangen teken was, de breedte van de letter werd aangegeven in eenheden. Achttien eenheden was een vierkant (0–15 uit schema). Dit werd dan op de rol genoteerd. Voor wetenschappelijk werk was het dan vaak zo erg, dat men om de vier à vijf manuscript pagina's een nieuwe rol (zeg maar rolletje) een nieuwe lijst moest maken. De gieter moest dan (met een loep vaak) de koperen matrijzen vervangen. Dit gold natuurlijk ook voor elke superieur en inferieur letter.

Het tikken van wetenschappelijk werk Op het toetsenbord (zie Figuur 3) waren 255 toetsen aanwezig waarvan men gebruik kon maken. Men kon zonder wisseling romein, cursief, en vet zetten in zowel onderkast als kapitaal. Ook had men nog de beschikking over klein-kapitaal en een aantal geaccentueerde letters. Deze laatste posities werden vooral gebruikt ter vervanging van de letters die nodig waren voor het wetenschappelijk



Figuur 5. Schematische voorstelling van de penblokken en het in positie brengen van het matrijzenraam.

werk, zoals grieks, superieure en inferieure letters, bijzondere tekens e.d. Indien een teken niet zoveel voorkwam, werd er een "vuile letter" voor getikt. Een vuile letter was een teken wat duidelijk opviel dat het er niet thuis hoorde en in de marge van het manuscript werd dan met een potlood een kruisje gezet. De handzetter die later de pagina's in lood ging opmaken (pagina's op lengte maken en andere korpsen) tussenbracht kon dan zien dat hij hier iets moest vervangen. Hij haalde dan de speciale letters of tekens uit n van de vele letterkasten. Men moest ook heel goed rekening houden met overhangende letters zoals bijvoorbeeld de I, J, T, Y cursieve letters. Deze moesten aangespatieerd worden zodra er direct iets tegenaan zou komen (bijv. superieure letters) anders werden de letters beschadigd en braken dus af. Als er in het wetenschappelijk werk inferieur en superieur boven elkaar moest komen, dan moest hier ook een "vuile letter" voor getikt worden. Men moest dan eerst uitrekenen hoe breed die letter (of letters) waren in de 6 punten



Figuur 6. Schematische voorstelling van een matrijzenraamindeling met 255 matrijzen. De vierkantjes zijn vastwit in eenheden.

en daarna omrekenen naar de 12 punten. Op het manuscript werd dit ook weer in potlood aangetekend en de handzetter wist dan wat hij hier moest doen. Alle 6 punts letters werden nadien op een aparte rol getikt en gegoten. De gieter moest hier een olieachtige substantie aan toevoegen omdat deze letters zo klein waren dat ze bijna niet op te pakken waren. Door de substantie plakten de letters een beetje aan elkaar.

Het zetten van formules met een teller en een noemer was een vak apart. Er waren voldoende tikkers die plat werk konden tikken, echter formule-tikkers waren een uitzondering.

Een vrijstaande formule werd achtereenvolgens gezet. Het meest gebruikte korps was 10 op 12, dus een letterbeeld van 10 punten op een voet van 12 punten. Onderstaande formule werd als volgt getikt op de monotype machine:

$$J(N, H) = \quad | S(\alpha) - T(\alpha)|^2 \, L(\alpha) \, d\alpha + O \quad \log^2 H + N \quad + O'(HN),$$
$$H^3 L^2$$

$$J(N, H) = \int_{-1/2}^{1/2} |S(\alpha) - T(\alpha)|^2 L(\alpha) d\alpha + O\left(\frac{H^3 L^2}{\log^2 H + N}\right) + O'(HN),$$

Hierin is H^3L^2 de teller en $\log^2 H + N$ is de noemer. Op de handzetterij werd alles in het midden gezet door zeven punten wit er boven en zeven punten wit eronder te leggen. Het breuklijntje werd er tussen gelegd en met de integraal en de parentheses gebeurde

het zelfde. De $1/2$ en $-1/2$ boven en onder de integraal werd er in een zespunts regel boven en onder gelegd. Men kwam dus op een totaal van 26 punten uit (het lijntje was twee punten dik). De letter uit de noemer werd gecentreerd t.o.v. de teller gezet. De tikker tekende dan met potlood op zijn em-liniaal aan waar de teller begon en eindigde en kon zo de "enkele uitvulling" berekenen die hij dan moest aanslaan om dit gecentreerd te krijgen. Indien de noemer breder was dan de teller, moest hij in de bovenste regel die eerst tikken en de handzetter moest het dan met de hand op zijn galei omwisselen. Dit is nog maar een heel eenvoudig voorbeeld. Laat staan hoeveel moeite en reken werk het kostte om een ingewikkelde formule te tikken, met o.a. integralen, sommatie tekens, e.d. De tikker had hier heel veel werk aan (20 manuscript pagina's per dag was al een heel goed resultaat), de gieter was hier geruime tijd mee bezig, de handzetter die de pagina's op moest maken kwam ook niet hoger uit dan 10 tot 12 opgemaakte pagina's per dag. En daarna kwam de correctie fase dan nog. Geen wonder dat het zetten van wetenschappelijk werd toen zo duur was en dat er maar enkele gespecialiseerde bedrijven waren in Nederland die dit werk konden verrichten. En thuiswerken was er al zeker niet bij.

Het maken van een proef-afdruk De opgemaakte pagina's (opgebonden met een touwtje) werden in een vorm van 8 pagina's (schoon en weerdruk) uitgezet op een proefpers. Deze werden ingesmeerd met inkt. Het papier werd op de rol geklemd en d.m.v. een cilinder over het zetsel bewogen. Hierdoor kwamen de letters op het papier te staan.

Bijlage 7
Typografische scanning

Taco Hoekwater
taco.hoekwater@wkap.nl

abstract

Dit artikel geeft een beschouwing over ,,Nederlandse" typografie, aan de hand van een aantal scans van in Nederland gepubliceerde boeken en tijdschriften uit de afgelopen drie eeuwen. Het artikel laat een aantal traditionele typografische vormen zien, met toevoeging van wat commentaren over het gebruikte proces.

keywords

Typografie, nederland, overzicht

Inleiding

Iedereen die met TEX werkt zal langzamerhand al de kreten in de trand van ,,vroeger deden we dat anders" en ,,zo hoort dat niet" wel kennen. Veelal komen deze uitspraken niet van de ,,oude rotten" in het vak (loodzetters & inbinders), maar van mensen die doorgeleerd hebben in het leveren van kritiek op andere mensen (uitgevers, bureau-redakteurs en neerlandici). Het is nog maar de vraag in hoeverre deze mensen recht van spreken hebben.

Het artikel van Gerrit Oomen elders in deze MAPS laat duidelijk zien dat in de periode die zij als ‚vroeger' aanduiden de situatie ook verre van koek en ei was. Als iets typografisch al perfect was, dan ging dat toch zeker gepaard met een hoop moeite en tijd. Zoals ook in vrijwel alle andere vakgebieden die grotendeels gebaseerd zijn op ervaring en kennisoverdracht, zo is ook de kwaliteit van zetwerk in eerste instantie een kwestie van economie: goed zetwerk door een professional met jarenlange training kost aanzienlijk meer geld dan ‚redelijk' zetwerk, gedaan door iemand die feitelijk nog van alles moet leren.

Evenmin als zetwerk vroeger een kwestie was van lukraak loden letters op een persblok stapelen, evenzo is zetwerk *anno nu* niet alléén goed kunnen werken met het programma dat we hebben (TEX dus). Minstens net zo belangrijk is een goed oog voor wat typografisch wél mooi is en wat niet, een kunst die zich vaak pas na een paar jaar zetten begint te ontwikkelen.

Als ik het hierboven heb over ,,goed kunnen werken", dan bedoel ik de mate waarin de gebruiker TEX beheerst. Het is bedroevend op te moeten merken hoeveel mensen TEX gebruiken, maar feitelijk *zelf* bestuurd worden door LATEX, waar eigenlijk *zij* LATEX zouden moeten besturen. Zulke mensen zijn geen zetters, maar auteurs. De zaak van een auteur is ervoor te zorgen dat de inhoud van een tekst interessant is, terwijl de taak van de zetter is ervoor te zorgen dat die tekst zo goed mogelijk overkomt op het gekozen medium (gewoonlijk papier).

Uiteraard kan een auteur wel een zetter worden. Na grondige bestudering van het TEXbook en de LATEX sources is het wel degelijk mogelijk om vrijwel alles gedaan te krijgen. Als dat gelukt is, wordt het tijd om te leren hoe het ,,zou moeten".

Vervolgens dient zich de vraag aan van *wie* we zouden moeten leren. Moeten we kijken naar ,,hoe het vroeger ging" en stug volhouden aan tradities die waarschijnlijk ouderwets overkomen en wellicht zelfs lelijk zijn? Of moeten we zelf beslissen wat we mooi vinden en vervolgens onze mening over schoonheid opdringen aan de lezer, met het gevaar dat we het helemaal mis hebben?

En als we vasthouden aan ,,de traditie", over welke traditie hebben we het dan? Vandaar dit artikel: in principe gewoon een stel scans van pagina's uit oude boeken[1], met wat commentaar toegevoegd. Hoewel ik liever niet te veel zeg over hoe het ,,zou moeten" – ik ben immers ook maar een amateur – kan ik toch in ieder geval vertellen hoe het in het verleden ,,gedaan is".

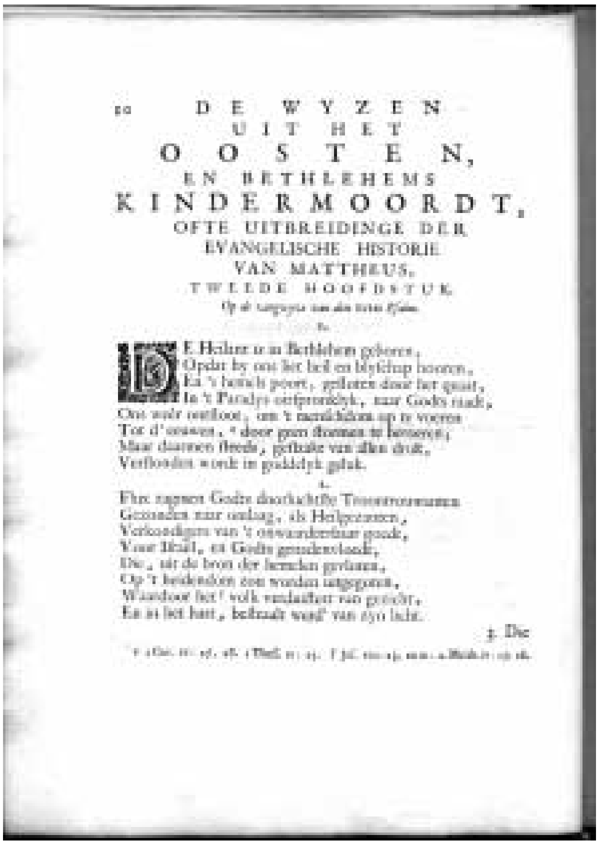
De achttiende eeuw

Figuur 1 is een scan uit het oudste boek dat ik heb gevonden dat gezet is in nederland (friesland). Het geschrift is uitgegeven door François Halma, een in die tijd er belangrijke uitgever en vertaler (voor zover ik kan nagaan is hij ook de auteur). Het boek is ingebonden in een omslag van perkament en merkwaardig goed bewaard gebleven.

Aan het gebruikte font vallen een aantal meteen dingen op: er is geen ‚ij' ligatuur. Later werd het gebruikelijk om in deze gevallen gebruik te maken van een ‚i' en een ‚j' (zoals we zullen zien in vrijwel alle volgende figuren), maar hier is dat nog de ‚y'. Aan de andere kant zijn er wel ligaturen met de s: ‚st' en ‚sl'. Het is ook duidelijk waarom er een noodzaak was voor zo'n ligatuur: de ‚s' lijkt nog het meest op een ‚f' en heeft minstens evenveel overhang.

Rechtsonderaan op de pagina staat alvast de eerste lettergreep van de volgende bladzijde. Heel lang is dit een

1. De meeste van deze boeken zijn in tijdelijk bruikleen afgestaan door het Sliedrechts Museum in Sliedrecht. Waarvoor mijn dank.


DE WYZEN
UIT HET
OOSTEN,
EN BETHLEHEMS
KINDERMOORDT,
OFTE UITBREIDINGE DER
EVANGELISCHE HISTORIE
VAN MATTHEUS,
TWEEDE HOOFDSTUK.

Figuur 1. Heilige Feestgezangen en de Zegepraal des Geloofs / Fr. Halma, Franeker – 1708

standaard hulpmiddel geweest, met name voor teksten die gedeclameerd of gezongen moeten worden (het zorgt voor een klein beetje overlap, waardoor net genoeg tijd beschikbaar is om de pagina om te draaien zonder te hoeven stoppen). Hier is de traditionele zetter duidelijk in het voordeel: als je de letters voor je ziet terwijl je pagina voor pagina opmaakt is dit makkelijk te doen, maar TEX zou er een zware dobber aan zou hebben.[2]

De uitvoering van het zetwerk laat nog een andere traditie zien die tegenwoordig in onbruik is geraakt: vóór leestekens wordt consequent een halve spatie toegevoegd. Merkwaardig hier is dat kennelijk een markering voor een voetnoot ook telt als leesteken, waardoor deze (voor ons gevoel) nogal los staan van de tekst waarop ze betrekking hebben.

Het boek is verder niet bepaald een kunstwerk: de letters zweven boven en onder de regel, er ontbreken een paar spaties en de opbouw van de titel is duidelijk uit de losse hand gedaan. Tenslotte is het merkwaardig is dat de titel gewoon door de kopregel heen loopt (er zelfs een stukje bovenuit steekt).


PREDICATIE.

Figuur 2. Keurstoffen, of verzameling van vijftig uitmuntende predicatien . . . / B. Smytegelt – H. Höveker, Amsterdam 1764

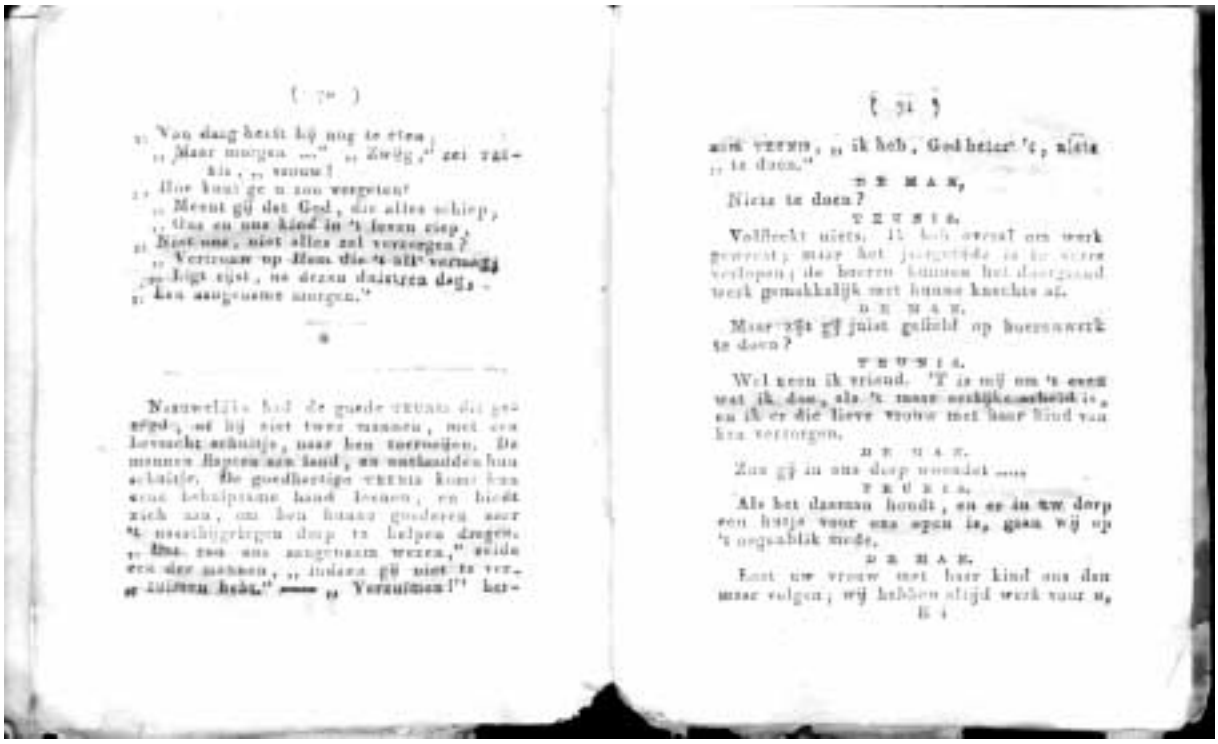
Figuur 2 is ruim vijftig jaar jonger dan het vorige boek, maar desalniettemin duidelijk ouderwetser van opzet. Het gebruik van een oudhollandse letter (waarschijnlijk het font van Fleischman voor Joh. Enschedé) voor de broodtekst met een moderne romein voor benadrukte tekst vergt een ruime periode van gewenning voor onze ogen (de romein heeft trouwens een mooie, wat eigenaardige ‚g').

Ook hier staat weer een leeshulpje rechtsonder, maar in dit boek is al te zien dat dit meer traditie was dan dat het een serieuze functie had; daarvoor is de tekst in vrijwel alle gevallen te kort gehouden.

In tegenstelling tot het vorige voorbeeld is hier sprake van (voorgedragen) proza, en daardoor zien we een paar nieuwe zaken in de regelopbouw: het gebruik van een schuin streepje op plaatsen waar een verandering in de toonhoogte van de zin plaatsvindt (een in onbruik geworden leesteken) en afgebroken woorden. De vorm van het afbreekteken zal pas veranderen in het nu gebruikelijke platte streepje met de opkomst van de machines zoals de linotype. Ik weet niet of dit een gevolg was van de beperkingen van deze machines of gewoon een verandering in opvatting, maar ik vind het ouderwetse schuin omhoog gaande (dubbele) streepje wel veel mooier.

---

2. Het kan wel, maar alleen met een hele hoop moeite en een ingewikkelde interactie met de `\output` routine.



Figuur 3. Verlustigende en Leerzame Prent-galerij voor de Jeugd in Poëzij en Proza. H. Gartman / Uitgave: wed. A. Bakker en zoon, Amsterdam 1811.

Over naar een luchtiger onderwerp: de volgende figuur (3) heeft een aanzienlijk vrolijker onderwerp (ondanks een niet minder ernstige titel). Dit is een piepklein boekje, het meet slechts 11,5 bij 9 centimeter. De kwaliteit van het zetwerk is ronduit slecht, maar het laat wel een paar nieuwe dingen zien.

Het is lange tijd gewoonte geweest om de openingsaanhalingstekens voor een citaat te herhalen aan het begin van alle volgende geciteerde regels. Alweer zo'n geval van een traditie waar een computer het danig moeilijk mee zou hebben. Het ziet er soms ook wel heel raar uit, zoals onderaan de linker pagina, waar de aanhalingstekens voorkomen aan het begin van het tweede deel van een afgebroken woord.

Nadruk wordt in dit boekje gecreëerd door de letters te spatiëren. Niet iets dat echt ongewoon was, met name bij dit soort uitgaves die zo goedkoop mogelijk moesten zijn: het gebruik van spaties binnen hetzelfde font was aanmerkelijk minder werk dan het gebruik van een apart font. Hoe voddig het zetwerk er verder ook uit ziet, dit font had wél degelijk een „ij" ligatuur en klein-kapitalen.

Aardig om nog even te vermelden is de „E 4" op de rechter bladzijde. Dit is een merkteken voor het vouwen en snijden (zodat de katernen niet in de war raken). Ook hier is sprake van de dwang van de economie: bij een duurder boek zou deze informatie in de snijrand van het papier staan, en als snijafval weggegooid worden. Maar papier kost ook geld, dus . . .

De „Gedenkschriften" (figuur 4) sluiten de rustige periode van de nederlandse typografie af. Aardig om op te merken is dat dit boek gezet is in een romein die sprekend lijkt op de versie uit het preken-boek. De mini-inhoudsopgave aan het begin van een hoofdstuk is ook al zoiets dat tegenwoordig vaak vergeten wordt of volstrekt anders wordt aangepakt. Deze layoutvorm – in wat feitelijk gewoon een losse alinea is – is heel wat minder opvallend (lees: irritant in het oog springend) dan de tegenwoordig vaak gebruikte lijst-vormgeving.

KORTE INHOUD

VAN HET

Eerste Deel.

INLEIDING.

1ste HOOFDSTUK.

Figuur 4. Gedenkschriften van den Majoor W.P. d' Auzon De Boisminart - Uitg. J.G. La Lau, Leiden 1841

AAN

MIJNE DICHTVRIENDEN

VAN

VROEGERE DAGEN.

Kindertjes van mijne handen,
Kleine dichtjes! los en vrij,
Spraakjes mijner jongelings droomen!
Zoo gegroeid, zoo gegroeid,
Dwaalt dat 'k u dit liedje zij.

Figuur 5. Allerlei / E.T. van Beusekom – J.F. Schleijer 1838

De romantiek

We komen nu in een duidelijk herkenbare andere cultuurhistorische periode: die van de romantiek. Ornamenten worden veelvuldig gebruikt in alle soorten drukwerk, en zelfs ‚serieuze' geschriften ontkomen niet aan een paar sierletters hier en daar. We zien de opkomst van de ‚schaduwletter' en fonts die specifiek zijn ontworpen voor reclame-doeleinden.

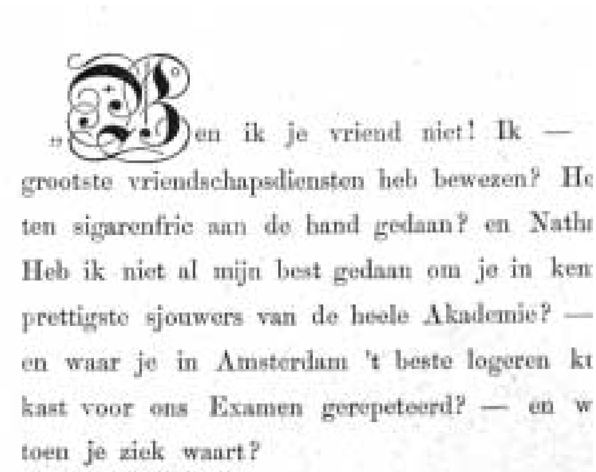
Het eerste van twee voorbeelden is figuur 5. Over de typografie heb ik niet zo veel te vertellen (vrijwel elke pagina ziet er zo uit als deze, kijk eens naar de *gaten* tussen de woorden). De titel slaat evenzeer op de vormgeving als op de inhoud!

Later (aan het einde van de 19de, begin 20ste eeuw) zou het nog iets erger worden, maar deze pagina alleen telt al negen verschillende fonts, vijf verschillende paragraafvormen en drie talen. Dan nog wat decoratieve streepjes, en het overzicht is volledig zoek. En wij maar denken dat dit soort pagina's alleen gemaakt konden worden door een knutselaar met behulp van een modern DTP pakket.

Het tweede voorbeeld is interessanter, al was het maar alleen om de zetspiegel: we zien hier (figuur 6) de linkerbladzijde van een plaatjesboek met om en om een spotprent en een toelichting en/of toepasselijk gedichtje. Het boek is een vervolg op het eerder verschenen *Zijn er zoo?*. Romantisch op en top, maar duidelijk gedaan door een vakman: de gekozen lettertypes zijn vrijwel altijd onverwacht maar nooit overdreven. Dit is voor een groot deel te danken aan


1847. Alexander V. H.

Figuur 6. Zoo Zijn Er / Alexander V. H. – J.H. Gebhard en comp^{ie}, Adam 1847


„Ben ik je vriend niet! Ik —
grootste vriendschapsdiensten heb bewezen? He
ten sigarenfric aan de hand gedaan? en Nath
Heb ik niet al mijn best gedaan om je in ken
prettigste sjouwers van de heele Akademie? —
en waar je in Amsterdam 't beste logeeren k
kast voor ons Examen gerepeteerd? — en w
toen je ziek waart?

Figuur 7. Zoo Zijn Er / Alexander V. H. – J.H. Gebhard en comp^{ie}, Adam 1847 (detail)

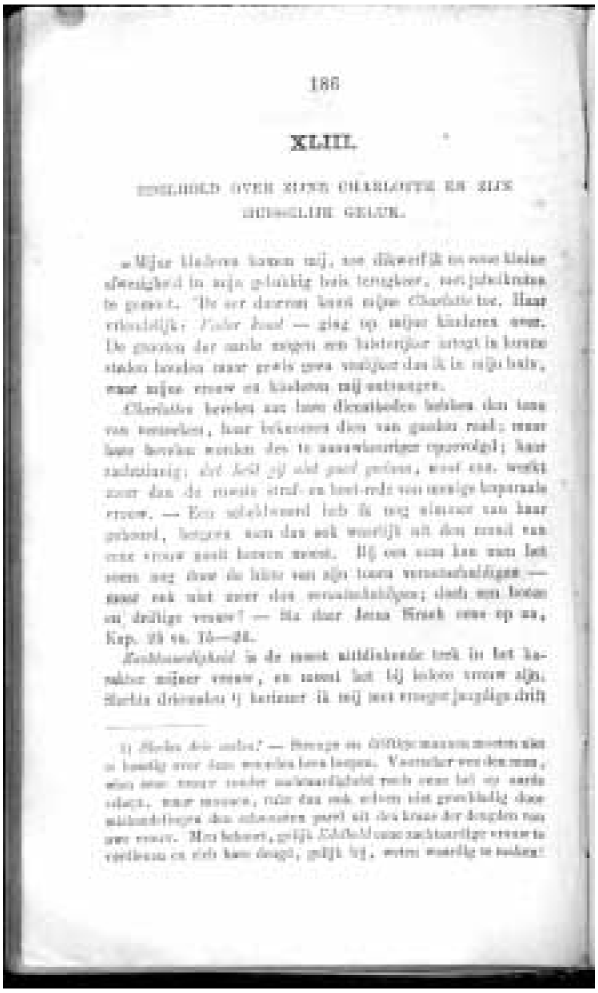
de grote marges.

Uit hetzelfde boek nog een tweede scan (figuur 7), ditmaal van één van de prachtige beginletters. Vergelijk deze beginkapitaal eens met de beginkapitaal uit de allereerste figuur. Wat opvalt is dat in deze letter het figuratieve karakter in eerste instantie wordt gehaald uit de vorm van de letter zelf, terwijl in de oudere versie nog werd vastgehouden aan een meer middeleeuwse vorm met een relatief kale letter, versierd door een miniatuur.

Traditioneel?

De volgende figuur (8) laat het begin zien van onze ‚moderne' typografische traditie. De opbouw van de pagina is vrijwel zoals we die tegenwoordig gewend zijn, met de voetnoot in een wat kleiner font, aangeduid met een nummer in superscript gevolgd door een sluithaakje. Nadruk wordt bereikt met een schuine letter, en de binnenmarge is kleiner dan de buitenmarge.

Tradities zijn er om te veranderen, en dat zien we dan ook gebeuren. Als we deze pagina vergelijken met huidige normen, valt op dat bijzinnen tegenwoordig niet meer —



186

XLIII.

Figuur 8. Pachter Martijn en zijn vader (H.G. Demme, vert uit het Hoog Duits: C.S. Adema van Sheltema) – C.L. Brinkman 1858

zo — gedaan worden, maar bij voorkeur – zo –, of ook wel (zo). De laatste vorm wordt nog steeds gezien als wat informeel, maar wint de laatste paar jaar terrein.

De endash als separator tussen twee nummers bestond in 1858 nog niet eens, dit is een ‚uitvinding' die gedaan werd aan het eind van de negentiende eeuw.

Voor deze periode zijn verder het nogal magere font (Computer Modern is afgeleid van een font uit deze periode) kenmerkend, de in verhouding lichtgrijze inkt[3], en de wat merkwaardige aanhalingstekens.

DE GESCHIEDENIS DES OUDEN TESTAMENTS.

Figuur 9. De bijbelsche geschiedenis, aan kinderen verhaald – Donner, Leiden 1883

Figuur 9 is weer iets heel anders. Kinderbijbels zijn intrigerende dingen. Ik vraag me al tijden af of er eigenlijk wel kinderen zijn die die dingen kunnen lezen zonder hulp van een volwassene.

Wél wordt er altijd erg veel tijd besteed aan het drukken van bijbels, of die nu voor kinderen zijn bedoeld of voor volwassenen. Maar met al die aandacht kan kennelijk niet verholpen worden dat het zichtbaar is dat de dubbele aanhalingstekens uit dit font geen ligatuur zijn, maar samengesteld uit twee tekens (dit terwijl er wel een ‚ij' is, wat toch een typisch nederlands font impliceert).

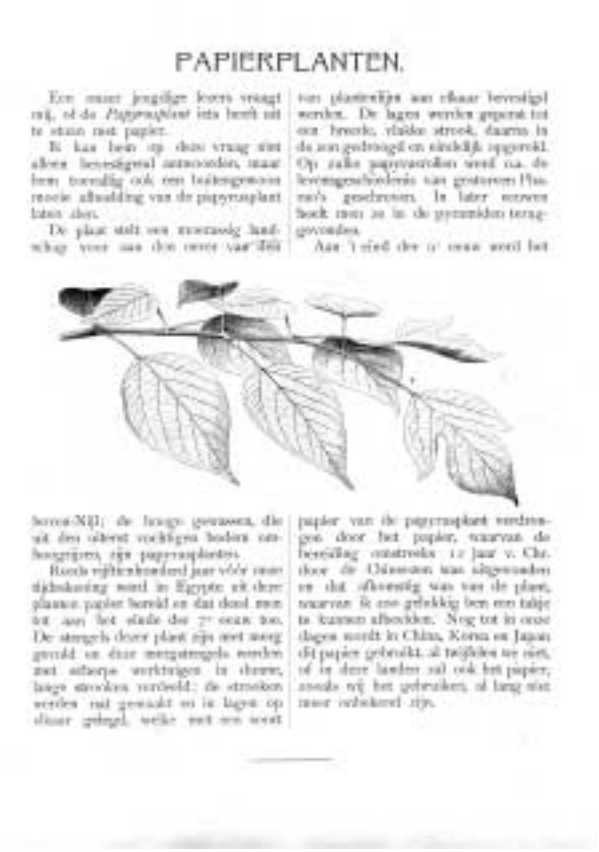
Aan de vierde en vijfde regel van onderen is nog iets merkwaardigs te zien: de ‚g'-s zijn hier kennelijk bijgevijld om een conflict met de regel daaronder te vermijden.

3. Dat zal waarschijnlijk hier slecht overkomen, ik moest nogal wat rommelen in Photoshop om een redelijke EPS te krijgen voor de MAPS.


PREDIKBEURTEN TE SLIEDRECHT.

Predikbeurten in den omtrek.

Figuur 10. Zondagsbode voor Sliedrecht en omstreken – A. van Wijngaarden, Sliedrecht 1890


PAPIERPLANTEN.

Figuur 11. Voor 't Jonge Volkje (red S. Abramsz) – P. van Belkum Az, Zutphen. volume 68 (1912?)

Tijdschrift–typografie

Aan het eind van de negentiende eeuw en in het begin van de twintigste eeuw vindt een indrukwekkende economische opleving plaats.

De ‚herontdekking' van reclame als medium (en het daarmee gepaard gaande reclamedrukwerk), gecombineerd met een toenemende mobiliteit (en de daarmee gepaard gaande behoefte aan goedkope reislektuur), en de uitvinding van zetmachines (resulterend in aanzienlijk goedkopere prijzen voor zetwerk) zorgen voor een enorme groei van het aantal drukkerijen in nederland, en het resultaat is een ware stortvloed aan gedrukte ‚lektuur'.

In deze periode zien we met name de opkomst van lokale nieuwsblaadjes en lokale kranten, en het uitontwikkelen van het tijdschrift-formaat (met een wat mooiere term: maandelijkse periodieken).

De eersten om gebruik te maken van de nieuwe mogelijkheden waren uiteraard de kerken en de – vaak door kerken gesubsidieerde – verenigingen. Overal in nederland zien we de ,,Zondagsboden" als onkruid uit de grond schieten. Dat de typografie duidelijk ondergeschikt is aan de behoefte om op te vallen, zien we in figuur 10 (weet u nog dat ik een paar pagina's geleden zei dat het nog erger zou worden?).

De pagina ziet er indrukwekkend uit, maar de kosten vielen wel mee: het geheel werd op stand bewaard tot volgende week. Dat kon natuurlijk makkelijk, want alle kerken hadden hun vaste tijden, en als er een keer geen dominee voor handen was werd er simpelweg ,,Geen Dienst" tussen gezet. Na wat bladeren blijkt dat deze pagina minstens vier jaar zó onveranderd gebruikt is geweest!

De volgende scan (figuur 11) is een mooi voorbeeld van een wat meer ingetogen stijl. In mijn ogen prachtig zetwerk, samen met een keurig verzorgd plaatje. Weinig afbrekingen, en de superscripted letters staan niet al te hoog (een euvel waar `\textsuperscript` nogal aan lijdt, vergelijk 7^{e} en 7^{e}).

Kennelijk had de heer Abramsz redelijk wat geld tot zijn beschikking, want niet alleen is het zetwerk keurig, maar de kwaliteit van het papier is ook hoog, en de fontgrootte is duidelijk gekozen om zo makkelijk leesbaar mogelijk te

DE HUISHOUDGIDS

Verschijnt elken Zaterdag onder redactie van

MEJUFFROUW N. CARIOT

Bureau: BURGSTRAAT 65 te UTRECHT.

RECEPTEN.

DEKKER's IJshandel

Figuur 12. De Huishoudgids (red. mej. N. Cariot) – 1911

FEUILLETON.

Een drama op zee.

IN een artikel in de Figaro, waarin Sarah Bernhardt een tragisch voorval vertelt, dat zij zelf te Belle-Isle-en-Mer beleefde, toont de groote actrice, dat zij ook zeer goed de pen weet te hanteeren. Het is trouwens bekend, dat Sarah een tamelijk universeel talent heeft, dat zij ook schildert, beeldhoudt, boetseert.

Sarah Bernhardt heeft op een der eilanden in het Zuiden van Bretagne een dier tragische tooneelen bijgewoond, die daar niet zeldzaam zijn, heel weinig zeldzaam zelfs, maar dat toch de licht gevoelige actrice onbeschrijfelijk aandeed. Minder om het drama zelf, dan om het stukje van de beroemde tooneelspeelster geven wij hier een uittreksel van „Un Drame en Mer".

Figuur 13. De Huishoudgids (red. mej. N. Cariot) – 1911

zijn (terwijl dat nog heel wat werk inhoudt binnen een zo kleine kolombreedte, om het nog niet te hebben over de hoeveelheid benodigd papier)

Een ander tijdschrift uit deze tijd is in schrijnend contrast: De huishoudgids is weliswaar een weekblad, maar dat is nog geen excuus voor de duidelijk mindere kwaliteit. De twee kolommen lijnen niet uit, het drukwerk staat vol van de halve letters en zelfs de vaste onderdelen zoals de aanhef en het kopje „recepten" zijn slordig uitgevoerd.

Maar eerlijk is eerlijk: er zitten ook prachtige stukjes in. Zoals de versierde letters die gebruikt worden voor het begin van het wekelijks feuilleton (figuur 13).

En dat was het dan. Ik hoop dat dit artikel de moeite waard was om te lezen. Ikzelf heb in ieder geval erg veel plezier gehad met het lezen van de gescande artikelen. Heeft u er wat van geleerd, dan is dat waarschijnlijk toevallig. Maar in ieder geval hoop ik dat ik heb weten uit te leggen dat nederland in het verleden niet alleen veel goede typografie en zetwerk heeft opgeleverd, maar daarnaast ook een vrij grote collectie rommel en prutswerk. Laat u door geen „expert" wat anders wijsmaken!

Bijlage 8
Vlakverdeling in CONTEXT

Hans Hagen
pragma@pi.net
ntg-context@ntg.nl

abstract
This article is actually chapter 3 of the CONTEXT reference manual, typeset in the MAPS layout. Attention will be paid to defining layout specific areas, rearranging pages, locating logos, typesetting on a grid and adding cutmarks.

Keywords
CONTEXT, layout, grids, two-up, logos, arranging pages

1 Inleiding

Bij het bewerken van een tekst houdt TEX rekening met (onder andere) de actuele `\hsize` (breedte) en `\vsize` (hoogte). Zodra de ingestelde `\vsize` wordt overschreden, roept TEX de zogenaamde output–routine aan. Deze handelt vervolgens het gezette deel, meestal een bladzijde, af. Dit afhandelen bestaat uit onder meer het plaatsen van hoofd- en voetregels, het zetten van het paginanummer, het aanbrengen van achtergronden en navigatiemiddelen en het plaatsen van voetnoten, verplaatste tabellen en figuren. Er zijn dan ook meer maten in het geding dan alleen de hoogte en de breedte van de te zetten tekst.

2 Papierformaat

Met het commando `\stelpapierformaatin` worden de afmetingen van het papier ingesteld. We maken daarbij onderscheid tussen het formaat waarop we zetten en het formaat waarop we afdrukken (printen).

```
\stelpapierformaatin[..,.1.,..][..,.2.,..]
.1.      A3 A4 A5 A6 CD naam liggend gespiegeld geroteerd 90 180 270
.2.      A3 A4 A5 A6 naam liggend gespiegeld geroteerd negatief 90 180 270
```

De afmetingen van de DIN–formaten zijn in tabel 1 weergegeven.

formaat	afmetingen in mm	formaat	afmetingen in mm
A0	841 × 1189	A5	148 × 210
A1	594 × 841	A6	105 × 148
A2	420 × 594	A7	74 × 105
A3	297 × 420	A8	52 × 74
A4	210 × 297	A9	37 × 52

Tabel 1 Standaard papierformaten.

Naast deze afmetingen zijn ook B0–B9 en C0–C9 beschikbaar, en verder: `letter`, `legal`, `folio` en `executive`, `envelop 9–14`, `monarch`, `check`, `DL` en `CD`.

Men kan een nieuw formaat definiëren met:

```
\definieerpapierformaat[...][..,..=..,..]
...         naam
breedte     getal
hoogte      getal
schaal      maat
```

Zo is bijvoorbeeld CD gedefinieerd als:

```
\definieerpapierformaat[CD][breedte=12cm,hoogte=12cm]
```

We kunnen nu bijvoorbeeld zeggen:

```
\stelpapierformaatin[CD][A4]
```

Wat zoveel betekent als: gebruik voor het zetwerk CD en voor het printen A4. We komen later op die tweede instelling terug.

Naast de op papier gerichte formaten zijn er enkele meer op het beeldscherm afgestelde varianten: S3–S6. Deze definiëren schermen met breedtes oplopend van 300 tot 600 pt en een hoogte van 3/4 maal de breedte. Deze S–formaten sluiten dus aan bij de aspect ratio van het beeldscherm.

Bij het instellen van een papierformaat wordt de actuele layout automatisch meegeschaald. Zo heeft men in ieder geval een vertrekpunt.

3 Bladindeling

We maken om praktische redenen onderscheid tussen wat we zullen noemen de zetspiegel en de marges. De zetspiegel is dat deel van de bladzijde waar de lopende tekst wordt geplaatst. In dit gedeelte worden standaard alle tekstelementen geplaatst (zie figuur 1).

Het hoofd bevindt zich *tussen* het kopwit en de zetspiegel. In het hoofd en de voet worden lopende titels en paginanummers geplaatst. Links en/of rechts van de tekst kunnen we structureel of incidenteel bepaalde informatie kwijt. Bijvoorbeeld nummers van hoofdstukken en paragrafen of trefwoorden. De marges maken deel uit van het rug- en snijwit.

links **rechts**

De breedte van de marges heeft *geen* invloed op de plaats van de zetspiegel, de hoogte van het hoofd en de voet bepalen *wel* mede de hoogte van het tekstdeel.

De zetspiegel en marges kunnen worden ingesteld met het commando \stellayoutin. Onder de zetspiegel verstaan we (hier) het gedeelte van de bladzijde dat overblijft als we de witte randen weglaten. In figuur 1 is dat het hoofd, het middendeel en de voet (het grijze gedeelte).

Het instellen van de marge heeft bij een papieren document geen gevolgen voor de zetspiegel. Deze instelling speelt alleen een rol bij het plaatsen (en afbreken) van tekst in de linker- of rechtermarge.

In papieren documenten kunnen we meestal volstaan met hoofd- en voetregels. In electronische documenten daarentegen hebben we ook ruimte nodig voor sturende elementen. Omdat bij electronische documenten het gebruik van achtergronden wat meer voor de hand ligt — door middel van een achtergrond kunnen we de verschillende delen van het scherm benadrukken — zijn ook de afstanden tussen bijvoorbeeld tekst en hoofd en tekst en voet instelbaar.

Het is mogelijk de zetspiegel zichtbaar te maken in de tekst. Hiervoor kunnen de volgende commando's worden gebruikt:

```
\toonkader[...]
...      tekst marge rand
```

```
\stellayoutin[..,..=..,..]
breedte                 maat passend midden
hoogte                  maat passend midden
rugwit                  maat
kopwit                  maat
marge                   maat
linkermarge             maat
rechtermarge            maat
hoofd                   maat
voet                    maat
boven                   maat
onder                   maat
linkerrand              maat
rechterrand             maat
hoofdafstand            maat
voetafstand             maat
bovenafstand            maat
onderafstand            maat
linkermargeafstand      maat
rechtermargeafstand     maat
linkerrandafstand       maat
rechterrandafstand      maat
rugoffset               maat
kopoffset               maat
letter                  normaal vet schuin vetschuin type kap klein... commando
markering               aan uit kleur
plaats                  links midden rechts onder boven enkelzijdig dubbelzijdig
schaal                  maat
nx                      getal
ny                      getal
regels                  getal
grid                    ja nee
```

De instellingen kunnen zichtbaar worden gemaakt met:

```
\tooninstellingen
```

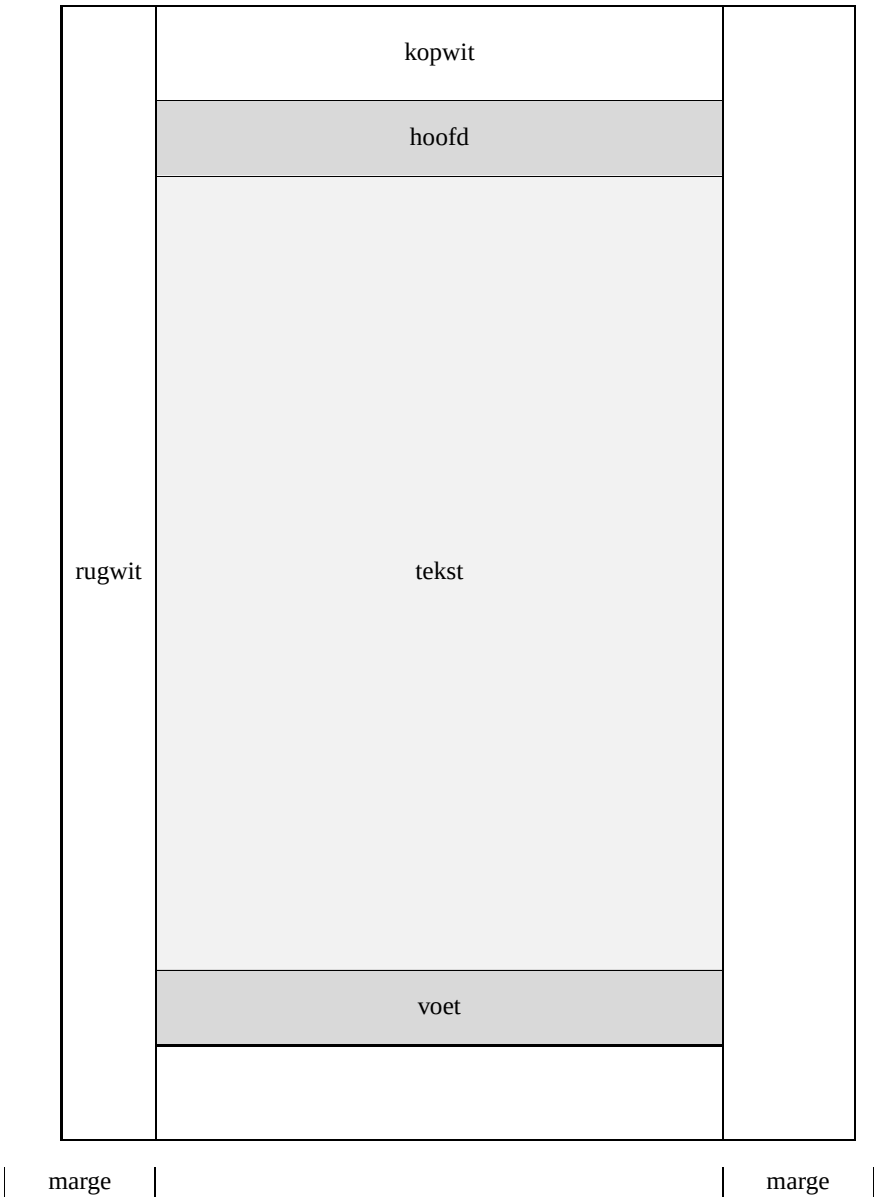
Een combinatie van beiden roepen we op met:

```
\toonlayout
```

De breedte van de tekst is meestal beschikbaar in `\hsize` en de hoogte in `\vsize`. Willen we echter op safe spelen dan kan men beter gebruik maken van de `\dimen`–registers `\tekstbreedte` en `\teksthoogte`, `\zetbreedte` en `\zethoogte`. Waar de zetbreedte constant is, geeft de tekstbreedte de breedte van de kolom tekst weer. Bij het zetten in twee kolommen is de tekstbreedte bijvoorbeeld iets minder dan de helft van de zetbreedte. De teksthoogte is dat wat overblijft als we de zethoogte verminderen met de hoogte van het hoofd en de voet.

Ook de andere afstanden en maten zijn beschikbaar, zoals `\linkermargebreedte` en `\voethoogte`, maar pas op: deze waarden kan men alleen gebruiken, niet instellen.

In principe wordt een tekst volledig automatisch opgemaakt. Het kan echter voorkomen dat het al dan niet verplaatsen van een regel aanzienlijk fraaier zetwerk oplevert. In dat geval kan men ter plaatse de hoogte van de zetspiegel wat aanpassen met:



Figuur 1 De A4 zetspiegel en marges (hoogte = hoofd + tekst + voet).

variabele	betekenis
`\zetbreedte`	breedte van de tekst
`\zethoogte`	hoogte van de tekst
`\tekstbreedte`	breedte van een kolom tekst
`\teksthoogte`	hoogte van de tekst − hoofd − voet

Tabel 2 Enkele `\dimen`–variabelen.

```
\paslayoutaan[..,...,..][..,..=..,..]
...       getal
hoogte    maat max
regels    getal
```

Dergelijke commando's kunnen echter ook negatieve gevolgen hebben, bijvoorbeeld wanneer we de tekst hebben gewijzigd en de aanpassing eigenlijk niet meer nodig is. Het is daarom verstandiger de aanpassingen (zichtbaar) bovenaan de tekst te definiëren. Een voorbeeld van zo'n aanpassing is:

```
\paslayoutaan[21,38][hoogte=+.5cm]
```

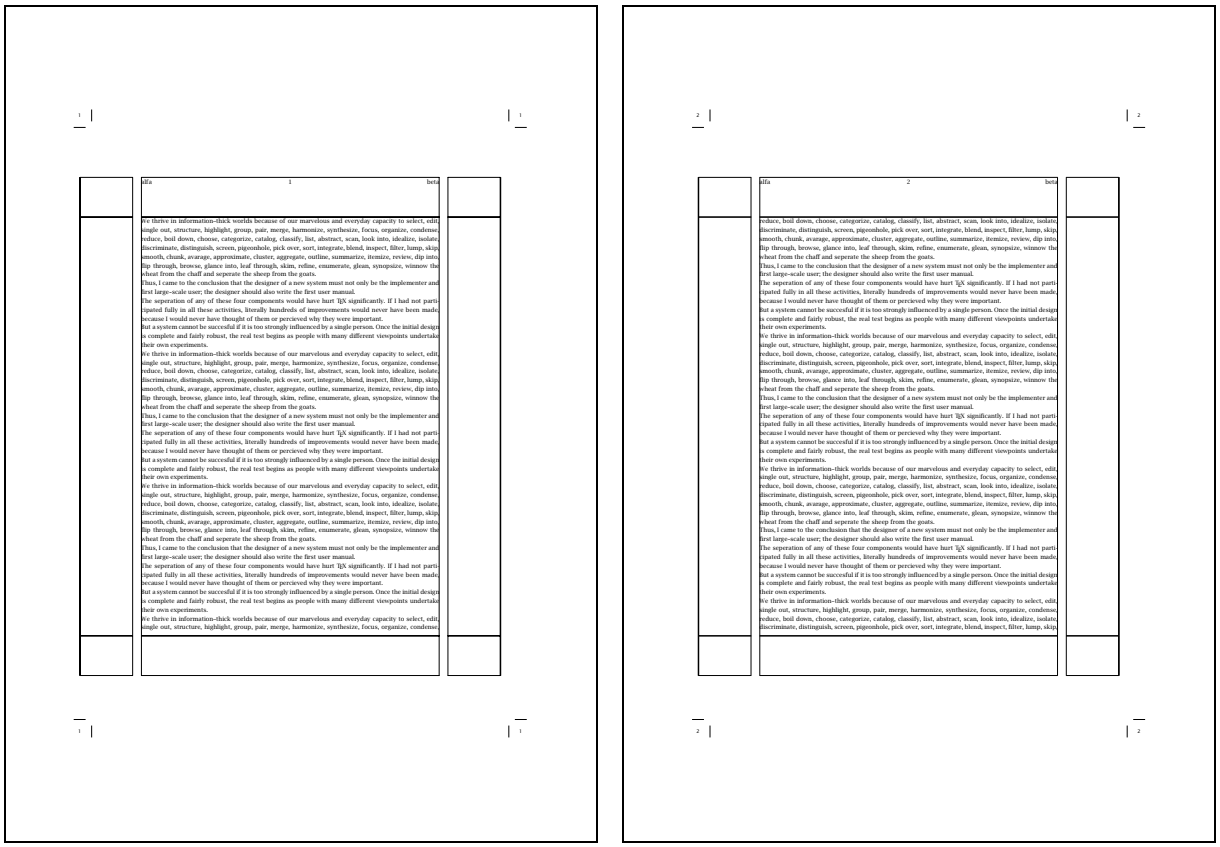
In dit geval wordt op de pagina's 21 en 38 de hoogte van de tekst tijdelijk 0.5 cm verhoogd, waarbij de voetregel netjes op de goede hoogte blijft staan. De opgegeven nummers zijn de volgnummers in de DVI–file.

Mocht onverhoopt de layout zijn verstoord, dan kan gebruik worden gemaakt van de volgende aanroep:

```
\stellayoutin[reset]
```

De oplettende lezer zal gezien hebben dat bij het instellen van de `breedte` en `hoogte` het trefwoord `passend` kan worden gebruikt. In dat geval worden de breedte en hoogte automatisch berekend, iets dat vooral handig is bij schermlayouts, waar men vaak symmetrie nastreeft.

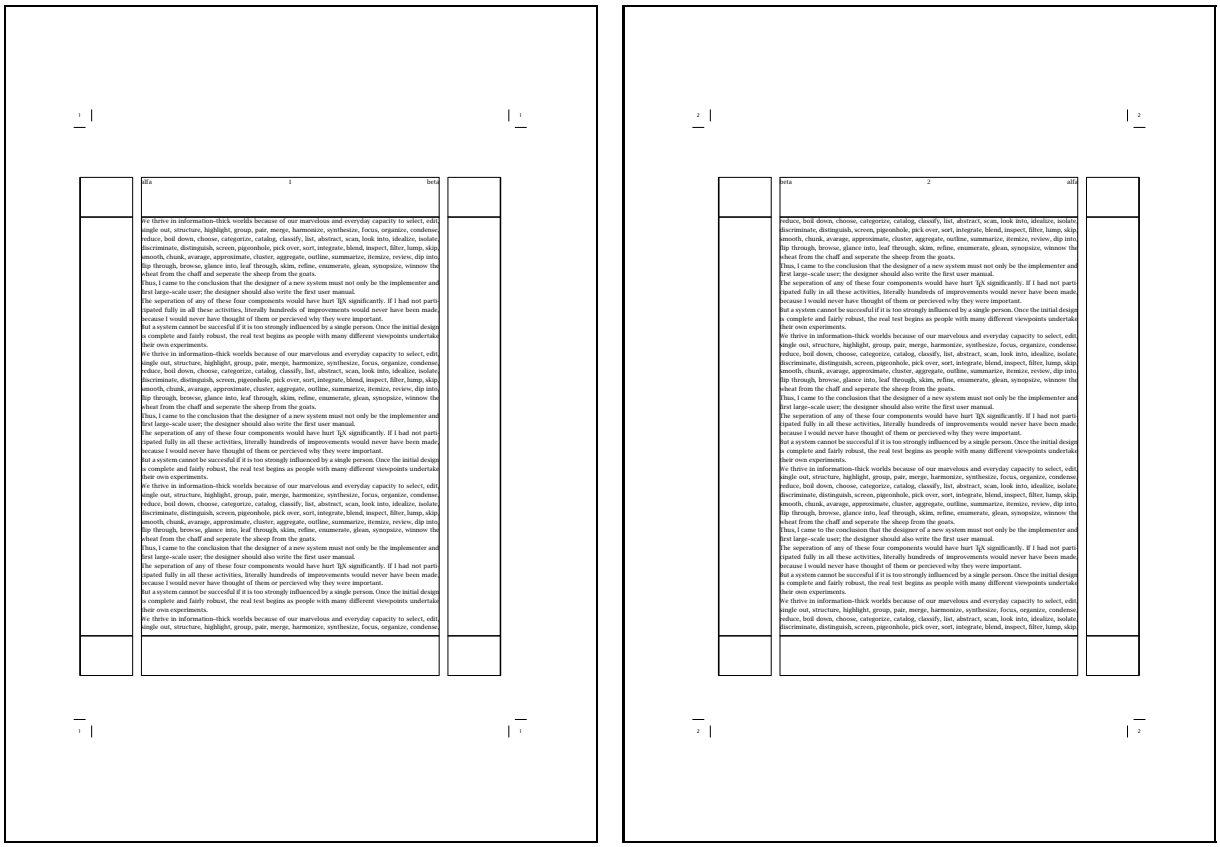
Op de volgende bladzijden tonen we enkele A5 bladspiegels gecentreerd op A4. De standaard instellingen (maten) leveren een soort compromis die goed bruikbaar is voor verslagen en notities. De instellingen passen zich automatisch aan het papierformaat aan. Let op het gebruik van het trefwoord `midden` bij het instellen van de breedte en hoogte.



rechts links

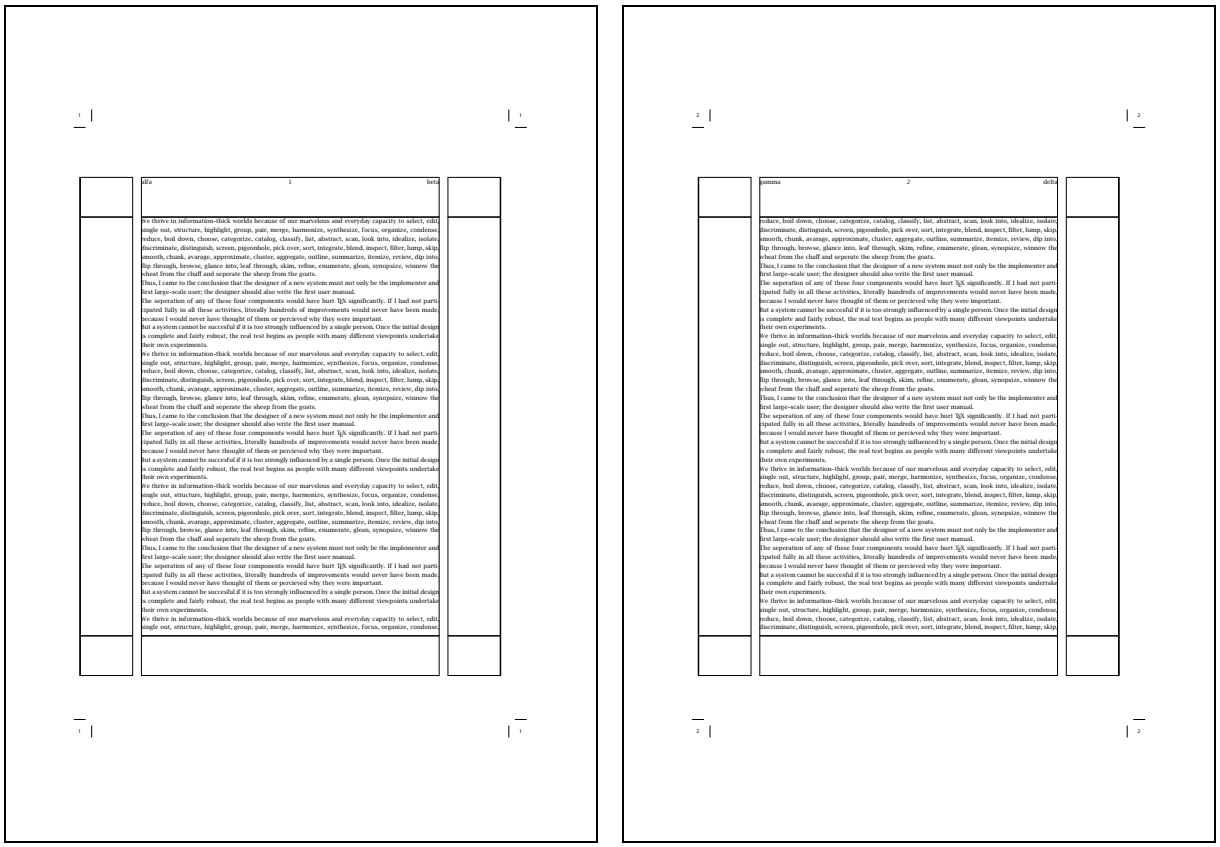
Figuur 2 De standaard verhoudingen (enkelzijdig).

```
\stelpapierformaatin [A5][A4]
\stellayoutin        [plaats=midden,markering=aan]
\stelnummeringin     [variant=enkelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



Figuur 3 De standaard verhoudingen (dubbelzijdig).

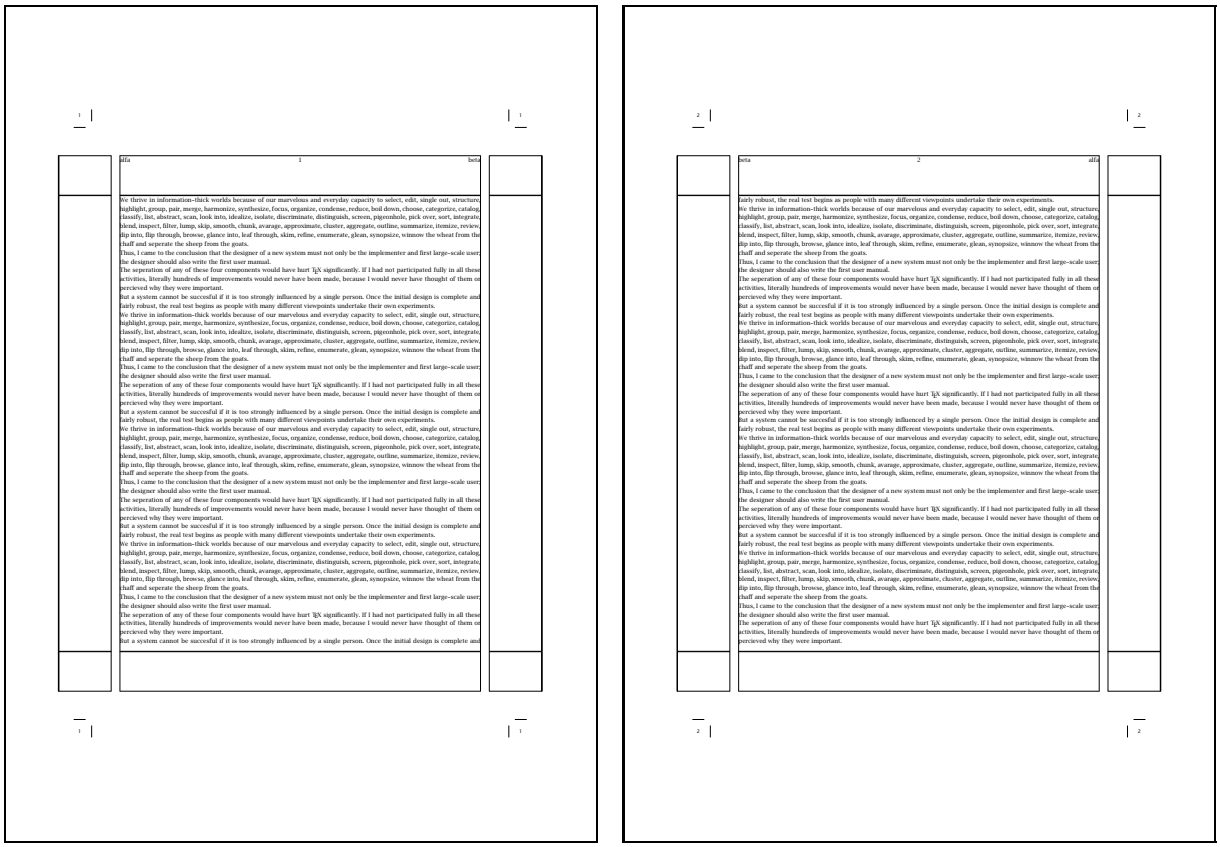
```
\stelpapierformaatin [A5][A4]
\stellayoutin        [plaats=midden,markering=aan]
\stelnummeringin     [variant=dubbelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



rechts links

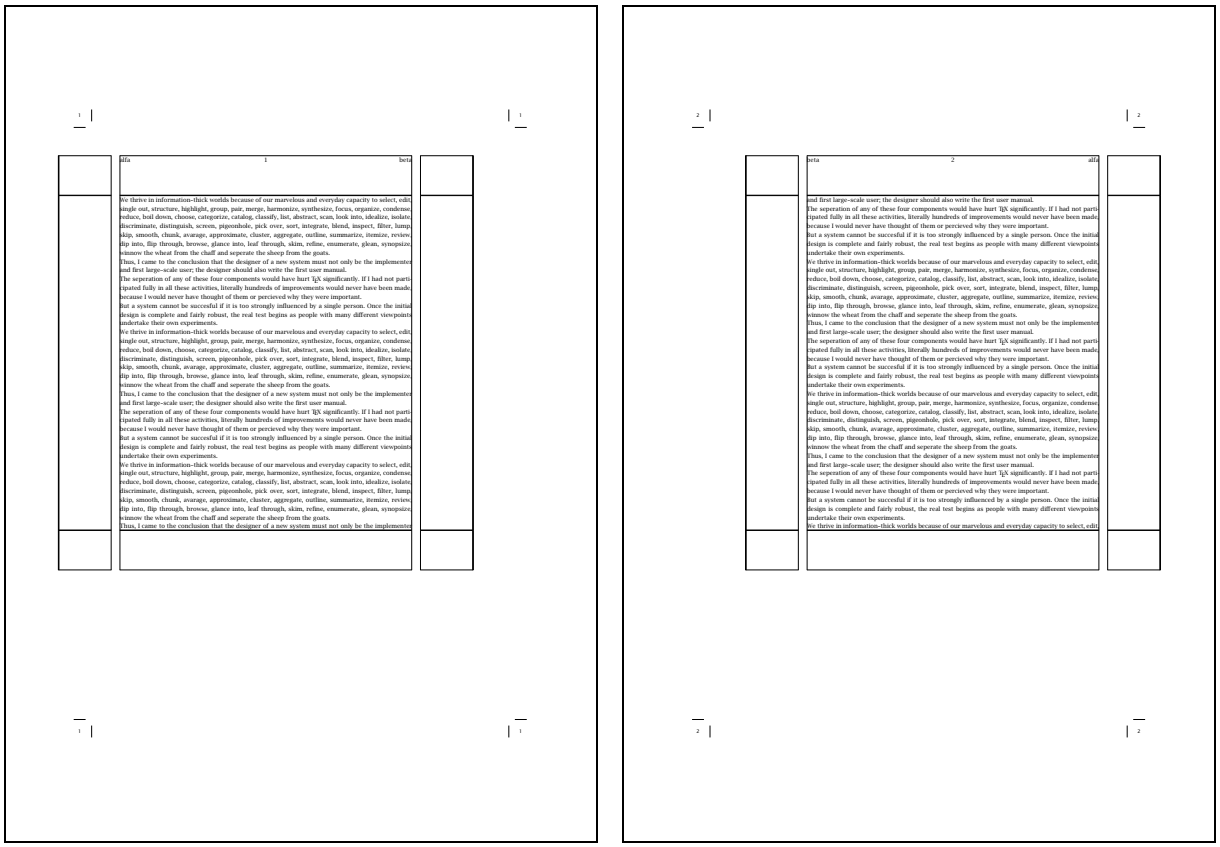
Figuur 4 De standaard verhoudingen (enkel-/dubbelzijdig).

```
\stelpapierformaatin [A5][A4]
\stellayoutin        [plaats=midden,markering=aan]
\stelnummeringin     [variant={enkelzijdig,dubbelzijdig}]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta][gamma][delta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



Figuur 5 Een automatisch gecentreerde bladspiegel.

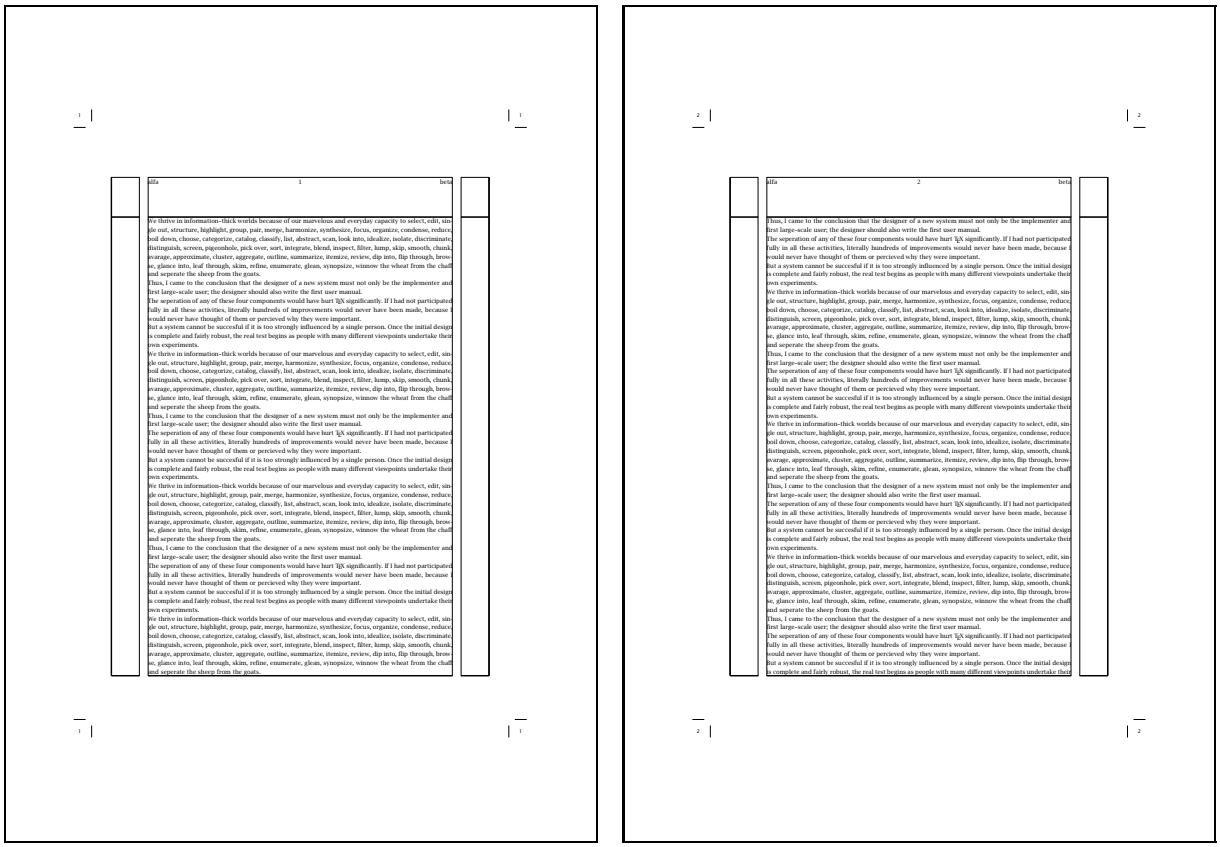
```
\stelpapierformaatin [A5][A4]
\stellayoutin        [rugwit=1cm,breedte=midden,
                      kopwit=1cm,hoogte=midden,
                      plaats=midden,markering=aan]
\stelnummeringin     [variant=dubbelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



rechts links

Figuur 6 Een a-symetrische bladspiegel.

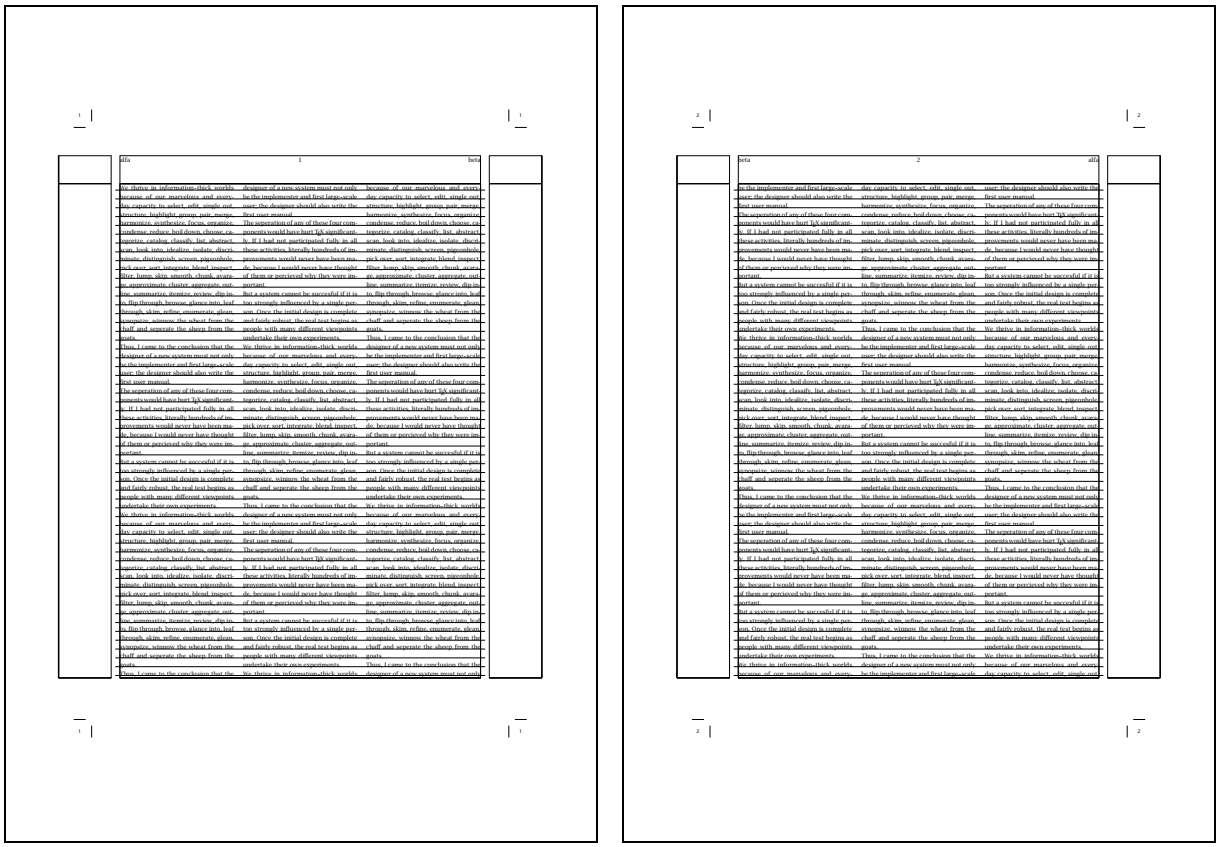
```
\stelpapierformaatin [A5][A4]
\stellayoutin        [rugwit=1cm,breedte=.7\papierbreedte,
                      kopwit=1cm,hoogte=.7\papierhoogte,
                      plaats=midden,markering=aan]
\stelnummeringin     [variant=dubbelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



rechts links

Figuur 7 Een bladspiegel zonder voet.

```
\stelpapierformaatin [A5][A4]
\stellayoutin        [rugwit=2cm,breedte=midden,
                      voet=0cm,marge=1cm,
                      plaats=midden,markering=aan]
\stelnummeringin     [variant=enkelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\toonkader

\starttekst
  \dorecurse{10}{\input tufte \par \input knuth \par}
\stoptekst
```



rechts links

Figuur 8 Een bladspiegel gebaseerd op een grid.

```
\stelpapierformaatin [A5][A4]
\stellayoutin        [plaats=midden,markering=aan]
\stelnummeringin     [variant=dubbelzijdig]
\stelkorpsin         [lbr,6pt]
\stelhoofdtekstenin  [alfa][beta]

\stellayoutin        [kopwit=1cm,regels=56,hoofd=1cm,voet=0cm,
                      rugwit=1cm,breedte=midden,grid=ja]

\toonkader \toongrid

\starttekst
  \startkolommen[n=3]
  \dorecurse{10}{\input tufte \par \input knuth \par}
  \stopkolommen
\stoptekst
```

4 Grids

Er zijn verschillende manieren om een bladzijde uit te lijnen, bijvoorbeeld:

alfa	alfa	alfa	alfa
beta	beta	beta	beta
gamma	gamma	gamma	gamma

De eerste varianten zijn in de regel ongewenst en kunnen worden bereikt door wat rek in de interlinie in te bouwen. De laatste variant heeft echter als nadeel dat bladzijden een verschillende lengte kunnen hebben. Om die reden voorzien we meestal het tussenwit van wat rek.[1]

alfa	alfa	alfa	alfa
beta	beta	beta	beta
gamma	gamma	gamma	gamma
delta	delta		

Een nadeel van deze benadering is dat de naast elkaar getoonde bladzijden of kolommen op een bladzijde zelden uitlijnen, wat vooral bij een wat grotere letter storend kan zijn. Bovendien kan bij dun papier een storend doorschijnend effect optreden.[2]

In dergelijke situaties gaat de voorkeur uit naar het zetten op een zogenaamd grid. Hoewel de middelen in TEX beperkt zijn, ondersteunt CONTEXT in beperkte mate het zetten op een grid.[3]

2 O la la, nog een voetnoot!
3 Ziezo, een laatste voetnoot!

Bij het zetten op een grid worden koppen, figuren formules en natuurlijk de lopende tekst op een vaste regelafstand gezet. Mocht onverhoopt een typografische component verstorend werken, dan kan men zo'n component op het grid plaatsen (engels: snap) met:

```
\plaatsopgrid{\omlijnd{Snapt u hier wat van?}}
```

Dit levert:

Snapt u hier wat van?

Men kan het mechanisme beïnvloeden door een argument mee te geven:

```
\plaatsopgrid[onder]{\omlijnd{Snapt u het nu?}}
```

waarna onder de omlijnde tekst een extra lege regel wordt geplaatst. Andere opties zijn: `boven` en `beide`. In het laatste geval wordt de extra regel verdeeld:

Nu snapt u het zeker?

Beide omlijnde voorbeelden zijn geen schoolvoorbeelden van schoonheid. Dit komt om-

1 Oeps, een voetnoot!

biedt CONTEXT de volgende oplossing:

```
\startregelcorrectie
\omlijnd{Dit is iets voor gevorderden.}
\stopregelcorrectie
```

Dit commando probeert zo goed en kwaad als het kan een en ander te plaatsen en houdt daarbij zonodig rekening met het grid.

Dit is iets voor gevorderden.

Omdat de regelcorrectie al rekening houdt met het grid, moeten we een ander commando gebruiken om de omkaderde tekst op te rekken:

```
\verplaatsopgrid[beide]
\startregelcorrectie
\omlijnd{Maar het kan geen kwaad een en ander te weten.}
\stopregelcorrectie
```

Inderdaad krijgen we nu wat meer ruimte:

Maar het kan geen kwaad een en ander te weten.

Men kan voor testdoeleinden het grid zichtbaar maken met het commando `\toongrid`. Samengevat hebben we dus:

```
\plaatsopgrid[.1.]{.2.}
.1.      zie p 93: \verplaatsopgrid
```

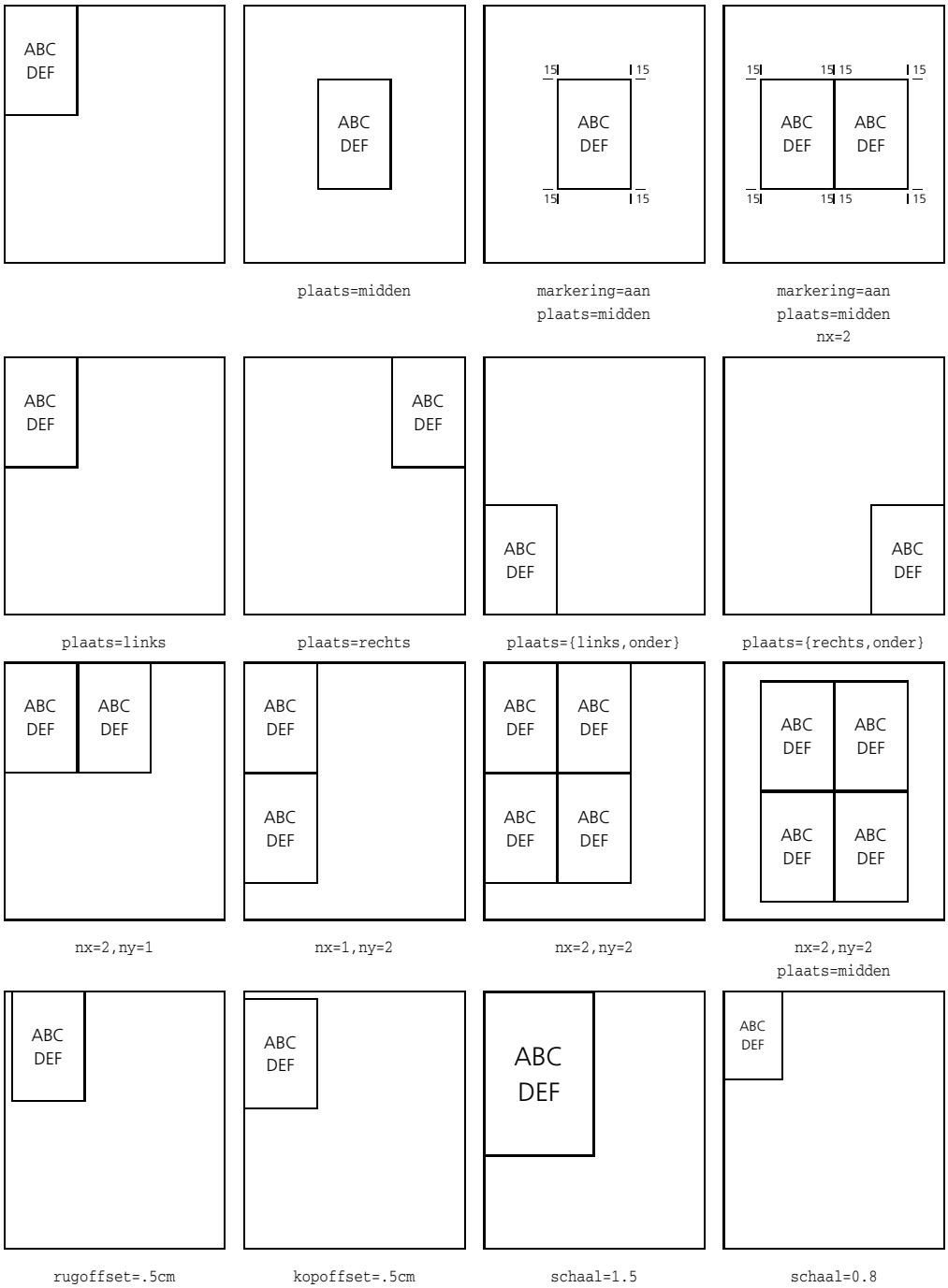
```
\verplaatsopgrid[...]
...      boven beide onder
```

```
\toongrid
```

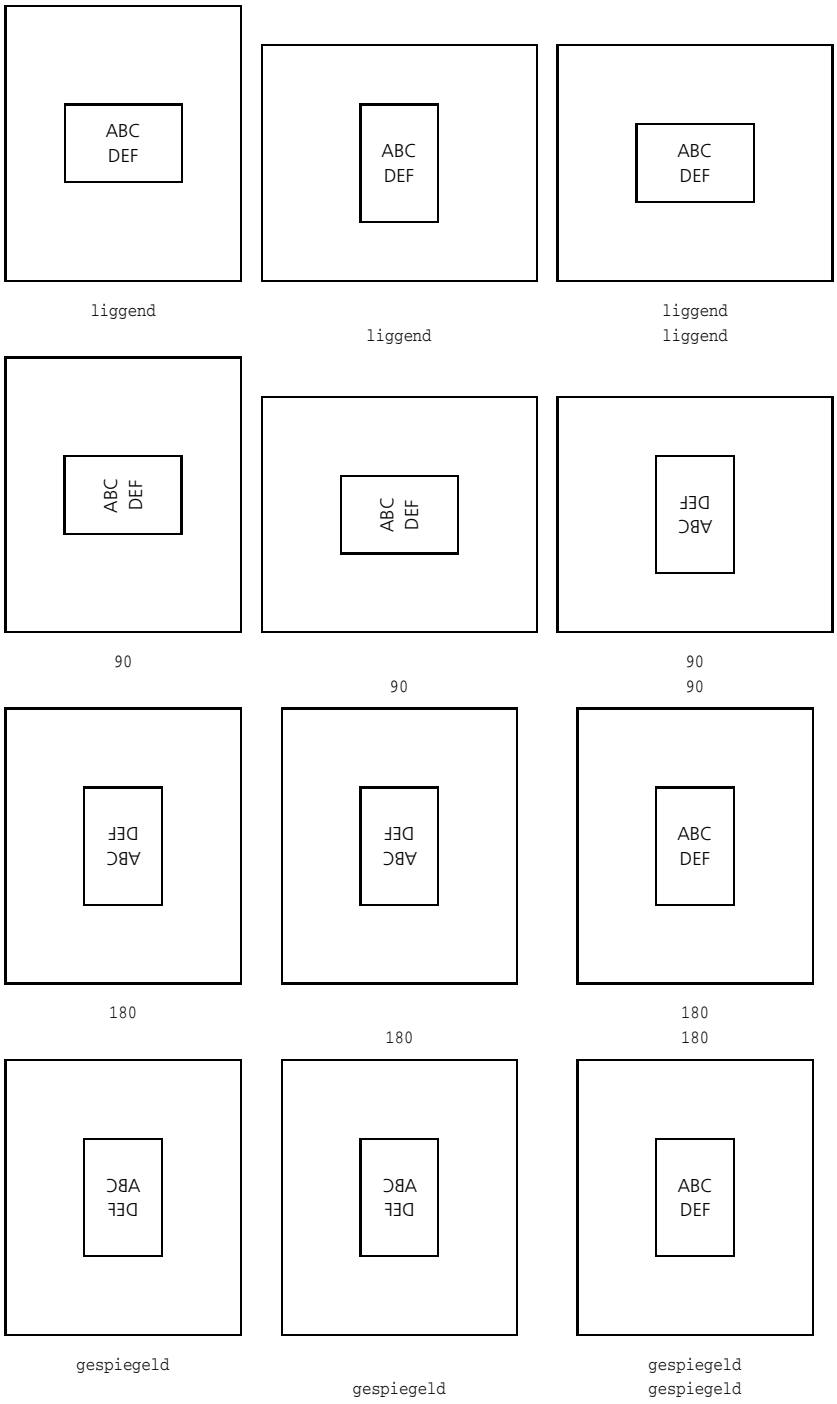
5 Printen

In een eerdere paragraaf hebben we onderscheid gemaakt tussen pagina– en papier–afmetingen. We zullen nu wat dieper ingaan op de wijze waarop deze twee kunnen worden gemanipuleerd.

In figuur 9 en 10 zien we enkele mogelijkheden om de bladspiegel te manipuleren met behulp van instellingen van de eerder besproken commando's `\stelpapierformaatin` en `\stellayoutin`. Het is dus mogelijk een pagina in een hoek of het midden van het papier te plaatsen, te copiëren en te voorzien van markeringen.



Figuur 9 Het manipuleren van de bladspiegel met `\stellayoutin`.



Figuur 10 Het manipuleren van de bladspiegel met `\stelpapierformaatin`.

Wanneer we bij het papierformaat `liggend` opgeven, dan worden breedte en hoogte omgewisseld. Er wordt dus niet geroteerd! Dit doen we met de instellingen `90`, `180` en `270`.

```
\stelpapierformaatin[A5,liggend][A4]
```

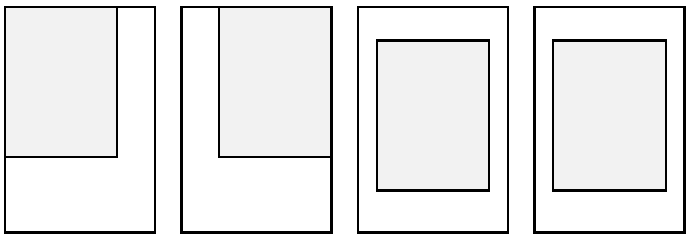
Wat de voorbeelden niet laten zien, is dat we ook kunnen corrigeren voor dubbelzijdig printen. Wanneer we zeggen:

```
\stelpapierformaatin[A5][A4]
\stellayoutin[plaats=midden,markering=aan]
```

dan komt op voor- én achterkant de tekst netjes in het midden te staan. De markering maakt het mogelijk het juiste formaat af te snijden. Als we echter maar twee keer willen snijden, dan kunnen we ook opgeven:

```
\stelpapierformaatin[A5][A4]
\stellayoutin[plaats=dubbelzijdig]
```

Dit komt overeen met `{dubbelzijdig,links}`. Bij deze instelling zorgt CONTEXT er voor dat de achterzijde automatisch wordt verschoven naar de juiste hoek. In figuur 11 laten we twee varianten zien.



Figuur 11 Het positioneren van de bladspiegel ten behoeve van het afsnijden.

Roteren, spiegelen, schalen, dupliceren en plaatsen zijn onafhankelijke operaties. Door ze slim te combineren kunnen echter vrij veel effecten worden bereikt. Roteren en spiegelen worden ingesteld tegelijk met het pagina- en papierformaat. De overige operaties worden ingesteld als layout opties.

```
\toonprint[..,.1.,..][..,.2.,..][..,..=..,..]
..=..    zie p 80: \stelpapierformaatin
..=..    zie p 80: \stelpapierformaatin
..=..    zie p 82: \stellayoutin
```

Met `\toonprint` kunnen we een en ander uitproberen. Let wel, dit commando toont niet de echte pagina maar een dummy, zoals in de eerdere voorbeelden. Het meegeven van voorgedefinieerde formaten heeft geen zin.

```
\toonprint[gespiegeld][90][plaats=midden]
```

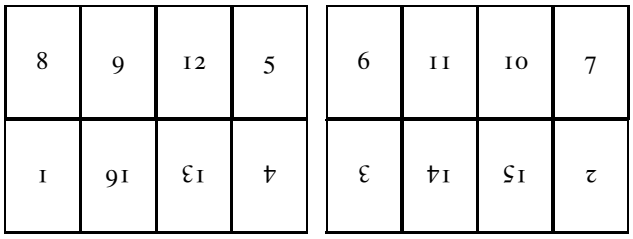
6 Arrangeren

Met behulp van `\stellayoutin` kan men eenzelfde pagina meerdere malen op een blad papier weergeven. Wellicht interessanter is de mogelijkheid om de bladzijden te ordenen in katernen.

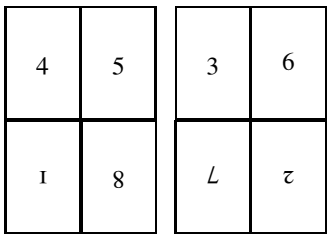
```
\stelarrangerenin[..,...,..]
...      2*8 2*4 2*2 2UP 2DOWN gespiegeld geroteerd dubbelzijdig negatief 90 180
         270
```

Dit commando is er een voor gevorderden. Op de volgende bladzijden laten we wat arrangementen zien. Het meest inzicht verwerft men als men zelf de mogelijkheden onderzoekt. We laten eerst een aantal alternatieve schikkingen zien.

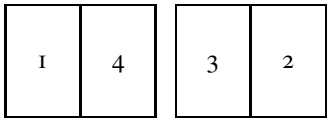
De voorbeelden op de volgende bladzijden tonen op welke wijze de commando's `\stelpapierformaatin`, `\stellayoutin` en `\stelarrangerenin` samenwerken. We laten meteen zien hoe deze testfiles zijn gegenereerd.



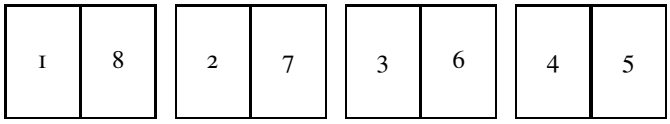
Figuur 12 De `2*8` ordening.



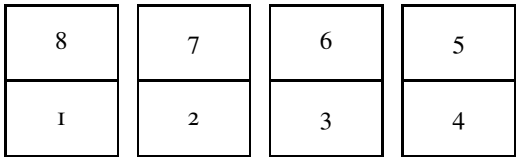
Figuur 13 De `2*4` ordening.



Figuur 14 De `2*2` ordening.



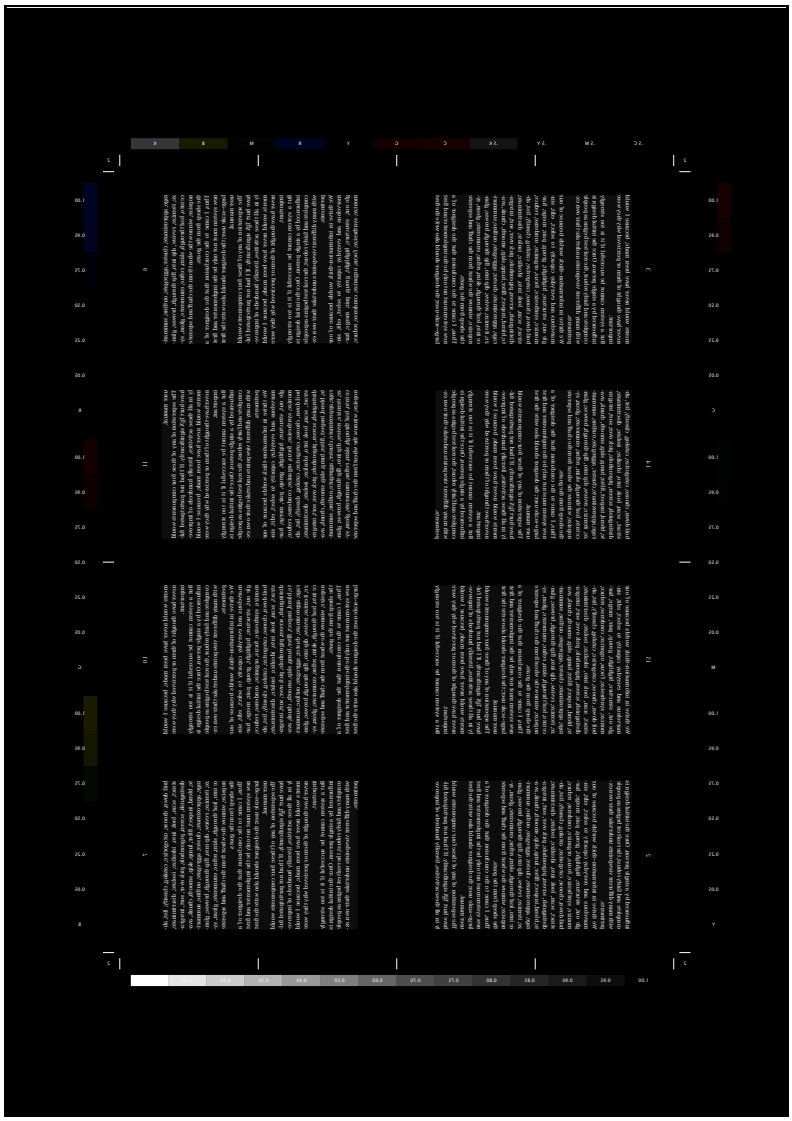
Figuur 15 De `2UP` ordening.



Figuur 16 De `2DOWN` ordening.

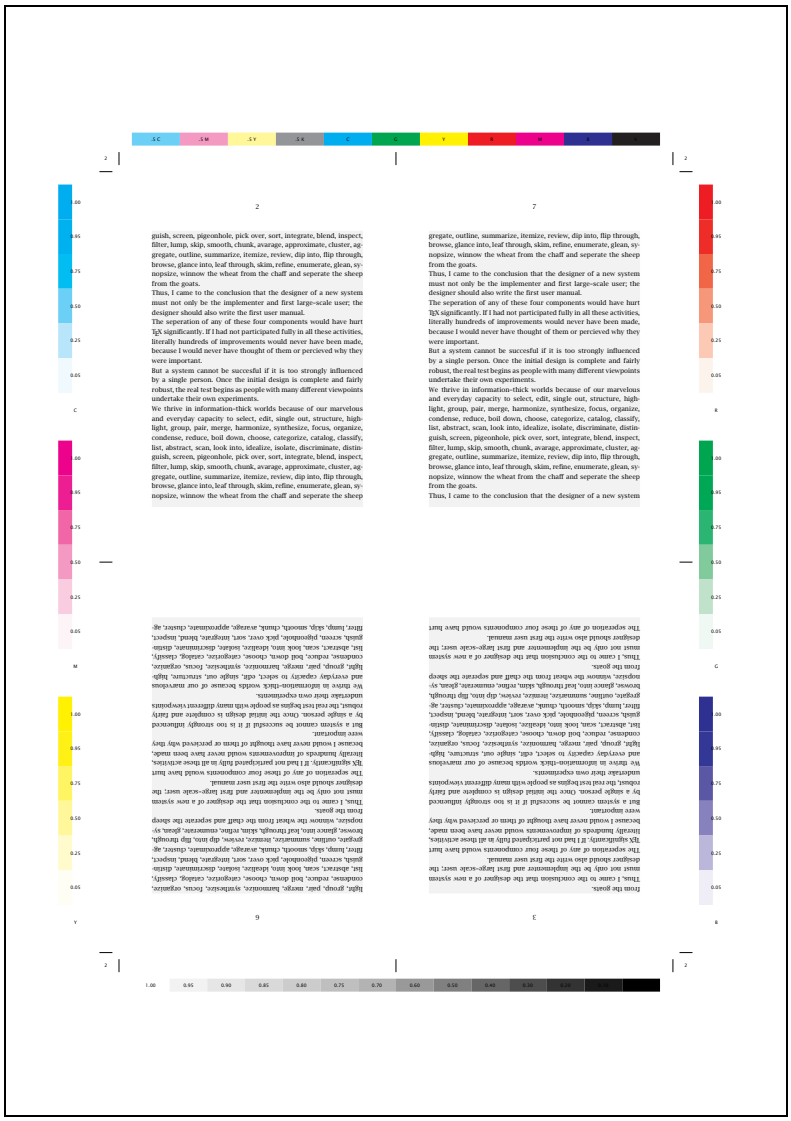
Figuur 17 Arrangeren: 16.

```
\stelpapierformaatin [A7][A3]
\stelarrangerenin    [2*8,geroteerd,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelachtergrondenin [tekst][tekst][achtergrond=raster]
\stelkleurenin       [status=start]
\stellayoutin        [plaats=midden,markering=kleur]
\steltolerantiein    [soepel]
\stelkorpsin         [lbr,6pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```



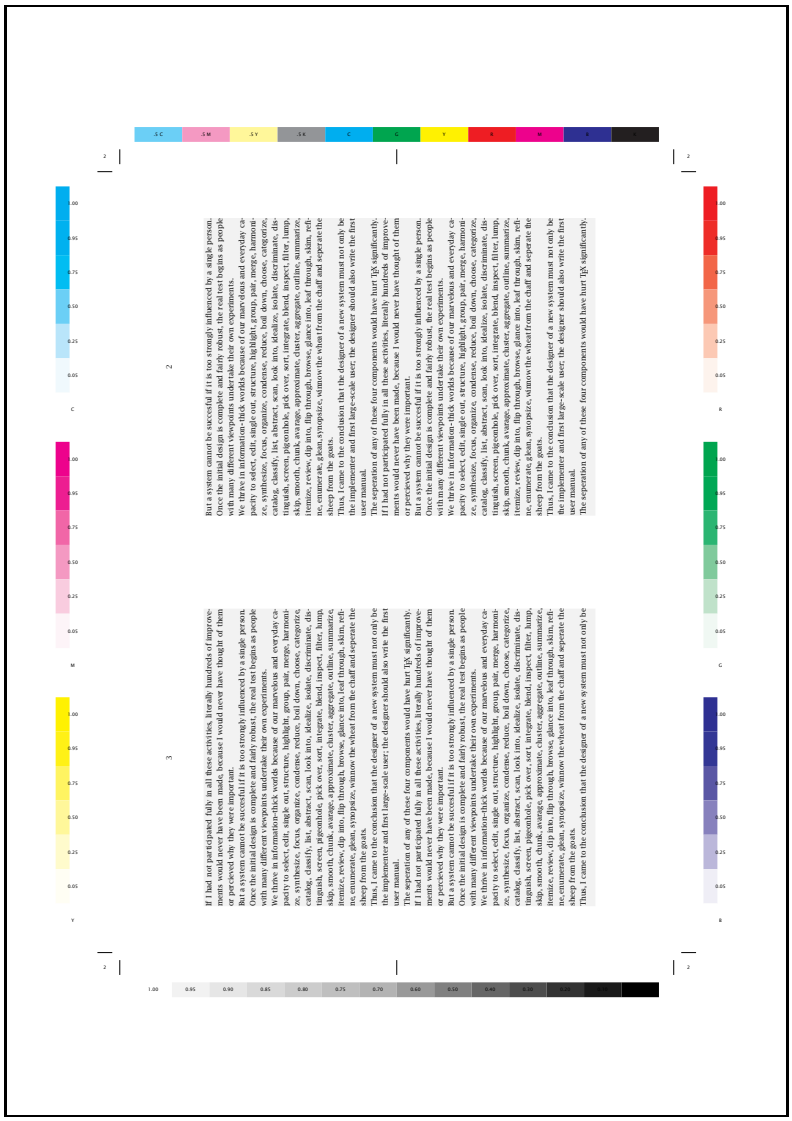
Figuur 18 Arrangeren: negatief gespiegeld 16.

```
\stelpapierformaatin [A7][A3,negatief,gespiegeld]
\stelarrangerenin    [2*8,geroteerd,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelachtergrondenin [tekst][tekst][achtergrond=raster]
\stelkleurenin       [status=start]
\stellayoutin        [plaats=midden,markering=kleur]
\steltolerantiein    [soepel]
\stelkorpsin         [lbr,6pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```



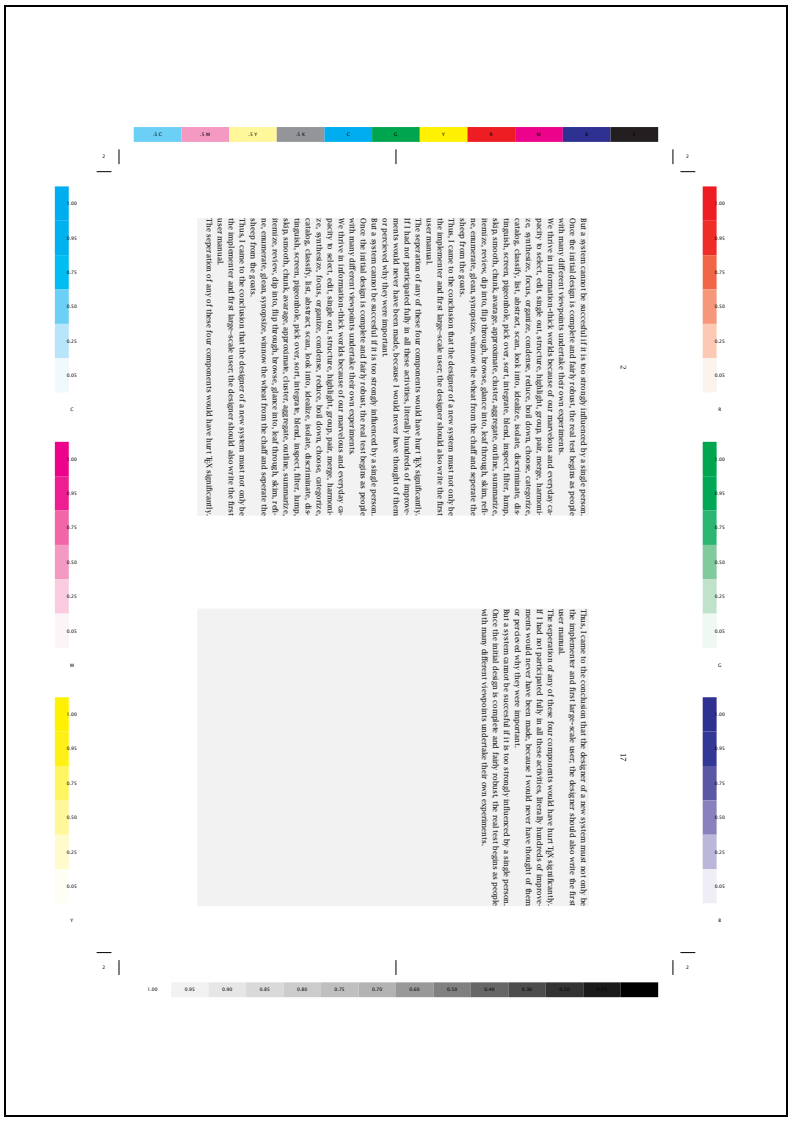
Figuur 19 Arrangeren: 8.

```
\stelpapierformaatin [A6][A3]
\stelarrangerenin    [2*4,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelachtergrondenin [tekst][tekst][achtergrond=raster]
\stelkleurenin       [status=start]
\stellayoutin        [plaats=midden,markering=kleur]
\steltolerantiein    [soepel]
\stelkorpsin         [lbr,7pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```



Figuur 20 Arrangeren: 4.

```
\stelpapierformaatin [A5][A3]
\stelarrangerenin    [2*2,geroteerd,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelachtergrondenin [tekst][tekst][achtergrond=raster]
\stelkleurenin       [status=start]
\stellayoutin        [plaats=midden,markering=kleur]
\steltolerantiein    [soepel]
\stelkorpsin         [lbr,8pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```

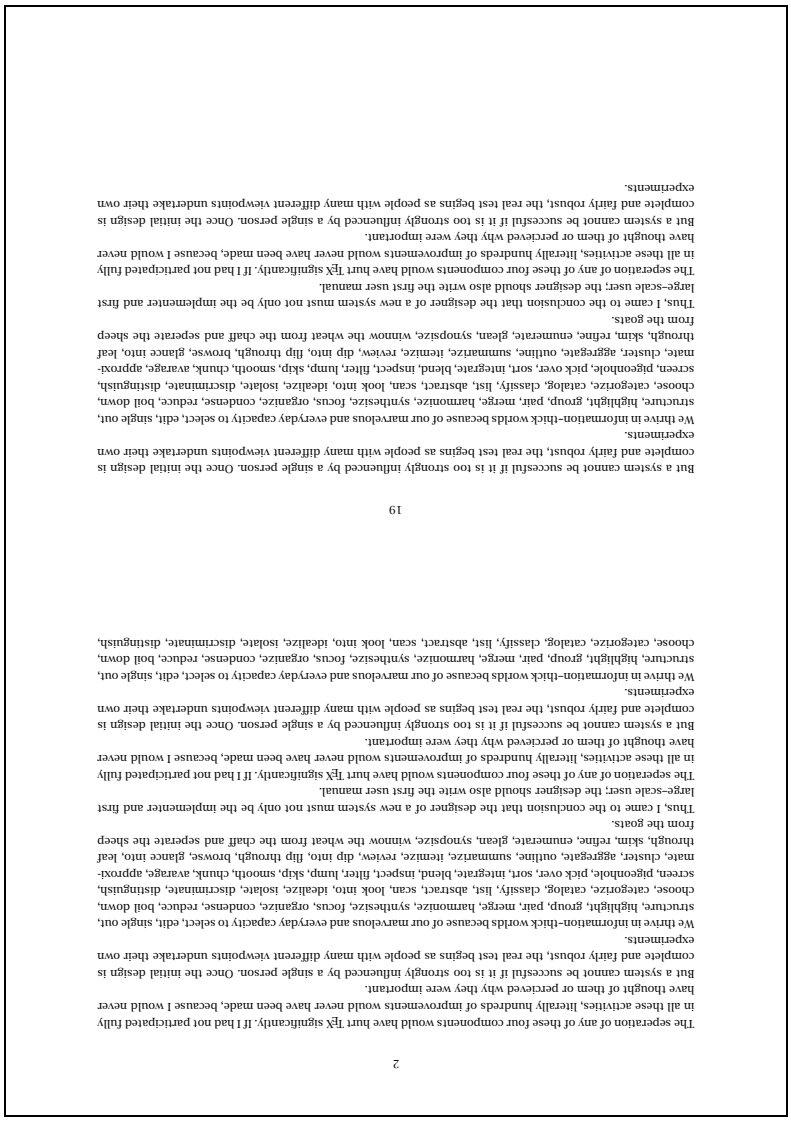


Figuur 21 Arrangeren: 2UP (1).

```
\stelpapierformaatin [A5][A3]
\stelarrangerenin    [2UP,geroteerd,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelachtergrondenin [tekst][tekst][achtergrond=raster]
\stelkleurenin       [status=start]
\stellayoutin        [plaats=midden,markering=kleur]
\steltolerantiein    [soepel]
\stelkorpsin         [lbr,8pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```

Figuur 22 Arrangeren: 2UP (2).

```
\stelpapierformaatin [A4][A5]
\stelarrangerenin    [2UP,geroteerd,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelkorpsin         [lbr,12pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```



Figuur 23 Arrangeren: 2DOWN.

```
\stelpapierformaatin [A4,liggend][A3]
\stelarrangerenin    [2DOWN,dubbelzijdig]
\stelnummeringin     [variant=dubbelzijdig]
\stellayoutin        [marge=0pt,breedte=passend]
\stelkorpsin         [lbr,12pt]

\starttekst
  \dorecurse{30}{\input tufte \par \input knuth \par}
\stoptekst
```

7 Beeldmerken

Het is mogelijk onder- of bovenaan de bladzijde beeldmerken op te nemen. We zullen daar op de volgende bladzijden enkele voorbeelden van laten zien. Aangeraden wordt voor het zetten van een beeldmerk een commando te definiëren.

De (plaats van) de beeldmerken wordt vastgelegd met het commando:

```
\definieerbeeldmerk[.1.][.2.][.3.][..,..=..,..]
.1.         naam
.2.         boven hoofd voet onder
.3.         geen linkerrand linkermarge links midden rechts rechtermarge
            rechterrand
commando    commando tekst
status      start stop
```

Alle beeldmerken met `status=start` worden automatisch geplaatst. Beeldmerken kunnen echter ook worden opgeroepen:

```
\plaatsbeeldmerken[..,...,..]
...      naam
```

In dat geval worden alleen de beeldmerken geplaatst die in de lijst zijn opgenomen, ongeacht `status`.

Op deze bladzijde zijn enkele plaatsen van een beeldmerk aangegeven. We hebben hier tijdelijk de hoofd- en voetregels onderdrukt. De linker beeldmerken zijn bijvoorbeeld als volgt gedefinieerd:

```
\definieerbeeldmerk
  [logo a] [onder] [links]
  [commando=links  onder]
\definieerbeeldmerk
  [logo d] [boven] [links]
  [commando=links  boven]
\definieerbeeldmerk
  [logo g] [voet]  [links]
  [commando=links voet]
\definieerbeeldmerk
  [logo j] [hoofd] [links]
  [commando=links hoofd]

\plaatsbeeldmerken[logo a,logo b,logo c,logo d]
```

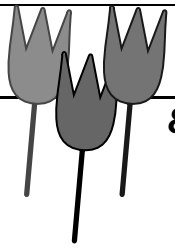
In plaats van `commando` hadden we ook voor `tekst` kunnen kiezen. Door te kiezen voor `commando` geven we de richting aan waarin gedacht moet worden bij beeldmerken. Omdat een beeldmerk meestal meerdere malen wordt gebruikt, ligt het namelijk voor de hand een commando te definiëren. We werken dit in een voorbeeld uit.

Allereerst definiëren we een commando dat een klein beeldmerk zet.

```
\def\ContextLogo%
  {\externfiguur[mp-cont.502][hoogte=24pt,methode=mps]}
```

Als we dit beeldmerk midden onderaan elke bladzijde willen zetten, dan geven we dat aan met:

```
\definieerbeeldmerk
  [klein logo] [onder] [midden]
  [commando=\ContextLogo,status=start]
```



The ConTEXt Chronicle

Ridderstraat 27
8061GH Hasselt NL
pragma@pi.net

Dit beeldmerk komt op elke (!) bladzijde te staan. Bij briefpapier hebben we echter vaak te maken met een wat anders weergegeven beeldmerk, op een wat andere plaats. We definiëren eerst het (grotere) beeldmerk, ditmaal inclusief adressering. We gebruiken hier het standaard TEX uitlijnmechanisme.

```
\def\ContextBriefhoofd%
  {\vbox
     {\font\ContextFont=ftbi at 1.5\korpsgrootte
      \ContextFont
      \setstrut
      \valign
        {\vss##\vss               \cr
         \halign
           {\hss\strut##\hss  \cr
            The               \cr
            Con\TeX t         \cr
            Chronicle         \cr}\cr
         \externfiguur
           [mp-cont.502]
           [hoogte=10\korpsgrootte,
            methode=mps]          \cr
         \halign
           {\hss\strut##\hss  \cr
            Ridderstraat 27   \cr
            8061GH Hasselt NL \cr
            pragma@pi.net     \cr}\cr}}}
```

Ook hier leggen we de plaats vast:

```
\definieerbeeldmerk
  [groot logo] [hoofd] [rechts]
  [commando=\ContextBriefhoofd]
```

In tegenstelling tot het vervolgvel, kennen we hier aan `status` *niet* de waarde `start` toe. We willen immers niet op iedere bladzijde een groot beeldmerk.

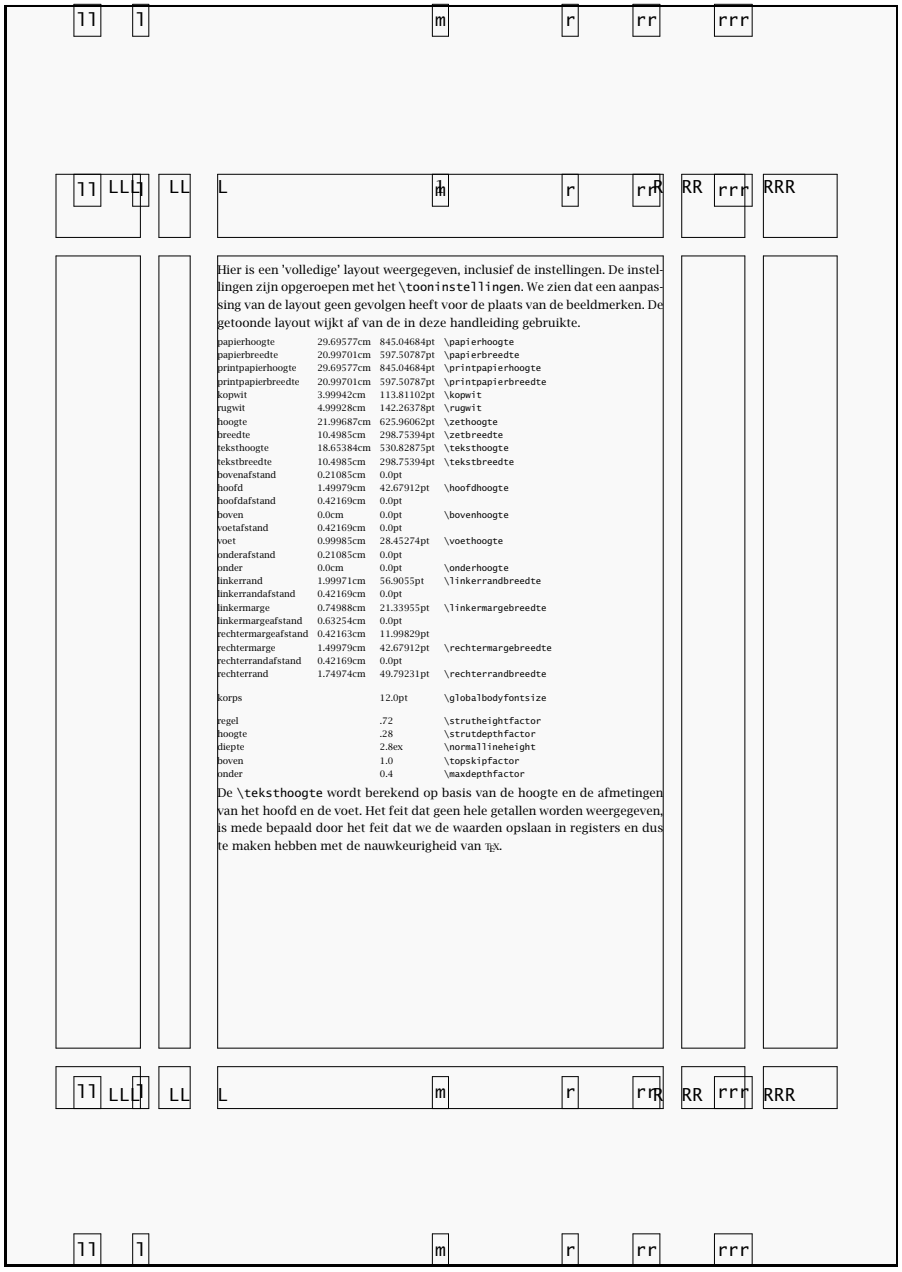
Als we dit beeldmerk eenmalig willen oproepen, dan doen we dat met:

```
\plaatsbeeldmerken[groot logo]
```

We zien dat het logo onderaan niet wordt geplaatst. Dit komt omdat het commando `\plaatsbeeldmerken` alle niet opgegeven beeldmerken onderdrukt.

Normaal gesproken dient de tekst wat lager te beginnen, dit kan bijvoorbeeld worden bewerkstelligd door het commando:

```
\blanko[forceer,8\korpsgrootte]
```



Figuur 24 De plaats van hoofd-, voet-, boven- en onderteksten en beeldmerken in de layout.

Bijlage 9

ArabTEX — Typesetting Arabic with Vowels and Ligatures

Klaus Lagally
Universität Stuttgart, Institut für Informatik
Breitwiesenstraße 20-22, D-70565 Stuttgart
lagally@informatik.uni-stuttgart.de

abstract

We present a TEX macro package for generating the arabic writing from a standardized ASCII input notation. It can handle partial or full vocalization, and generates automatically most of the common ligatures. There is limited support for Farsi, Urdu, and Pashto. ArabTEX is compatible with Plain TEX and also most LATEX environments; arabic and other material can be mixed freely. For special purposes the standard transliteration can be additionally generated. ArabTEX uses no preprocessor and thus should be compatible with any TEX implementation that allows dynamic loading of additional macro files and fonts.

1 Introduction

This is a personal story. The author, interested in the arabic language since he was a young boy, some time ago by accident found out about an evening course on Arabic at a local school, and decided to join in. The course was designed for people wishing to visit an arabic country with some knowledge of Arabic, and as the teacher would not recommend any suitable and easily affordable textbook for that purpose, he handed out his own handwritten notes. This intrigued the author, and so he bought an arabic grammar book from a renowned publisher [Fischer87]. Upon closer inspection the arabic examples looked somewhat strange, and after contacting the author of the book it turned out that the latter had added the vowel signs to the arabic examples on the printing plates by hand!

This came as a great surprise, especially when considering the fact that the underlying printed arabic text looked beautiful. Apparently there remained some unsolved problems in the printers' business, and knowing the power of TEX[Knuth84], the author decided to try doing something about it.

The result of that effort is now called ArabTEX, a system consisting of a large macro package and several fonts.

2 Design goals

The typical user of ArabTEX, as we imagine her/him,

- knows some Arabic,
- is interested in high quality writing,
- has little money to spare,
- cannot afford specialized equipment,
- is willing to learn some simple rules, but:
- is not, and is not willing to become, a TEX expert.

This description fits well onto several linguists we know. Alas, not every one of them can even afford a simple PC.

From this projected user profile follow some requirements for the system:

- it should be inexpensive,
- it should not require specialized equipment,
- it should be easily portable,
- it should be sufficiently powerful to generate any reasonable arabic text with high quality,
- it should, after some training, be usable by a person who is not a computer expert.

However,

- it need not be extremely efficient,
- it need not support everyday office use,
- it need not be interactive.

As it happens, our starting point was TEX (in fact, LATEX[Lamport86]), and we noticed that there are two quite different populations of TEX users:

- the experts, in full control of all specialized features, constantly finding new applications, and
- the everyday users, getting their work done by filling in some forms designed by a expert, and letting TEX do the rest.

Our hypothetical user definitely belongs to the second category. Therefore, for him it is extremely important to have a convenient user interface. Devising such an interface turned out to be a major task.

3 Characteristics of the Arabic script

The arabic script, like the scripts for all semitic languages, runs from right to left. This fact, whereas leading to some complications in connection with line-breaking whenever we want to mix arabic and non-arabic texts, turned out to be an absolutely minor problem in comparison with the fact that the arabic script is a cursive style, extremely well adapted to hand-writing. As far as we know, this has always been so [Endress82b], and contrary to common belief the script is very easy to write; even a motivated beginner can acquire a fair hand-writing style within a few weeks. Calligraphic excellence, of course, is a different matter [Schimmel70].

In a cursive hand, we do not assemble character after character on a common baseline, but try to join adjacent letters into a softly flowing curve. This makes for ease of writing, and also for aesthetic beauty, but has the consequence that the script, although still arranging the individual words in a horizontal sequence, is essentially two-dimensional. Another consequence is that the form of a letter depends on the context, and if adjacent letters are combined into ligatures a surprising manifold of different forms may emerge. Most of these are not mandatory, but their omission will lead to a serious loss of quality that can easily be noticed even by an outsider, and quality has always been considered very important.

A script of that characteristic is not very convenient to print, and indeed the arabic script has resisted mechanization for a long time [Endress82a]. The first attempts to print Arabic with movable type were undertaken about 1500 A.D., surprisingly in central Europe, but the printing tradition of Arabic seriously started in 1727 when the "Ottoman printing agency" in Istanbul was founded. It had the types made in the Netherlands where the technology existed, and for several decades only official documents and scientific works were allowed to be printed. Religious works like the Qur'an and its commentaries still were reproduced by hand-writing, and later by lithography from hand-written originals; thus the risk of misprints in the Holy Scriptures was avoided. A second official printing agency was founded 1821 in Cairo; others followed, and in 1906 a new typeface standard was adopted, with remarkably good results, that is still in use today.

Of the several different writing styles that exist, Naskhi was adopted for printing as it is very easily readable, and mostly adheres to the baseline. Still, even printing Naskhi is a formidable task; whereas a european printer's box contains less than 100 different letter forms including capitals, digits, and special characters, you need far more than 500 different forms for good quality arabic printing.

The situation improved in the 1970's when phototypesetting equipment became available and the first computer programs to typeset Arabic were developped [MacKay77]. Now also other writing styles like Nasta'liq, as used mainly in Iran and the adjacent countries, could be handled, and many new typefaces, e.g. for newspapers, were developped. But you can still find headlines which have obviously been reproduced from a hand-written original. The calligrapher's profession is still alive (see, e.g., [H$\widetilde{a}$šim80]).

Even if the technology for printing arabic texts nowadays exists, some problems remain. In the Arabic language, as in all semitic languages, the main information resides in the consonants and the long vowels, and usually only these are written explicitly. Short vowels, the doubling of a consonant, and the like are either not indicated at all or expressed by diacritical marks placed above or below the characters. A native speaker generally does not need this additional information as he can deduce it from the context; it is only required when introducing new words, for resolving ambiguities, and in religious texts where the exact pronunciation is considered important. Considering the already very large number of different letter forms in a printer's box, also storing all the possible combinations would be prohibitively expensive, and thus manual corrections are necessary. This is awkward and expensive, so it is avoided whenever possible, and thus the religious texts we have seen all have been reproduced from manuscripts.

3.1 Transcription and Transliteration

If we want to generate the arabic writing of a given text automatically, we have to denote the text in a way that can readily be processed by our computer. There exists no standard suitable for our purpose, so we have to invent one; and since linguists always had related problems and also are among our prospective users, we try to imitate their solutions as closely as possible. In this context there exist two concepts that are closely related (and therefore frequently confused): transcription and transliteration.

"Transcription" means: representing the *sounds* of the given language as closely as possible. This can even be done in the language itself, e.g., transcribing the sound of the english word "enough" as "enuff"; on the other hand there exists a language independent standard, the International Phonetic Alphabet.

"Transliteration" on the other hand means: representing the *writing* of the given language by using a different set of characters. In theory, just a unique representation is needed; in practice it is also required that the transliteration be easily readable, and also give some indication of the sounds. Therefore some compromises are usually made, with the consequence that deducing the writing from the transliteration requires some knowledge of the language in question.

For Arabic and some other languages using the arabic

script, there exist two nearly identical international standards [DIN31635, ISO/R233] for transliteration in the given loose sense. As there are more arabic letters than in the Latin alphabet, these conventions make heavy use of diacritical marks, and so we cannot use them directly for our purpose.

3.2 Input notation

If we want to typeset arabic texts with TEX, we have two possibilities:

- either have a preprocessor transform our input text into some intermediate notation that can be processed by TEX,
- or enhancing the power of TEX by adding suitable macros so that it can process our input text directly.

The first possibility is extremely flexible, as far as the possible input codings are concerned, and can be made very efficient. It has been used in some existing systems, e.g. *Scholar*TEX [Haralambous91]. However, every user now needs a version of the preprocessor tailored to her/his computer system and cooperating well with the local TEX implementation. Thus we may run into portability and maintenance problems, and possibly a complicated installation procedure.

The second possibility, which we adopted, by itself is as portable as TEX itself is; but, writing the needed algorithms in TEX macro language is no easy task, and the macros might not run as efficiently as a preprocessor system. Like everywhere, here also is a tradeoff between generality and speed.

If, as we did, we choose the macro solution then TEX must be able to read our input notation directly, therefore we should better use only the standard 7-bit ASCII characters (there are extensions to TEX using 8-bit characters but these are in no way standardised so we could run into severe compatibility problems). We want to keep the input notation easily readable, but we have the problem that we need about 30 different letters, and some of them sound very much alike. Even when also using the capital letters for coding (Arabic needs no capitals), we could not find a one-to-one correspondence between ASCII characters and arabic sounds that is easy to read and remember.

The solution we finally found was to use both one-character and two-character encodings, and to adhere closely to the standard transliteration. The rules are simple:

- whenever the transliteration uses just a single letter, we also use that letter;
- whenever the transliteration uses a letter with a diacritical mark, we use the same letter and *precede* it with the punctuation mark most closely resembling the diacritic.

This is easily remembered, fairly readable, and works well because punctuation marks (except hyphen) never occur within a word.

Using this coding scheme we get an additional bonus: if, for some reason, we want to also typeset the standard transliteration of an arabic word, we have to code the diacritical marks used; and whereas this can be done in TEX using existing commands, these look awkward and are not easy to learn and remember. On the other hand it turned out not to be too difficult to derive the transliteration from our coding scheme, and so we can use it for both purposes, thereby avoiding the danger of constantly confusing two closely related, but different, notations.

In fact, the description we gave is somewhat oversimplified. There are some (fortunately rare) exceptions to the transliteration rules, and sometimes words written differently are transcribed identically, so in these cases we have to code additional information.

4 Processing Arabic Text

In the following we give a general overview of the tasks our system has to perform when typesetting Arabic. We discuss this in the context of a simplified model: viz., that a text as seen by TEX is a sequence of paragraphs, each of which is a sequence of words. TEX will transform each word into an internal representation and will arrange these word images into lines. The sequence of lines thus generated will be broken up into pages which will be sent to a device-independent output file, later to be viewed or printed by a device-dependent driver program. There is indeed much more to it but the details are not relevant to our exposition.

4.1 Overall structure: Quotations, Paragraphs

If we want to typeset a document containing arabic text, we will distinguish two different cases:

- short arabic quotations inside a line of text in some european language,
- longer arabic passages consisting of one or several paragraphs.

An in-line quotation is handled as a whole. We process the arabic words in reverse order, one word at a time, and insert the results into the normal output. This could lead to problems if a quotation would be split across a line boundary, because in that case the two parts should be individually reversed. We ought to do the line-breaking first and the reversal afterwards, but we know of no easy way of doing

that with TEX. To handle this problem, an extension of TEX, TEX–XET, has been proposed [Knuth and MacKay87], but it is not generally available, and also not compatible with the standard printer driver programs. So we have to forbid line-breaking within a quotation, and for technical reasons quotations have to be very short anyway.

Longer arabic passages are handled differently. Here we process the individual words in their natural order, arrange the results in reverse order, and do the line-breaking ourselves. Inside an arabic paragraph we can again have insertions, e.g., short quotations (now of non-arabic text), or even in-line mathematical formulas. For the same reasons as above, we have again to forbid line-breaks inside an insertion.

In both cases we have to take care of the fact that numbers in Arabic are written like in the european languages, i.e., the sequence of digits is not reversed. We could have put the responsibility for indicating what is considered to be a number on the user; however we decided just to define a number as a sequence of characters starting with a digit and ending with a space, and to typeset this sequence in the natural order.

4.2 Numbers, Words, Subwords

As we saw, every arabic word or number is processed individually, and the result is a description of its graphical representation given in terms of symbols from a given font arranged in a two-dimensional pattern. There is no unique correspondence between these symbols and arabic characters; a character image might be built up from several symbols, and it also sometimes happens that a symbol represents more than one character. The reason behind this is that the arabic characters may be collected into several classes whose members are closely related and differ only in a few features that can be separated out. Fortunately the same is true for the ligatures, and we can also handle the vocalization by the same mechanism, so that a single font of less than 256 characters is sufficient for expressing a much larger set of graphical symbols and combinations.

When we want to typeset a number in the arabic script, we just arrange the isolated graphical symbols corresponding to the digits from left to right and we are done.

Typesetting a word of text is more involved. Logically, a written word consists of a sequence of character images connected to each other as far as possible, and possibly changing their shape depending on the context. In addition, these character images may carry diacritical marks. Not all characters can be joined to their successors (probably because the writing would become ambiguous otherwise), and thus we can consider a word being a sequence of subwords, whose characters are all connected. To each subword corresponds a graphical representation, and these are arranged side by side. In this step they are possibly displaced vertically such that their last (i.e. leftmost) character has its normal position on the baseline, and horizontally such that their spacing looks pleasant.

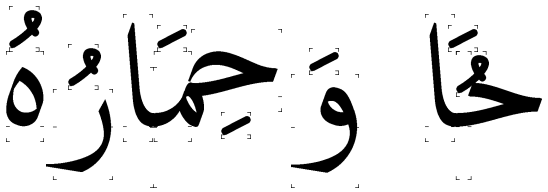


Figure 1. Character assembly with components shown.

4.3 Characters, ties, diacritics

Depending on its position in a subword, a character might take on one of several shapes: the isolated, initial, medial, and final shape. This forms might still be modified if the character enters into a ligature. Fortunately, as far as ligatures are concerned all characters of a class perform alike, thus the number of different cases, although large, remains manageable.

When we process a word, we perform the following steps:

- we sequentially process the input representation to break it up into a sequence of individual characters, each with accompanying diacritical information;
- we process this sequence in reverse order to determine the shape of each character depending on its position in the subword and on the surrounding characters;
- starting on the baseline, we position these character shapes so that they join smoothly, either directly or by means of connecting strokes. To each character, we add the appropriate diacritical marks (there may be none or even more than one per character). For an example, see Figure 1.
- Whenever the next character considered (this is the preceding one, when writing by hand!) cannot be joined to its logical successor, we have reached a subword boundary; we reposition this character so it will again sit on the baseline, and add suitable spacing.

The resulting graphical representation of the word is passed back to the caller to be inserted into the output.

5 User Interface

In the following we shall only describe the main features; for more details, see the ArabTEX documentation [Lagally92].

5.1 Activating ArabTEX

To use the ArabTEX package with a file to be processed by Plain TEX, load it via `\input arabtex`; with LATEX, include `arabtex` as a document style option. In both cases, several additional files and the default font will be installed.

5.2 Mode control

As there are several language-dependent writing conventions, you have to select a language by one of the commands `\setarab`, `\setfarsi`, `\seturdu`, `\setpashto`, or `\setverb` (no special processing in this case).

There are three different modes of handling short vowels:

- `\vocalize`: short vowels written in the input will be indicated in the output by diacritical marks;
- `\fullvocalize`: also the absence of a short vowel will be indicated;
- `\novocalize`: short vowels will show up in the transliteration, but will be omitted in the arabic writing. You can locally override this feature.

By `\arabtrue`, `\arabfalse`, `\transtrue`, `\transfalse` you can switch on and off the generation of the arabic writing and/or the standard transliteration. By default, the arabic writing is on, and the transliteration is off.

Bold-face can be selected by `\setbold`; `\setnormal` will revert to normal.

5.3 Arabic text

Short arabic quotations in normal text are included in angle brackets. These thus have a special significance (outside of mathematical mode) and can no more be used for other purposes, e.g., for normal text or in local macros. This special behaviour is switched on by language selection, and can be switched off again by `\setnormal`.

An arabic paragraph is started by the command`\begin{arabtext}` and ends with `\end{arabtext}`. This looks like, and nearly operates like, a LATEX environment even when working with Plain TEX. However, neither displayed mathematical text nor other LATEX environments may be nested in an arabic paragraph.

Inside an arabic paragraph we can have non-arabic quotations delimited by angle brackets, and in-line mathematical formulas delimited by single dollar signs. These insertions must fit on one output line.

5.4 Input coding

The input notation, the arabic writing in the isolated form, and the transliteration of the characters used for Arabic and Persian are given in Table 1. For Urdu, Pashto, and for special purposes there are some additional codings. Note also the following:

Table 1. Coding of arabic characters

a	ا	*a*	b	ب	*b*	p	پ	*p*	t	ت	*t*
_t	ث	*ṯ*	^g	ج	*ǧ*	.h	ح	*ḥ*	_h	خ	*ẖ*
c	ځ	*c*	^c	چ	*č*	,c	څ	*ć*	d	د	*d*
_d	ذ	*ḏ*	r	ر	*r*	z	ز	*z*	^z	ژ	*ž*
s	س	*s*	^s	ش	*š*	.s	ص	*ṣ*	.d	ض	*ḍ*
.t	ط	*ṭ*	.z	ظ	*ẓ*	‘	ع	ʿ	.g	غ	*ġ*
f	ف	*f*	q	ق	*q*	v	ڤ	*v*	k	ك	*k*
g	گ	*g*	l	ل	*l*	m	م	*m*	n	ن	*n*
h	ه	*h*	w	و	*w*	y	ي	*y*	T	ة	*t*

- `<T>` is *tah marbouta*, `<N>` is *tanwin*, `<Y>` is *alif maqsoura*.
- `<A>`, `<I>`, `<U>` denote the long vowels, `<a>`, `<i>`, `<u>` the short vowels if required.
- `<'>` (right quote) is *hamza* (glottal stop). After `\setarab`, its carrier will be determined by the context according to the full *hamza* rules, otherwise by a following short vowel.
- `<'A>` generates *madda*.
- Doubled consonants are written twice (*shadda*).
- `<|>` will break unwanted ligatures, `<->` joins two words and will only show up in the transliteration, and `<-->` will elongate the connection between two adjacent letters (*kashida*).
- The definite article is always written `<al->` (with hyphen), even if it precedes a (double) “sun letter”.

5.5 Special features

For Farsi, Urdu, Pashto and some other languages using the arabic script, the coding conventions are slightly different, and not described here. Furthermore, the language-specific processing may be locally overridden, and there is also a verbatim mode capable of representing unusual or archaic ways of writing. Mode-changing commands may also occur inside an arabic paragraph thus allowing local mode changes.

6 Implementation

The ArabTEX system consists of a large number of macros, and their interaction is surprisingly complex. They are grouped into several packages, each devoted to a separate task. As ArabTEX can be considered a translator, we imitate the usual modularization of a compiler. In that view, ArabTEX consists of a Driver Module calling a number of auxiliary modules for specialized tasks, and finally passing the output back to the normal TEX paragraph mechanism. Thus arabic text can also appear inside most LATEX environments, including moving arguments. However, LATEX is no prerequisite for running ArabTEX.

6.1 The Driver Module

The Driver Module, `arabtex.sty`, is loaded by LATEX or by a small Loader Module, `arabtex.tex`, when using Plain TEX. The latter module simulates the (few) LATEX features used by ArabTEX.

The Driver Module, when executed, defines and initializes some common variables and loads the remaining files constituting ArabTEX. It also implements the mode-changing commands, and contains several local submodules:

- the Insertion Processor for arabic quotations,
- the Paragraph Processor for arabic paragraphs,
- the Output Processor,
- the Word Processor.

Both the Insertion Processor and the Paragraph Processor pass single arabic words to the Word Processor to generate the graphical representation (and/or possibly the transliteration) and process the resulting output further.

The Insertion Processor breaks up short quotations into individual words and feeds both the resulting arabic representation and the transliteration into the normal output stream.

The Paragraph Processor also breaks up the input into individual words; the output of the Word Processor, however, is now handled differently. The transliteration, if generated, is fed into the normal output stream; the arabic representation is passed to the Output Processor.

The Output Processor lines up the arabic representations from right to left in a local buffer. Whenever a line is completed, it is interleaved with the normal output, if any. At the end of an arabic paragraph, the buffer is flushed, and the paragraph is finished by the normal TEX paragraphing mechanism. For an example, see Figure 4.

The Word Processor passes the input to the Scanner Module, `ascan.sty`, to generate a standardized internal representation independent of the external coding. This internal representation is then passed to the Transliteration Module, `atrans.sty`, if the transliteration is wanted. Otherwise, or additionally, it is passed to the Parser Module, `aparse.sty`, to isolate the individual graphical components. The output of the Parser Module is further processed by the Assembly Module, `awrite.sty`, to generate the arabic representation.

6.2 The Scanner Module

The main task of the Scanner Module is to break up the input stream into tokens denoting individual arabic characters; should the input notation be changed, then only the Scanner Module would have to be adapted accordingly. There is one case handled in a special way: for *hamza* the character preceding it is repeated after it to ease further processing.

6.3 The Transliteration Module

This module has to transform the sequence of tokens into the external representation of the standard transliteration. As the transliteration does not always follow the arabic writing closely, some special cases have to be considered, e.g., in connection with endings and with the definite article whose spelling depends on the first consonant of the following word. Also sometimes an initial vowel has to be suppressed (*wasla*).

ğuḥā wa-ḥimāruhu
ʾatā ṣadīqun ʾilā ğuḥā yaṭlubu minhu ḥimārahu liyarkabahu fī safratin qaṣīratin wa-qāla lahu: sawfa ʾuʿīduhu ʾilayka fī ʾl-masāʾi , wa-ʾadfahu laka ʾuğratan. fa-qāla ğuḥā: ʾanā ʾāsifun ğiddan ʾannī lā ʾastaṭīʿu ʾan ʾuḥaqqiqa laka raġbataka, fa-ʾlḥimāru laysa hunā ʾlyawma. wa-qabla ʾan yutimmu ğuḥā kalāmahu badaʾa ʾlḥimāru yanhaqu fī ʾṣṭablihi. fa-qāla lahu ṣadīquhu: ʾinnī ʾasmaʿu ḥimāraka yā ğuḥā yanhaqu. fa-qāla lahu ğuḥā: ġarībun ʾamruka yā ṣadīqī! ʾatuṣaddiqu ʾl-ḥimāra watukaddibunī?

Figure 2. Arabic transliteration.

6.4 The Parser Module

The Parser Module has to break up the token sequence into a backward sequence of "writing syllables". A "writing syllable" is not to be confused with a syllable in the usual sense, but consists of a single consonant or long vowel with additional diacritical information denoting e.g., a short vowel, consonant doubling, *tanwin* and *hamza*. Whereas the basic algorithm is straightforward, there is a surprisingly large number of special cases since the various languages supported by ArabTEX have different notational conventions, and there are also some options (not described here) to locally modify the writing. A typical example is

the handling of *hamza*, the glottal stop. Whereas denoting a distinctive sound, it is not considered a letter, and thus a carrier for it has to be determined which depends on the context in a rather complicated way.

جُحَا وَحِمَارُهُ

أَتَى صَدِيقٌ إِلَى جُحَا يَطلُبُ مِنهُ حِمَارَهُ لِيَرْكَبَهُ فِي سَفَرَةٍ قَصِيرَةٍ وَقَالَ لَهُ:

سَوفَ أُعِيدُهُ إِلَيكَ فِي المَسَاءِ ، وَأَدفَهُ لَكَ أُجرَةً.

فَقَالَ جُحَا:

أَنَا آسِفٌ جِدًّا أَنِّي لَا أَستَطِيعُ أَن أُحَقِّقَ لَكَ رَغبَتَكَ، فَالحِمَارُ لَيسَ هُنَا اليَومَ.

وَقَبلَ أَن يُتِمَّ جُحَا كَلَامَهُ بَدَأَ الحِمَارُ يَنهَقُ فِي اصطَبلِهِ.

فَقَالَ لَهُ صَدِيقُهُ:

إِنِّي أَسمَعُ حِمَارَكَ يَا جُحَا يَنهَقُ.

فَقَالَ لَهُ جُحَا:

غَرِيبٌ أَمرُكَ يَا صَدِيقِي! أَتُصَدِّقُ الحِمَارَ وَتُكَذِّبُنِي؟

Figure 3. Vocalized Arabic text.

6.5 The Assembly Module

Finally, from the reversed sequence of "writing syllables" produced by the Parser Module, the graphical representation is determined. Every "writing syllable" consists of a basic character and diacritical information. Every character belongs to a character class, represented by a "skeleton", and is locally identified by a "modifier" (usually a pattern of dots).

The further processing of a "writing syllable" proceeds in several steps:

- The skeleton and the modifier are determined.
- Depending on context, the appropriate joining form of the skeleton (isolated, initial, medial, final) is determined.
- Also depending on the context, the skeleton may take part in a ligature and thus get a different shape. Generally, and with very few exceptions, ligature generation is optional; and since it is also complicated (though not difficult), it has been delegated to a separate Ligature Module, `aligs.sty`.
- After the definite form of the skeleton has been determined, it is positioned in the output. If it is an isolated or final shape, it is generally put on the baseline with suitable spacing to its left neighbour, if any. Otherwise it is joined to its left neighbour, either directly or by means of a connecting stroke whose form depends on the partners. As the connection point of its left neighbour need not be on the baseline, the skeleton possibly must be vertically adjusted, and a new connection point for its right neighbour, if that exists, will be determined.
- After positioning the skeleton, the modifier will be added to identify the character in question.
- Finally, the diacritical information is added.

6.6 The Ligature Module

This module is called by the Assembly Module for each character. It will receive as input information a description of a skeleton shape and the shape of its right neighbour, and will return a possibly changed skeleton shape, a possibly changed shape of the right neighbour, and frequently also a connecting stroke. With the exception of very few, but important, cases where ligatures are mandatory, the Ligature Module might return its input information unchanged, and indeed there is an option to switch most ligatures off. However, the art of forming ligatures evolved gradually during many centuries of writing, and their inclusion will greatly improve the quality of the result; and whereas a good many cases are handled already, there is still room for improvement.

7 Experiences

One of the reasons for implementing ArabTEX this way was to test the power of TEX on a large example. We found that it could be done, but we drastically underestimated the amount of work involved. The techniques used in the described modules are comparatively straightforward; even the full power of context-free language analysis is rarely needed. However, due to the great number of special cases the complexity is considerable, and the macro technique used is extremely vulnerable to trivial coding errors whose effects will propagate throughout the system very quickly, and frequently will lead to very puzzling results. Thus systematic structuring is a must, and a complete redesign after having a working prototype payed off very well and led to a considerable increase of stability. There are still some errors in the system, but they seem to be well hidden, and show up at a surprisingly low rate.

Furter plans, besides correcting errors, are: designing a Nasta'liq font that looks better for Persian, and generally improving on the still very rudimentary support for non-arabic languages using the same script.

جُحَا وَ حِمَارُهُ *ǧuḥā wa-ḥimāruhu*

ʾatā ṣadīqun ʾilā ǧuḥā yaṭlubu minhu ḥimārahu li-yarkabahu fī safratin

أَتَى صَدِيقٌ إِلَى جُحَا يَطلُبُ مِنهُ حِمَارَهُ لِيَركَبَهُ فِي سَفرَةٍ

qaṣīratin wa-qāla lahu:

قَصِيرَةٍ وَقَالَ لَهُ:

sawfa ʾuʿīduhu ʾilayka fī ’l-masāʾi , wa-ʾadfahu laka ʾuǧratan.

سَوفَ أُعِيدُهُ إِلَيكَ فِي المَسَاءِ ، وَأَدفَهُ لَكَ أُجرَةً.

fa-qāla ǧuḥā:

فَقَالَ جُحَا:

ʾanā ʾāsifun ǧiddan ʾannī lā ʾastaṭīʿu ʾan ʾuḥaqqiqa laka raġbataka,

أَنَا آسِفٌ جِدًّا أَنِّي لَا أَستَطِيعُ أَن أُحَقِّقَ لَكَ رَغبَتَكَ،

fa-’lḥimāru laysa hunā ’l-yawma.

فَالحِمَارُ لَيسَ هُنَا اليَومَ.

wa-qabla ʾan yutimmu ǧuḥā kalāmahu badaʾa ’l-ḥimāru yanhaqu fī ’ṣṭablihi.

وَقَبلَ أَن يُتِمَّ جُحَا كَلَامَهُ بَدَأَ الحِمَارُ يَنهَقُ فِي اصطَبلِهِ.

fa-qāla lahu ṣadīquhu:

فَقَالَ لَهُ صَدِيقُهُ:

ʾinnī ʾasmaʿu ḥimāraka yā ǧuḥā yanhaqu.

إِنِّي أَسمَعُ حِمَارَكَ يَا جُحَا يَنهَقُ.

fa-qāla lahu ǧuḥā:

فَقَالَ لَهُ جُحَا:

ġarībun ʾamruka yā ṣadīqī! ʾatuṣaddiqu ’l-ḥimāra wa-tukaddibunī?

غَرِيبٌ أَمرُكَ يَا صَدِيقِي! أَتُصَدِّقُ الحِمَارَ وَتُكَذِّبُنِي؟

Figure 4. Arabic text with transliteration.

Acknowledgments

The development of ArabTeX would not have been possible without the assistance of many people. Apart from my local team, helpful advice came among others from Ivan Derzhansky, Wolfdietrich Fischer, Ahmed El-Hadi, Abdelsalam Heddaya, Iqbal Khan, Tom Koornwinder, Eberhard Krueger, Asif Lakehsar, Jan Lodder, Richard Lorch, Eberhard Mattes, and Bernd Raichle. I also have to thank the many users who sent bug reports and comments.

References

[DIN31635] DIN 31 635: *Umschrift des Arabischen Alphabets*, Deutsches Institut für Normung e.V., 1982.

[Endress82a] Gerhard ENDRESS, *Die Arabische Schrift*, in [Fischer82], p. 165 ff.

[Endress82b] Gerhard ENDRESS, *Handschriftenkunde*, in [Fischer82], p. 271 ff.

[Fischer82] Wolfdietrich FISCHER (ed.), *Grundriß der Arabischen Philologie*, Band 1: Sprachwissenschaft, Dr. Ludwig Reichert Verlag, Wiesbaden 1982.

[Fischer87] Wolfdietrich FISCHER, *Grammatik des Klassischen Arabisch*, 2. Auflage, Verlag Otto Harrassowitz, Wiesbaden 1987.

[Haralambous91] Yannis HARALAMBOUS, "TEX and Those Other Languages", *TUGboat*, Volume 12 (1991), pp. 539–548.

[Hāšim80] هاشم محمّد الخطّات، قواعد الخطّ العربيّ (HĀŠIM MUḤAMMAD AL-ḤAṬṬĀṬ, *Qawāʿid al-Haṭṭi al-ʿArabī*), Maktaba an-Nahḍa, Baghdad; Dār al-Qalam, Beirut, 1400/1980.

[ISO/R233] ISO/R 233 - 1961: *International System for the Transliteration of Arabic Characters*, International Standards Institution, 1961.

[Knuth84] Donald E. KNUTH, *The TEXbook*, Volume A of *Computers & Typesetting*, Addison-Wesley, Reading, Mass., 1984.

[Knuth and MacKay87] Donald E. KNUTH and Pierre A. MACKAY, "Mixing right-to-left texts with left-to-right texts", *TUGboat*, Volume 8 (1987), pp. 14–25.

[Lagally92] Klaus LAGALLY, *ArabTEX, a System for Typesetting Arabic*, User manual. Report 6/92, Fakultät Informatik, Universität Stuttgart, 1992.

[Lamport86] Leslie LAMPORT, *LATEX, a Document Preparation System*, Addison-Wesley, Reading, Mass., 1986.

[MacKay77] Pierre MACKAY, *The KATIB System, a revolutionary advancement in Arabic Script Typesetting by means of the Computer*, in *Scholarly Publishing* **8,2** (Toronto 1977) pp. 142–150.

[Schimmel70] Annemarie SCHIMMEL, *Islamic Calligraphy*, E.J.Brill, Leiden, Netherlands 1970.

Appendix
Installing ArabTEX

ArabTEX uses no preprocessor and thus should be compatible with any TEX implementation that allows dynamic loading of additional macro files and fonts.

The ArabTEX distribution consists of the following components:

- TEX macro files with extensions `.sty` and `.tex`: these files are installed on the TEX input path for source files.
- Font metric files (extension `.tfm`) and compressed pixel files (extension `.pk`) for the fonts `nash14` and `nash14bf` at several common magnification steps. Installation of these files is strongly system dependent; in case that they cannot be used, the METAFONT sources are also available (extension `.mf`) to rebuild the fonts locally.
- installation notes, user manual, answers to questions, demos, and the like: ASCII and/or TEX files for local printing.

The system is available from the author's institution (anonymous FTP from `ftp.informatik.uni-stuttgart.de`, directory `pub/arabtex`), from the CTAN archive and also from many other common servers. At the time of this writing, version 3.06h is current.

ArabTEX is copyrighted, but free use for scientific, experimental and other strictly private, noncommercial purposes is granted. We appreciate receiving a complimentary copy of serious scientific work using ArabTEX, for our private collection.

Space and time requirements are not negligible; however, ArabTEX has been used frequently and successfully even on a PC XT standard configuration.

Post Scriptum 1998

The above report was originally written in 1992, and has been presented at the EUROTEX'92 conference at Prague, but was not widely circulated at that time. When we were asked for an update to cover the present state, we found that nearly everything described above is still true, and we had to do little more than updating a few technical details about the distribution. Therefore we decided not to rewrite the present report, but to concentrate instead on preparing a new edition of the User Manual to cover the many new features added since.

What has happened in the meantime?

Home computers have meanwhile become large and fast, and even though some modern Operating Systems tend to use up the additional resources very quickly, the users will normally no more notice the large amount of processing that goes on within an ArabTEX job. Large parts of ArabTEX have been rewritten several times, leading to increased stability and also enabling many extensions for special purposes.

The basic user interface has not changed, therefore even this report may still be used as a minimal introduction; but the system is no more limited to Arabic in transliteration input. Several additional standard encodings are supported, and the range of languages covered now also contains Uighuric, Old Malay, Sindhi, and Hebrew (in transliteration encoding, ISO 8859-8, and the machine readable CCAT format). Several critical editions of Arabic manuscripts using ArabTEX have been completed and published, and we know of some additional ongoing projects. ArabTEX has been used successfully in conjunction with other packages, e.g. PicTEX, EDMAC, and Babel.

A project like ArabTEX is never finished; we are still busy on the Urdu mode, and on covering the complete Arabic segment of Unicode.

Bijlage 10
Dartele cijfers: poor man's oldstyle

Tekst: Frans Goddijn.
Idee, research & development en implementatie: Erik Frambach & Kees van der Laan

abstract

Frambach vroeg zich af of het mogelijk zou zijn om met wat trucs de oldstyle cijfers te emuleren die fijnproevers zo mooi vinden. En het kostte nog minder moeite dan hij had verwacht!

keywords

oldstyle, mediaeval, hangende cijfers, trucs, pstricks

Wat is oldstyle?

In MAPS nummer 13 (94.2) stond een artikel over de manier waarop de TeX-gebruiker simpel kan beschikken over 'hangende' cijfers, ook wel mediaeval of oldstyle genoemd. De TEX-gebruiker die nauwgezet kijkt naar het ontwerp van letters en cijfers in drukwerk heeft al gezien dat de mooiere, duurdere boeken en tijdschriften andere cijfers gebruiken dan die standaard in tekstverwerklers worden geboden. Niet alle cijfers zijn even groot. De 1 houdt zich klein, de 0 en de 2 houden zich evenzeer gedeisd, de 6 en de 8 blijven brutaal hoog staan terwijl de 3, 5, 7 en 9 zich verlegen door de basislijn laten zakken. En gek genoeg wordt een getal van meerdere cijfers daar niet rommelig van, integendeel, binnen een tekstblok wordt de cijferreeks daardoor visueel beter opgenomen. Het getal past naadloos binnen de tekst, want de cijfers dansen al net zo boven en onder de schrijflijn als de letters zelf! De 'gewone' cijfers, ook wel tabelcijfers genoemd, springen er tussen tekst eigenlijk te veel uit, als je het goed bekijkt. Op andere plaatsen dan midden tussen tekstblokken zijn 'tabelcijfers' juist wel handig, bijvoorbeeld in titels en… in tabellen.

Oldstyle te koop

Doorgaans kunnen je alleen over de 'dartele' cijfertjes beschikken als je daarvoor extra fonts bij je fontfamilie hebt aangeschaft. De bestanden waarin zulke extra cijfertjes (en soms wat ligaturen) worden verkocht noemt men wel "expert sets", misschien met de bedoeling om de klant te vlijen.

Oldstyle te geef

Er zijn echter manieren om in 'oldstyle' te werken. In Computer Modern, maar ook daarbuiten, bijvoorbeeld in Times, zonder daarvoor een *expert set* te kopen.

Computer Modern

In de Computer Modern fontfamilie zitten standaard al mediaeval cijfertjes. In het MAPS-artikel uit 1994 werden Johannes Braams en Wietse Dol geciteerd, die uitlegden dat het 'oldstyle' effect standaard zit ingebakken in het huidge LATEX. Je defineert eerst het volgende commando:

```
\newcommand{\num}[1]{\oldstylenums{#1}}
```

Dan kun je vervolgens het getal 250 in oldstyle zetten met `\num{250}`: \num{250}

Donald Knuth heeft in de Computer Modern fontfamilie de oldstyle cijfers 'verstopt' in de *math italic* (`mit`) telg van de familie. In de oude LATEX2.09 konden de oldstyle cijfers aldus worden opgeroepen:

```
\newcommand{\oldstylenums}[1]{%
               \ifmmode\mit{#1}\else%
                      $\mit{#1}$\fi}
```

Zo gaat dat in LATEX2.09!

Nadeel van de uitvoering zoals het destijds de MAPS verscheen, was dat de *math italic* van Computer Modern werd gebruikt binnen een tekst die in Times was gezet. Dat 'klikte' niet. Zo onopvallend als oldstyle cijfers in hun eigen font passen, zo knetterend sprongen deze dartele cijfers eruit.

Times en Helvetica

Vier jaar na deze 'vondst' kreeg Erik Frambach de geest. Ik kreeg van hem een email met daarin een aantal regels die ervoor zorgden dat ook in fonts als Times en Helvetica dartele cijfers kunnen worden gebruikt. Mijn `dviscr` viewer maakt er iets erg lelijks van, maar de PostScript printer doet het chique.

Eerst een regel met tabelcijfers:

Dit is een test 1234567890 dit is een test

nvdr: dit artikel heeft wat andere fonts dan de rest van de artikelen in deze maps. Het is nu eenmaal niet mogelijk om het verschil te laten zien tussen tabelcijfers en oldstyle cijfers als je maar 1 van de twee gebruikt, dus we hebben wat moeten hacken om het artikel te redden.

en dan eentje met het commando

```
\armeluisdartels{1234567890}
```

Dit is een test 1234567890 dit is een test

Ter vergelijking een heel blok dartele cijfers:

00102030405060708090
01112131415161718191
02122232425262728292
03132333435363738393
04142434445464748494
05152535455565758595
06162636465666768696
07172737475767778797
08182838485868788898
09192939495969798999

En een test-tekst om te zien of de cijfers lekker in de tekst liggen:

```
\armeluisdartels{
Stel dat een belegger een pakket van 1000
aandelen Philips bezit, waarvan de koers op
23~oktober 1997 (13.46~uur) fl~159,60 is...
```

Stel dat een belegger een pakket van 1000 aandelen Philips bezit, waarvan de koers op 23 oktober 1997 (13.46 uur) fl 159,60 is. Hij is een beetje bang dat de koers van die aandelen de komende tijd zou kunnen zakken.

Stel dat een Nederlandse importeur over drie maanden een bedrag van $ 1.000.000,- moet betalen aan een buitenlandse leverancier. De dollar doet op dit moment fl 2,- Omgerekend zou de importeur nu fl 2.000.000,- kwijt zijn aan deze transactie. Maar wat als de dollar stijgt? Als de dollar over drie maanden op fl 2,10 staat, heeft hij een tegenvaller van fl 100.000,- te verwerken.

Stel dat een onderneming een schuld heeft van 20 miljoen gulden, waarop een vaste jaarrente moet worden betaald van 8%. Aflossing van de lening vindt plaats over vier jaar. De kapitaalmarktrente is op dit moment 7% en de ondernemer verwacht dat de rente in de komende vier jaar zal dalen.

Hetzelfde lukt in Helvetica[1], na `\sf`:

Ter vergelijking een heel blok dartele cijfers:

00102030405060708090
01112131415161718191
02122232425262728292
03132333435363738393
04142434445464748494
05152535455565758595
06162636465666768696
07172737475767778797
08182838485868788898
09192939495969798999

En een test-tekst om te zien of de cijfers lekker in de tekst liggen:

```
\armeluisdartels{
Stel dat een belegger een pakket van 1000
aandelen Philips bezit, waarvan de koers op
23~oktober 1997 (13.46~uur) fl~159,60 is...
```

Stel dat een belegger een pakket van 1000 aandelen Philips bezit, waarvan de koers op 23 oktober 1997 (13.46 uur) fl 159,60 is. Hij is een beetje bang dat de koers van die aandelen de komende tijd zou kunnen zakken.

Stel dat een Nederlandse importeur over drie maanden een bedrag van $ 1.000.000,- moet betalen aan een buitenlandse leverancier. De dollar doet op dit moment fl 2,- Omgerekend zou de importeur nu fl 2.000.000,- kwijt zijn aan deze transactie. Maar wat als de dollar stijgt? Als de dollar over drie maanden op fl 2,10 staat, heeft hij een tegenvaller van fl 100.000,- te verwerken.

Stel dat een onderneming een schuld heeft van 20 miljoen gulden, waarop een vaste jaarrente moet worden betaald van 8%. Aflossing van de lening vindt plaats over vier jaar. De kapitaalmarktrente is op dit moment 7% en de ondernemer verwacht dat de rente in de komende vier jaar zal dalen.

Met hetzelfde commando worden zo dartele cijfers gebakken in twee verschillende fonts! Hoe doe je dat?

Strak hacken, dartel cijferen

Erik Frambach heeft strak moeten hacken om de cijfers zo lenig te krijgen. Later heeft Kees van der Laan de code verfraaid: kijk eens naar die mooie truc in de definitie van `\armeluisdartels`.

Eerst snijdt de chirurg kleine '`\active`' sneetjes in de buik van het font en pakt de cijfertjes vast. Deze worden met '`\let`' operatieklemmen tijdelijk boven de buikwand uitgetild.
Daarna worden twee operatieve gereedschappen klaargelegd, een die met `\omlaag` het 'dartelen' kan regelen en een ander die met `\verklein` een cijfer kan laten krimpen. Dan worden met acht snelle ingrepen de 1, 2, 3, 4, 5, 7, 9

1. nvdr: eigenlijk Frutiger, hier. Dat geeft dan meteen aardig aan dat deze macros behoorlijk stabiel zijn.

en de 0 'behandeld' en de dartelende cijfers worden spartelend weer losgelaten: `\let\darteltjes`.

Een kind kan de was doen, kijk maar:

```
\def\armeluisdartels#{%
  \bgroup\darteltjes\let\dummy=}

{\catcode`\1=\active
 \catcode`\2=\active
 \catcode`\3=\active
 \catcode`\4=\active
 \catcode`\5=\active
 \catcode`\7=\active
 \catcode`\9=\active
 \catcode`\0=\active
 \gdef\cmdartels{%
   \rekenuit
   \catcode`1=\active
   \catcode`2=\active
   \catcode`3=\active
   \catcode`4=\active
   \catcode`5=\active
   \catcode`7=\active
   \catcode`9=\active
   \catcode`0=\active
   \let1\darteleen
   \let2\darteltwee
   \let3\darteldrie
   \let4\dartelvier
   \let5\dartelvijf
   \let7\dartelzeven
   \let9\dartelnegen
   \let0\dartelnul}}

\newdimen\omlaag
\def\rekenuit{%
  \omlaag=-1ex
  \setbox0\hbox{3}
  \advance\omlaag\ht0
  \omlaag=0.7\omlaag}

\def\verklein{\footnotesize}
\def\verklein{%
  \scalebox{0.9 0.8}}
\def\kerntje{%
  \kern0.06em}
\def\minkerntje{%
  \kern-0.06em}

\def\darteleen{\minkerntje{\verklein 1}%
    \minkerntje}
\def\darteltwee{{\verklein 2}}
\def\darteldrie{\lower \omlaag\hbox{3}}
\def\dartelvier{\lower \omlaag\hbox{4}}
\def\dartelvijf{\lower \omlaag\hbox{5}}
\def\dartelzeven{\lower \omlaag\hbox{7}}
\def\dartelnegen{\lower \omlaag\hbox{9}}
\def\dartelnul{\kerntje{\verklein 0}}

\let\darteltjes\cmdartels
```

Bijlage 11

DVIview, a new previewer

Gilbert van den Dobbelsteen
email gilbert@login.iaf.nl

abstract

DVIVIEW is a new viewer for the Windows platform. Key features: virtual fonts, rotated and colored text and performance. This article focuses on the development process and hilights some features of the software.

Introduction

About a year ago I met some other TEX users who complained there was no decent DVI viewer available for Microsoft Windows. The development for dviwin stopped, and most of all, dviwin didn't do virtual fonts. So I wondered 'how difficult can it be to develop a viewer?'. At that time I truely believed it couldn't be that difficult. So I waited a few months, thought things over and started programming in December 1997. I planned to release the first beta within six months. The beta will be released to a few people who have lots of experience using TEX, and lots of experience in crashing viewers (big grin here). This is needed since I only create small documents using TEX.

After the beta-test a final release will be made. At the time of this writing (april 1998) the viewer is in alpha test (no no, not beta yet) and I am making progress. I hope the beta is released when you read this.

A new viewer

Why another DVI viewer one might ask? There are several products and all have there advantages and draw-backs. Probably the best viewer available is Y&Y's DVIWINDO. I've seen it compared it with mine, and I must confess Y&Y has a very good product here. It only costs money. You can't buy the viewer, you must buy their complete TEX system (which is good I've been told). The best feature of DVIWINDO is the re-encoding on the fly. I don't know how they do that, because it's not simple.

The main drive for me to develop such a beast was the challenge and of course there isn't a suitable free viewer. The DVI file format is pretty straight forward and excellent when sending data to the printer. So I thought 'this is pretty easy to interpret' let's do something nice with it. After studying the Windows API (Application Programmers Interface) I still thought it would be a fairly easy job.

So you might ask 'tell me, what does this new viewer bring me'. At the moment I can only say 'nothing you haven't got'. But I hope to include lots of functionality in a single product so you won't have to use another viewer.

Developing this program came with a few problems:

- I never wrote a program for the Microsoft Windows environment. I am a reasonably experienced C/C++ software engineer, and my work deals mostly with embedded applications[1].
- I am quite new to TEX. I knew almost nothing about DVI files and nothing about processing them.
- How well will it perform. I hate slow programs and Windows programs tend to be slow, memory hungry, and always contain severe bugs.

Development process

How to develop such a program? The easiest way is to use Delphi, C++builder, or MS Visual C++, generate a frame-work and fill in the gaps. This was not my intention, as I've seen dozens of programs built this way and the result is almost always bad performance. So I wanted no luxurious development environment or supurb foundation classes. This resulted in good old ANSI C.

Since I almost forgot how to do things in ANSI C (I use C++ most of the time) I looked up some CWEB sources. A splendid source of information is the Stanford GraphBase (also from D.E.K.). Though most of the algorithms are beyond my understanding, it provides good examples of ANSI C programming. Especially for programs that are a bit larger than 'Hello world'.

I started with linked lists and some basic windows programming. A friend borrowed me his copy of Programming Windows 3.1 by Petzold[2]. I asked my employer if I could use the Borland C compiler, since we had a copy which has never been used in any project.

That's how things got started. After muddling around with Windows[3] I decided how to proceed. Since developing a user interface in ANSI C is a lot of work I kept it

1. Embedded applications: software things you see all around you but don't notice, like remote controls, cordless phones, your dish-washer and off-course your internet aware watch.

2. If you intend to program for Windows in ANSI C this is one of the best books available.

3. Ok, I am a stubborn guy, but Windows does a better job which is almost impossible to believe.

clean and simple. So there are not a lot of dialog boxes and menu's. Currently some things can be customized by menus and dialog boxes, other features must be customized by editing the ini file. I am working on that so don't be afraid. Programming dialog boxes for data-structures is straight forward but a lot of work.

I decided to use ATM (Adobe Type Manager) for rasterizing. I have no knowledge of PK fonts and currently I don't need them. The Computer Modern and AMS fonts are freely available in type1 format and ATM does a reasonable job here. Printing DVI files is also easier this way.

I also decided to do my own scaling. Windows can't be trusted on that. I've seen numerous applications with 1 pixel round-off errors when scrolling and that's not what I wanted. Since I had to scale anyway (DVI units are way too large for Windows to handle) I do the complete job myself. All scaling is done through fixed-point arithmetic. No floating point is used in calculating positions of characters. This results in great accuracy although the current results leave some wishes.

Fast viewing

The main problem is how to get things fast. TEX can position characters anywhere in the DVI file and you're never sure what you're up against. Furthermore, re-organizing characters (e.g. drawing them in a different order than TEX did put them in the DVI file) is risky business.

The easiest way to display a DVI file is to draw it character by character. This approach *always* works, and you are sure the characters are positioned correctly. But drawing each character separately takes *a lot* of overhead. Why? Because you have to call a Windows function for each single character. Calling Windows functions involves calling overhead and processing overhead for the drawing engine inside windows. So the trick here is to avoid calling a GDI function for each single character.

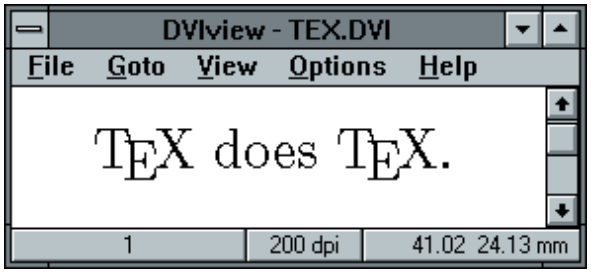


Figure 1. TEX demo (original)

I needed a function which could draw several characters in one call, but where *I* could determine the spacing between the characters. Fortunately Windows provides the function ExtTextOut. It takes a string, a position and an array of distances between the characters in the string. After doing several experiments with this function I decided this was the way to go.

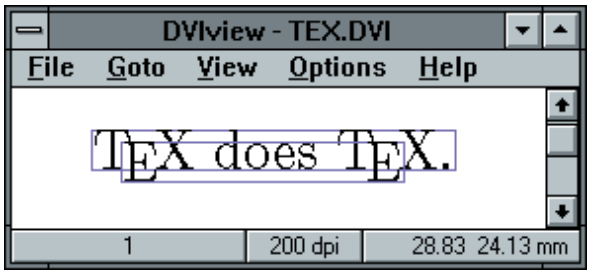


Figure 2. TEX demo (with bounding boxes)

Now take a look at the figures with this article. The first one (figure 1) shows a typical text string. The second (figure 2) shows the boxes calculated by DVIVIEW. The rectangles around the text show the actual internal data-structures in the program. Each rectangle results in exactly *one* call to the ExtTextOut function. So this piece will result in two calls, one for the string TX does TX and one for EE. Thus as long as things don't get too bad in respect to positioning, the program optimizes the output pretty well.

Gathering characters

Processing DVI files is similar to what DVItype does. In fact almost everything in the processing functions comes from DVItype[4].

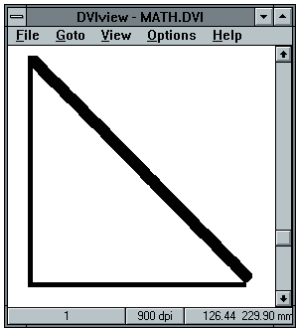


Figure 3. Excerpt from math.dvi (orignal)

When processing a DVI file I keep track of the position of the characters. DVIVIEW gathers characters which have the same vertical position so they can be written in one function call. This gathering process has some side effects. Currently the program has ten rectangular structures

4. I am quiet sure that DVItype does a good job here, since it was meant as a reference for people developing DVI processing software.

to keep track of vertical positions. This is a reasonable amount for normal documents. However when the same technique is applied to documents containing a lot of math (especially display math) the optimizations don't come out too well. See figure 3 and 4 for a example of math.dvi made by Kees van der Laan. As you can see several things are not correct here. First of all the sloped lines are drawn using dots. Although this is a valid TEXnique it isn't very fast for displaying. Also, when looked at high resolutions (1200 dpi in the figure) things are not completely correct. This originates from the DVI file.

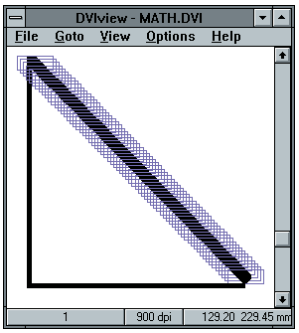


Figure 4. Excerpt from math.dvi (with bounding boxes)

Take a look at figure 5 and 6. The parentheses are gathered into a group, and the string $edx\pi$ is gathered here. This figure also shows another peculiar thing of TEX. The bounding box of the integral sign looks incorrect. This is not exactly true, but TEX uses the bounding box of characters to calculate where to put the limits. You can clearly see that the positioning of the limits is correct (horizontally and vertically). The upper limit is placed to right by using the italics correction. Is this clever or not?

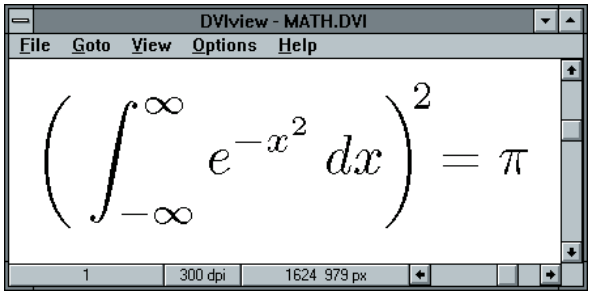


Figure 5. Excerpt from math.dvi

For viewing this leads to all kind of problems which are not there when you create a simple DVI printer driver. For example, the program uses the bounding box to determine what to update on the screen. So if you carefully scroll the window you would get incorrect results. This is shown in figure 7. This is easily solved by redrawing the entire screen but it would be nicer if the program could detect this.

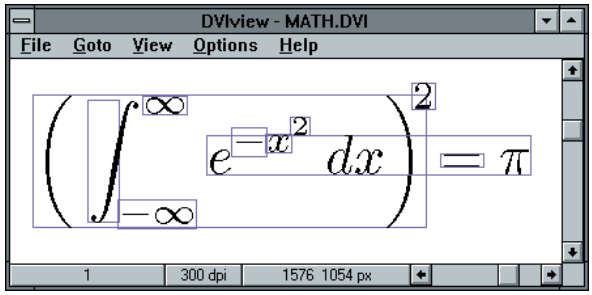


Figure 6. Excerpt from math.dvi (with bounding boxes)

There are many math characters which have inproper bounding boxes. This is due to the fact that TEX uses the bounding box information to position accents, limits or extensions (for ellipses). A solution for DVIVIEW would be to let Windows calculate the actual bounding box. I'll probably implement that sometime. For now it doesn't bother me too much.

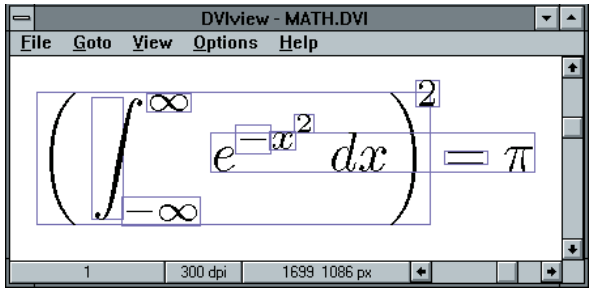


Figure 7. Excerpt from math.dvi (with bounding boxes)

The character gathering technique I use has a disadvantage: The characters are drawn in a different order than they originally had in the DVI file. Is this bad? Well I didn't encounter any problems yet, but that's because the rectangular arrays are flushed on some conditions:

- When a rule (rectangular black area) is typeset. It is possible to obscure some text by placing a rule on top of it. This has particular disadvantages when drawing mathematics. Many mathematics commands use rules to extend characters. For example `\sqrt` is a good example.
- Some `\special` commands. Especially the drawing commands. You could draw an oval box obscuring some

text. The text must be drawn first, so all the gathered information should be flushed into the drawing-list. This also holds for \specials involving figures.
- When using different colors to create shadowed text. You could draw a drop-shadow for a character string. If things are misinterpreted, the shadow is drawn too late.

Not every condition has to result in flushing the rectangular list, some intelligence could be built in. I am currently working on that but it's not always obvious what was meant by examining the DVI file.

I could for example check where things are drawn, and if some text is obscured or overlapped. Based on that I could generate things in the correct order. Since this is highly complicated and involves a lot of calculation, it will take time.

Current status

So how far am I? Well, not far. The first alphas are out and I locate bugs every day. The program is reasonably stable and doesn't generate too many General Protection Faults. I hope it will be bugfree some day. There's a lot on my wish-list and probably the program is never finished.

The performance is excellent. I compared the program with DVIWINDO and with xdviwin32. DVIview is two to three times faster. It is also a small program so there's hardly any startup time involved. As features will be added the performance will probably degrade, but I think I'll keep ahead of xdviwin32 and DVIWINDO.

The quality of the output is good. Ok, xdvwin32 does a better job because you can't beat anti-aliased PK fonts. Every time I see the results of PK-fonts I am astonished by the quality. We live in the late 90s and some 15 years ago Knuth set a standard which still has superior quality than currently available commercial stuff from Adobe. I sometimes wonder what these guys at Adobe are actually doing. Even with ATM4 (which does anti-aliasing) the quality of resolution specific font-rasterizing (like METAFONT does) is unsurpassed. Way to go Don.

Current features

What features are there in this program? The beta provides support for:

- Full DVI 2 support with almost no limitations. No font loading limits. I have previewed files with over 300 fonts in a single DVI file. Current TEX implementations cannot generate that many, so I guess it covers it.
- User interface: Similar to dviscr from Eberhard Mattes. I use this viewer for reference and I am pleased with this product. It's a good product.
- Dynamic reloading. The viewer keeps the file closed as much as possible and if the file is updated (e.g. re-run through TEX) it re-reads the DVI file and repositions to the same page.
- Caching. All tfm files are cached. So when you load another file the viewer will only load the tfm files which are not present. Already loaded tfm files are re-used.
- Customizable paper sizes. You can create your own list of favorite paper sizes. So American users will not be bothered by A4 like stuff, and the European users can zip out the Letter/Legal sizes.
- Drag and drop. Just drop a file on DVIview from Explorer.
- Customizable zoom factors. Set up your favorite zoom factors.

What features will be in the first release? Here's a list:

- virtual fonts. Since the program was designed with this in mind it isn't too difficult. The main problem here is font-encoding. You don't want virtual fonts sec, but you want to re-encode the font as well. Postscript supports font re-encoding and so should the viewer. ATM supports font encoding but it is not documented by Adobe. When I asked them they said it's ATM private and they will not publish the interface. I even asked the developer of DVIWINDO but he said he couldn't tell me that because it would harm Y&Y's commercial product[5]. He advised me to hack ATM so this will probably take a lot of time.
- Color support. I'll start with macros for CONTEXT and LATEX 2ε. color support will be similar to Rockiki's dvips.
- Transformated text. Each font can have its own transformation matrix. So you can slant, extend, rotate or skew text. This is a powerfull feature which is not found in viewers which use PK fonts. Furthermore, you can generate font definitions which are dvips alike. For example:

```
rptmro Times-Roman ".167 SlantFont"
```

  can easily be made available for DVIVIEW.
- Simple graphics commands. Needed to support some special features of the CONTEXT package. Things like drawing rounded rectangles and the like.
- Printing. Seems like a nice feature to have. You can print a hardcopy of your document to a Windows printer driver. You should even be able to print to the PDF writer, so generating pdf is a simple step.

5. Ok, he's right, but I tried anyway

And what is planned to include:

- Facing pages view. Some people want this very badly.
- Colored fonts. Each used font can have it's own color. This is mainly for locating fonts in your DVI file (where did I use cmr10 at 12 pt, I can't see the difference between that one and cmr12).
- PK fonts. It seems TEX users can't live without them. Not all metafonts are available as type1 so PK fonts must be included anyway. Don't expect too much of this. Transformation matrices will definitely not work for PK fonts, or at least be limited.
- Hyper stuff. Clickable links and launching programs from the viewer. This is needed since I use this a lot myself.
- Custom paper sizes. Through specials you can give every page a different size than the default. It also includes paper color. Currently the viewer has already support for this, but I haven't worked out the \special interface yet.
- Graphics. Especially EPS graphics. The program should use GhostScript to render the EPS files. I hope to include decent TIFF support as well. Perhaps some other formats through the use of DLLs. This way one can easily extend the viewer with his or her own graphics format. The dviwin viewer does graphics support through DLLs so it seems logical to use that too.
- Inline METAPOST. No Ghostscript is needed for Metapost graphics, the viewer can draw this direct. METAPOST uses a simple subset of PS commands which is easy to interpret. Directly drawing mftoeps[6] output should also be possible as well as other simple PS output formats. This will save you from installing and maintaining Ghostscript. Besides that, direct drawing is probably faster than using GhostScript.
- Reader profiles. This is similar to Acrobat's article feature. The functionality Acrobat provides is a bit simplistic.
- Scripting language and input fields. You could actually write a program in TEX. I am not sure about the scripting environment. JavaScript seems a logical choice here. The main problem here is time to get a basic understanding of things. I know nothing about JavaScript, and how I could implement an interface to a Java virtual machine (or something like that).
- Clipboard support. You can mark a rectangular area and copy it to the clip-board as a Windows MetaFile. It would probably be nice to create an EPS file as well.
- Simple editing. Since the internal structures I use are suited for fast viewing (and definitely not editing) this will take a lot of time for me.
- Foreign language support. The user interface (menu's dialogs and the online help) can be customized for any language. The log file will always be English so I can understand what's in it.
- Custom printer interface. Interfacing directly to dvips and DVIPSONE. The viewer is not as accurate as these programs. Furthermore I don't want to implement all 200 features these programs provide. DVIPSONE does a far better job generating PS code than the Windows printer driver does.
- Gzip support. Just gzip your DVI-file. The viewer automatically unzips it. This is done through zlib.

When can you have it

I am sorry to say, but it will take some time. I hope to release it somewhere in October. Currently I spend five hours on the program every week. So that's not a lot, but it's all the time I have. I hope I can distribute the program through CTAN. Don't worry too much, I'll let you know when it is good enough to go public.

Will it cost you?

Yes it will, but not what you expect. No, I am not charging money here but you'll probably spend some time configuring the tool. As in good TEX tradition it comes with many options and they probably don't suit your personal needs. For emTEX users it should be fairly easy to set up since some settings are similar. There will be no registration fee, the program is absolutely free (and with no warranty). The first version will also include the full source code.

O yeah, for all you users who are getting tired of reconfiguring Ghostscript's fontmap file, be prepared. The viewer also uses a font map file and yes you must adapt it to your personal needs. You'll probably need to do this when you update the program (as in Ghostscript). I hate this too and I hope to find some solution for this some day. I will try to include all the fonts that are on the 4AllTEX cd, but you probably have to customize things. At least the complete Computer Modern family and the AMS fonts. Also some support for basic postscript fonts.

So that's it. I hope you're a bit enthousiastic about the viewer-to-be-born. Again I hope to have the features implemented as soon as possible but as it is with these freeware tools, there are no guarantees. I'll come back to it in the next MAPS.

Needless to say but there are many people on the background supporting and motivating me. Special thanks to: Hans Hagen, Taco Hoekwater and Erik Frambach.

6. mftoeps is a package from Jackowski to create eps files from metafont code.

Bijlage 12
Toolbox

Maarten Gelderman
Vrije Universiteit
Amsterdam
mgelderman@econ.vu.nl

abstract
Er lopen in de wereld zo veel TEX-gebruikers rond, dat het zelden zal voorkomen dat een normale gebruiker de eerste is die een probleem tegen komt. Meestal zal dit probleem ook al door iemand anders zijn opgelost en vaak staat de oplossing op de 4allTEX, TEXlive of een andere CD-ROM. Indien daar niets te vinden is staat er vast wel een oplossing op CTAN[1] of één of andere Internetsite. Het probleem zit hem niet in de beschikbaarheid, maar in het opsporen van de oplossing. Artikelen in MAPS zijn daarvoor natuurlijk een belangrijke informatiebron. Sommige oplossingen zijn echter zo eenvoudig dat er nooit een artikel aan gewijd zal worden. Over een draadstriptang valt nu eenmaal niet al te veel te vertellen. In *Toolbox* zal niet alleen de draadstriptang worden besproken, maar ook het gebruik van lucifers om het plastic van het einde van het snoer af te branden. Literatuurverwijzingen vormen deze keer de hoofdmoot van het verhaal.

Hoe werkt dat ook al weer?

Eén van de meest tijdsbesparende functies van LATEX is wel de mogelijkheid om op gemakkelijke wijze literatuurlijsten bij artikelen en boeken te maken, die 'altijd' in overeenstemming zijn met de laatste versie van het artikel. Dit scheelt een hoop controlewerk (haal ik geen literatuur aan die niet op mijn literatuurlijst staat en staan er op mijn literatuurlijst geen overbodige verwijzingen) en een hoop typewerk. De titelbeschrijvingen zelf, kunnen immers keer op keer opnieuw gebruikt worden. Laten we nog even kort herhalen hoe dit ook al weer werkt. In het LATEX-document doen we het volgende:

```
Een belangrijke verwijzing is \cite{Doll88}.
In een later artikel wordt letterlijk de
opmerking gemaakt: 'een letterlijke opmerking'
\cite[p.~455]{Doll94}. Een opmerking die meer
impliciet is terug te vinden in andere
artikelen over dit onderwerp
\cite{Alavi83,Benbasat83,Doll95}.
\nocite{Gelderman98hicss} % Mezelf haal ik
                          % helemaal niet aan,
                          % maar zo kom ik
                          % lekker wel op de
                          % literatuurlijst terecht

\bibliographystyle{plain}
\bibliography{data}
```

Dit voorbeeld laat 'alle' mogelijkheden van het \cite-commando op een rijtje zien. Met \cite{naam$_1$[,naam$_2$,...,naam$_n$]} Worden één of meer literatuurverwijzingen ingevoegd. Indien slechts één verwijzing wordt ingevoegd is het zinvol om bij voorbeeld een paginanummer mee te geven als optioneel argument. Het \bibliographystyle-commando geeft aan hoe de literatuurlijst eruit moet gaan zien (in dit geval is er gekozen voor de

1. CTAN is het Comprehensive TEX Archive Network, de dichtsbijzijnde CTAN-site is ftp://ftp.cs.ruu.nl/pub/tex-archive. Ook ftp://ftp.dante.de/pub/texarchive is goed bereikbaar en biedt bovendien zoekfunctionaliteit.

standaardlijst `plain`, een genummerde en gesorteerde lijst. Als verwijzing worden in de tekst standaard de nummers tussen blokhaken ('[1]') opgenomen. En het `\bibliography`-commando wordt gebruikt om aan te geven hoe het bestand met de benodigde gegevens heet. Het gebruik van meerdere bestanden is overigens ook mogelijk: net als bij het `\usepackage`-commando moeten de bestandsnamen gewoon door komma's gescheiden worden ingegeven.

Bij het de eerste keer runnen van LaTeX genereert LaTeX de foutmelding 'There were undefined references'. In de `.aux`-file die standaard door LaTeX wordt aangemaakt zijn de literatuurverwijzingen (de `\cite`-commando's) echter wel opgenomen. Wanneer nu BibTeX wordt gedraaid leest dit de `.aux`-file in om allereerst vast te stellen welke database gelezen moet worden (in dit geval `data`) en welke stijl moet worden gebruikt voor de literatuurlijst (in dit geval `plain`). Vervolgens worden de literatuurverwijzingen ingelezen uit de database en opgemaakt conform de opgegeven stijl. De zo ontstane literatuurlijst wordt opgeslagen in een `.bbl` bestand. LaTeX leest bij een volgende run dit bestand automatisch in, en na twee keer draaien (net als bij het gebruik van `\label` en `\ref` is twee keer runnen van LaTeX noodzakelijk) is de mededeling over de ongedefinieerde verwijzingen verdwenen en is op de plaats van het `\bibliography`-commando de literatuurlijst toegevoegd.

Voordat dit werkt, moeten we natuurlijk wel een database hebben. Lamport [1] beschrijft in appendix B hoe zo'n database eruit dient te zien. De database is een ASCII-file en de titels dienen hier in te staan op een wijze die vergelijkbaar is met het volgende voorbeeld:

```
@STRING{MISQ = "Management Information Systems Quarterly"}

@ARTICLE{Doll94,
  author = {Doll, William J. and Xia, Weidong and Torkzadeh, Gholamreza},
  title = {A confirmatory factor analysis of the end-user computing
          satisfaction instrument},
  year = 1994,
  journal = MISQ,
  pages = {453-461},
  month = dec
}
```

Op de eerste regel wordt een afkorting gedefinieerd.[2] De overige regels bevatten de beschrijving voor een artikel, waarin onder andere de betreffende afkorting en de voorgedefinieerde afkorting `dec` worden gebruikt. Daar de database een gewone ASCII-file is, kunnen we deze in dezelfde editor onderhouden, als waarin we de TeX-bestanden zelf ingeven. Er zijn echter weinig mensen die het intypen van dit soort data tot hun hobby rekenen en bovendien is de kans op fouten maar al te groot. Derhalve ga ik in de volgende paragraaf in op wat programma's die het bewerken van de database vergemakkelijken. Daarna kom ik toe aan mogelijkheden om de layout van de database en de literatuurlijst te veranderen.

Het bijhouden van de database

Diegenen die met emacs werken hebben weer eens geluk. Zij hebben vermoedelijk reeds lang geleden aucTeX geïnstalleerd. Wannneer een `.bib`-file wordt opgevraagd switcht emacs automatisch naar BibTeX-mode. Via de menu's of met behulp van een paar toetsaanslagen kunnen nu de namen van de velden van de database automatisch worden

2. In de praktijk is het handig de afkortingen in een apart bestand te bewaren. Ook artikelenbundels en dergelijke plaats ik altijd in een aparte database, voor gebruik van de cross-ref functionaliteit moet BibTeX deze items immers als laatste inlezen. Mijn `\bibliography`-commando is normaalgesproken dan ook `\bibliography`{afkortingen,gegevens,artikelenbundels}.

ingevoegd. Verder is het mogelijk items te sorteren en de syntax van de buffer te controleren (zijn alle haakjes wel netjes gesloten, staan de komma's wel op de juiste plaats). Zoals gebruikelijk bij GNU-software is bestandsomvang geen probleem. Alleen de syntax-highlighting geeft het op met wat grotere bestanden, maar de bestanden zelf hebben hier niet onder te leiden en alle andere functionaliteit blijft beschikbaar (alhoewel de syntax-controle met grotere bestanden wel erg traag wordt).

Wie niet met emacs werkt, of niet houdt van het bijhouden van ASCII-databases met behulp van een editor zal zijn heil moeten zoeken bij een meer specialistisch programma. Voor MS-DOS- en MS-Windows-gebruikers is de siamese tweeling BIBDB en BIBEDIT beschikbaar.[3] BIBDB is zowel verkrijgbaar voor MS-DOS als in een MS-Windows-versie en bevat een forse (62 pagina's) handleiding. Het pakket is uitgebreid configureerbaar, kan Tib, Refer en comma-delimited bestanden inlezen, kan sorteren, kan filteren en kan (zeer krachtig) zoeken. Het belangrijkste nadeel is dat het pakket een nogal stugge interface heeft, waarbij voor iedere handeling een nieuw venster wordt geopend. Problemen die eerdere versies hadden met het inlezen van grote bestanden, lijken met versie 2.0 verholpen. Mijn eigen literatuurdatabase (met iets meer dan 1000 items, in totaal ruim over de 10.000 regels groot) werd in ieder geval zonder problemen ingelezen en kon ook gewijzigd worden. BIBDBis beschikbaar op `http://pluto.mpi-hd.mpg.de/~doron/bibdb.html` en `http://www.tcisoft.com/bibdb.html`. Op het moment dat ik deze *Toolbox* schrijf is de CTAN versie niet up-to-date.

BIBEDIT bevat geen handleiding, en dat is niet nodig ook. De interface conformeert zich aan hetgeen in de MS-Windows-wereld gebruikelijk is en iedereen met een beetje computerervaring zou er in één keer mee moeten kunnen werken. De functionaliteit is beperkt gehouden. BIBEDIT is in principe alleen geschikt voor het bijhouden voor databases, maar dat is voor de meeste gebruikers meer dan genoeg. Vervelend is dat velden maximaal één regel lang kunnen zijn en het programma zich derhalve minder leent voor het maken van 'annotated bibliographies'. Ook worden Tabs niet goed ingelezen. Problematisch is echter dat het pakket een (onduidelijke) beperkte capaciteit heeft. Van mijn literatuurdatabase was (zonder waarschuwing) na inlezen, bijwerken en opslaan de laatste 100 KiloByte verdwenen. Desalniettemin is BIBEDIT voor de niet veeleisende gebruiker (250 entries werden in ieder geval zonder problemen verwerkt) door de superieure interface een beter pakket dan BIBDB. BIBEDIT is dankzij de auteur van BIBDB beschikbaar op `ftp://ftp.tcisoft.com/tcisoft/BibEdit/`.

Er zijn nog vele andere pakketten om databases bij te houden. De twee besproken tools dekken tezamen met aucTEX echter precies af wat een normale TEX-gebruiker nodig heeft. BIBEDIT is voor de niet veeleisende gebruiker, BIBDB voor wie met grotere bestanden wil werken, veel functionaliteit wil hebben en bereid is te leven met een stugge interface en aucTEX voor de emacs-verslaafde. Het enige wat eigenlijk ontbreekt is een grafische interface voor Unix- en MacIntosh-gebruikers. xbibtex schijnt een bruikbaar programma voor Unix-gebruikers te zijn, maar weigerde op mijn PC te compileren. Voor de MacIntosh bestaat een programma genaamd hyperbibtex. Beide laatstgenoemde programma's zijn op CTAN te vinden.

Naast de pure database-programma's is op CTAN nog ander bruikbaars te vinden. Met name bibclean en bibtool zijn de moeite van het vermelden waard.[4] Beide programma's helpen de gebruiker bij extraheren van een nieuw `.bib`-bestand uit een groter bestand met in het nieuwe bestand alleen de titels die in een opgegeven `.aux`-file worden genoemd (handig bij het inzenden van een artikel voor een tijdschrift). Verder is de fuctionaliteit gericht op het vinden van fouten in bestanden, controle op dubbele entries, het sorteren van bestanden etc. bibtool bevat een handleiding van 63 pagina's die in 1997 voor het laatst is

3. In de documentatie van BIBDB wordt de lezer die het pakket te moeilijk vindt aangeraden BIBEDIT te gebruiken, terwijl de auteur van BIBEDIT veeleisende gebruikers aanraadt over te stappen op BIBDB.
4. Naamgeving is bij dit soort programma's een probleem, op CTAN zijn meerdere bibcleans, een bibtool en een bibtools etc. te vinden. Zoek bij het downloaden dus even door.

Figure 1. De MS-DOS- en MS-Windows-versie van BIBDB.

bijgewerkt. bibclean is gedocumenteerd middels een TugBoat-artikel van de auteur (Nelson Beebe) en bevat tevens een uitgebreide man-page. Geen van beide tools heeft een probleem met mijn database, maar alleen bibclean attendeert me op het feit dat een aantal annote-velden te lang is. Onder Unix compileren beide pakketten zonder problemen, maar alleen bibclean heeft een aparte `ibmpc`-directory bij de broncode zitten. Vermoedelijk is het laatste pakket dus het geschiktst voor de MS-DOS/MS-Windows-gebruiker, een voorgecompileerde versie is echter van geen van beide pakketten beschikbaar.

Afwijkende formaten voor literatuurlijst en verwijzingen

Bij het gebruik van BIBTEX blijken drie vragen bij herhaling op te komen:

1. Ik wil een literatuurlijst per hoofdstuk
2. Ik wil dat mijn literatuurlijst er anders uitziet
3. Ik wil anders citeren, bij voorbeeld niet [1], maar Jansen (1986)

Voor de eerste wens is chapterbib beschikbaar. Dit is een stijlbestand, en wordt dus opgeroepen met `\usepackage{chapterbib}`. Als ieder hoofdstuk in een apart bestand staat en wordt opgeroepen met `\include` wordt automatisch bij ieder hoofdstuk een aparte literatuurlijst opgenomen. Alhoewel dit natuurlijk geen ideale oplossing is, is het in de meeste gevallen toereikend. De documentatie van deze tool is onderaan de betreffende style-file

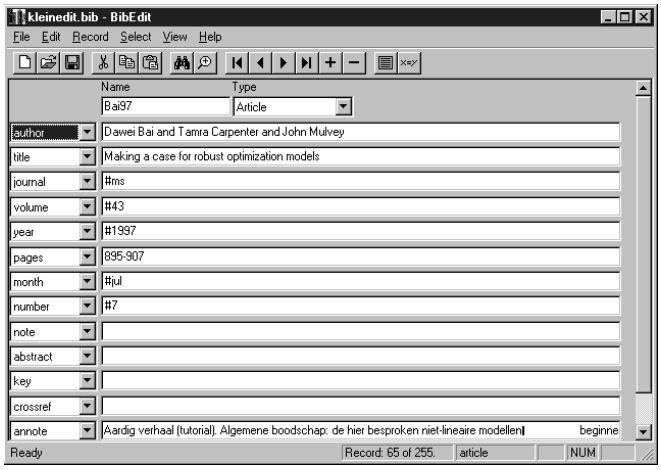


Figure 2. De MS-Windows-versie van BIBEDIT.

te vinden.

De tweede en derde vraag laten zich het beste gecombineerd beantwoorden. Wijzigingen van de wijze van verwijzen impliceren immers een ander soort literatuurlijst. In beide behoeften wordt voorzien door het package natbib en het plain-TEX-pakket makebst (beide tools zijn van de hand van Patrick Daly.).

Zoals hierboven reeds is aangegeven, dient in een LATEX-document waarin gewerkt wordt met literatuurverwijzingen die uit een database moeten worden gehaald het commando `\bibliographystyle` te worden gebruikt. De parameter van dit commando bevat de naam van een `.bst`-file die BIBTEX inleest om uiteindelijk de bibliografie op te maken. Het grote probleem met `.bst`-bestanden is dat ze zo goed als onleesbaar zijn en de gemiddelde TEX-gebruiker is dan ook niet in staat om deze bestanden zelf aan zijn wensen aan te passen. Op CTAN is wel een grote hoeveelheid bestanden beschikbaar, maar deze doen over het algemeen precies niet wat je wilt, en bovendien is het nogal een klus om het juiste bestand te vinden. Dit probleem is met de komst van makebst grotendeels verdwenen. Na het intypen van `tex makebst` stelt TEX een schier eindeloze reeks met vragen over hoe de literatuurlijst eruit moet zien (waaronder de taal die gebruikt moet worden). Vervolgens kan het `.bst`-bestand automatisch gegenereerd worden. Door als argument van `\bibliographystyle` geen `plain`, maar de naam van het nieuwe `.bst` bestand te gebruiken, wordt *na de volgende* BIBTEX*-run* de literatuurlijst opgemaakt conform de nieuwe eisen.

Het maatje van makebst is natbib. Dit is een gewone style-file, die (soms in combinatie met een door makebst aangemaakt `.bst`-bestand) vele nieuwe mogelijkheden biedt voor het opmaken van de literatuurverwijzingen zoals die in de tekst komen te staan. De functionaliteit van natbib is te uitgebreid om hier volledig te bespreken, wie het onderste uit de kan wil halen zal de handleiding die bij het package zit (20 pagina's) moeten lezen. Een korte bespreking moet hier volstaan.

Afhankelijk van de optie waarmee we natbib aanroepen, wordt het soort citaten aangepast. Als we natbib aanroepen met de `\usepackage[numbers]{natbib}` verandert er vrijwel niets, we houden gewoon de traditionele wijze van citeren. Met de extra opties `sort` en `sort&compress` wordt er bij meerdere verwijzingen echter zorg voor gedragen dat ze gesorteerd (en samengevoegd worden). [1,4,7,3,2] wordt met `sort` [1,2,3,4,7] en met `sort&compress` [1–4,7]. Met de optie `super` in plaats van `numbers` wordt er superscript gebruikt voor literatuurverwijzingen. `\usepackage[sort&compress,super]{natbib}` resulteert dan in de verwijzing.[1–4,7] Interpunctie etc. kunnen naar wens worden aangepast.

Indrukwekkender worden de veranderingen wanneer we de optie `authoryear` gebruiken. Het is dan wel noodzakelijk met een door makebst gegenereerde `.bst`-file voor auteur-jaar citatie te werken (en op de vraag of dit bestand geschikt moet zijn voor natbib 'ja' te antwoorden). De meest elementaire voorbeelden van commando's die dan mogelijk zijn kopieer ik direct uit de handleiding.

`\citet{jon90}`	⇒	Jones et al. (1990)
`\citet[chap.~2]{jon90}`	⇒	Jones et al. (1990, chap. 2)
`\citep{jon90}`	⇒	(Jones et al., 1990)
`\citep[chap.~2]{jon90}`	⇒	(Jones et al., 1990, chap. 2)
`\citep[see][]{jon90}`	⇒	(see Jones et al., 1990)
`\citep[see][chap.~2]{jon90}`	⇒	(see Jones et al., 1990, chap. 2)
`\citet*{jon90}`	⇒	Jones, Baker, and Williams (1990)
`\citep*{jon90}`	⇒	(Jones, Baker, and Williams, 1990)

Ook hier kan de interpunctie naar wens worden aangepast. Bovendien is het mogelijk automatisch auteursindexen te laten aanmaken.

Al die verwijzingen

Maar daarmee genoeg over literatuurverwijzingen. Nog slechts twee opmerkingen in deze *Toolbox* en wel over verwijzingen van een heel andere soort: URL's. In het vorige nummer van *Toolbox* heb ik eindeloos lopen emmeren om die erin te krijgen, met name die tildes zijn zo vervelend. Taco had echter de oplossing: gewoon het package url gebruiken en het commando `\url` komt ter beschikking. Het gebruik spreekt voor zich.

Mocht in dit artikel de URL eens niet beschikbaar zijn, dan gewoon zoeken op CTAN. Dante is daarvoor geschikter dan Utrecht, het biedt een eenvoudige zoekfunctie: inloggen op de bekende manier en vervolgens `quote site index` *`zoektekst`* ingeven. Met `quote site index url` moet bovenstaand package te vinden zijn.[5]

Volgende keer

Geen idee, waar ik het de volgende keer over ga hebben. Misschien wel iets over conversie van en naar andere tekstverwerkers. Mijn email-adres staat boven deze bijdrage: suggesties voor andere onderwerpen en te bespreken tools en trucs zijn welkom.

References

[1] Leslie Lamport. *LaTeX—A Document Preparation System—User's Guide and Reference Manual*. Addison-Wesley, Reading, MA, USA, 1985.

5. Er wordt natuurlijk gezocht naar alle bestanden die het woord 'url' bevatten. De bedoelde file bevindt zich, zoals verwacht ergens in de directory `/macros/latex`.

Bijlage 13

A LaTeX Tour, part 3: mfnfss, psnfss and babel

David Carlisle

1 Introduction

This third installment of my tour covers three more distributions that are supported via the standard LaTeX bug report mechanism described in Part 1.

The mfnfss distribution provides LaTeX support for some popular Metafont produced fonts, that do not otherwise have any LaTeX interface.

The psnfss distribution consists of LaTeX packages giving access to PostScript fonts.

The third distribution in this part of the tour is babel, which provides LaTeX with multi-lingual capabilities.

2 The Mfnfss Distribution

The mfnfss distribution is something of a 'collecting point' for files in the distribution that have not got anywhere else to go.

2.1 Font Packages

These packages provide LaTeX interfaces to some publicly available fonts. They do *not* provide the fonts themselves, which are available from the `fonts` tree in the standard CTAN archives.

pandora The 'Pandora' family of fonts designed by Nazneen N. Billawala is an alternative to the standard 'Computer Modern' fonts of Knuth. The family consists of a full range of text fonts, including sans-serif and slanted.

oldgerm The old German fonts designed by Yannis Haralambous. There are three styles of text font, Schwabacher, Fraktur and Gothic. (The terms 'Fraktur' and 'Gothic' tend to be used interchangeably by English speaking mathematicians such as the present author, but the fonts in this collection have clearly distinguishable styles.)

There is also a font of 'initials', highly ornate uppercase letters, suitable for use as the first letter of a section. If you wish to use this in 'drop caps' style you may also want to use one of the contributed packages available on CTAN such as drop, or dropping, that automate the setting of a suitable paragraph shape and inserting the initial letter at the correct size.[1]

2.2 T1 Encoded 'Concrete' Fonts

Note: The following two files require the old release 1.1 of the dc fonts. Walter Schmidt very recently (March 1997) released a test version of a set of 'Concrete' fonts based on the new ec font base. The LaTeX support for these new fonts is available from `macros/latex/contrib/supported/ccfonts`. Once this release is stable, the following files will probably be removed from the mfnfss distribution.

dccr.mf Metafont source file used by the output files from `dccrstd.tex` to generate Concrete Roman fonts in T1 encoding.

dccrstd.tex TeX file used in the generation of Concrete Roman fonts in T1 encoding. It will produce a number of `.mf` files corresponding to Concrete Roman fonts in different sizes. By modifying the table inside this file further Metafont driver files can be generated. The `.fd` files for the Concrete Roman fonts can be produced with `cmextra.ins` which is part of the LaTeX base distribution.

3 The Psnfss Distribution

With the release of LaTeX 2_ε, LaTeX gained inbuilt support for the use of alternative font families in documents, and in particular for the use of scalable font formats such as Type 1 (PostScript) or TrueType.

The collection of packages, coordinated by Sebastian Rahtz, known as psnfss offers convenient interfaces to most of the more common font sets.

Most of the files here relate to font files renamed to a consistent naming scheme, promoted and maintained by Karl Berry. This encodes the font vendor, and details of the font such as its weight, style and encoding into a compact name that usually fits in the eight letter filenames used by some common filesystems. More information about the font naming scheme can be found on CTAN in `info/fontname`. It should be noted however that the packages themselves, such as the times package, do *not* depend on any particular font naming convention. LaTeX isolates packages from the details of the external font files by the use of '`fd`' (Font Descriptor) files which map the LaTeX 'NFSS' model of fonts to the external font metric files.

1. The `fd` files provided here load the original `yinit` font. The CTAN archives also contain '`yinitas`', a modified version of this font.

In principle, there is no real need for packages to load text fonts into LaTeX. For example, once the font metrics and font descriptor files for Times Roman (which is ptm in the Karl Berry Naming Scheme) are installed, then one could in principle switch to Times Roman in a LaTeX document by simply specifying `\fontfamily{ptm}\selectfont`. Normally one would instead want to assign the new font to one of the 'default' LaTeX families, Roman, as used by `\rmfamily`, Sans Serif (`\sffamily`) and Typewriter or Monospace (`\ttfamily`).

The support for PostScript fonts is split into two. The CTAN `fonts/psfonts` area contains material that is mainly automatically generated from the Adobe font metric files that are distributed with all Type 1 fonts. This includes the font metrics themselves, the Font Descriptor files, the 'map' files used to make fonts known to the dvips driver, and some basic packages to declare single fonts to LaTeX. This is supplemented in `macros/latex/packages/psnfss` by the 'hand written' packages of the psnfss collection that load popular *combinations* of font families, or deal with mathematics.

This section refers at various points to PostScript or Type 1 fonts, but in fact the TeX support for these fonts applies equally well to True Type, or other scalable formats. As long as TeX has access to the font metrics, the font format does not matter (to TeX; it matters to the driver you use to print the DVI file).

3.1 Psfonts

The CTAN psfonts area primarily contains the font metric and LaTeX font descriptor files, organised by font vendor, as outlined below. The basic format of the file structure is the same for each font family, so only the top level directories are given here, except for the Adobe Times family, which is further expanded as an example.

Font Vendors The font subdirectories of `fonts/psfonts` are:

adobe Fonts sold by Adobe, or built into PostScript devices.
bh Fonts designed by Bigelow and Holmes, these are mainly sold through Y&Y.
bitstrea Bitstream fonts.
monotype Monotype fonts.
textures Textures Fonts for the Blue Sky Research Macintosh TeX implementation.
urw Fonts distributed by URW.
xadobe Adobe 'expert' font sets.
xmonotype Monotype 'expert' font sets.

Each of the vendor directories contains subdirectories corresponding to the font families supported by the psfonts distribution. (Using the tools provided one can generate TeX support files for most other text fonts, the selection here is really just a set of examples.)

The subdirectories of the `adobe` directory are:

agaramon Adobe's rendition of a Garamond serif Roman family. (Commercial.)
avantgar Avant Garde sans serif (built into most PostScript devices).
baskervi Baskerville, a commercially available serifed Roman family. (If you are reading this in *Baskerville* then it is similar to the text you see, which is Monotype Baskerville).
bembo Bembo, a commercially available serifed Roman family.
bookman Bookman (built into most PostScript devices).
centaur Centaur, a commercially available serifed Roman family.
courier Courier (built into all PostScript devices).
garamond Garamond 3. Another Garamond serif Roman family. (Commercial.)
gillsans Gill Sans, a commercially available sans serif family.
helvetic Helvetica (built into all PostScript devices).
nbaskerv ITC New Baskerville, another variant on the Baskerville theme. (Commercial.)
ncntrsbk New Century Schoolbook (built into most PostScript devices).
optima Optima, a commercially available sans serif family.
palatino Palatino serifed Roman family (built into most PostScript devices).
symbol Symbol (built into all PostScript devices).
times Times Roman (built into all PostScript devices).
univers Univers, a commercially available sans serif family.
utopia Utopia, a commercially available serifed Roman family.
zapfchan ITC Zapf Chancery. A script font built into most PostScript devices.
zapfding ITC Zapf Dingbats. A symbol font built into most PostScript devices.

All the directories corresponding to a font family look essentially the same, each with the following subdirectories.

dvips Contains the 'map' file for the dvips driver program. This file can be appended to `psfonts.map` or used via a configuration file to tell dvips where to find the specified fonts. A suitable configuration file is included in the directory.

Other drivers will need similar information, but perhaps in a different format.

tex This directory contains the font descriptor files which must be placed in the input path for LaTeX, so that LaTeX has available the information about the available fonts. For some font families this directory would also contain a LaTeX package that assigns the fonts to one of the standard LaTeX font families, such as `\sffamily`. Some packages, such as times, are not distributed here as they would clash with the packages distributed as part of psnfss, as described below.

tfm The font metrics, in 'tfm' format. These files contain all the information about letter sizes, ligatures, and kerning that TeX needs to typeset text.
There are several files, as each font in the original family is made available in several encodings, the two main ones being the 'Classic' TeX encoding used by Computer Modern. This is known as OT1 in LaTeX, and as '7t' in the Karl Berry font naming scheme used here. Similarly the files with names ending in '8t' relate to fonts encoded to the eight bit 'Cork' encoding, known as T1 in LaTeX.

vf The virtual fonts. Most (but not all) drivers handle the re-encoding of the original fonts to the encodings that TeX expects by means of the virtual font mechanism. Some special fonts, such as Zapf Dingbats are not re-encoded, and so do not have a `vf` directory.

There is one very important thing to note about the above list. *There are no fonts!* Almost all of the `fonts/psfonts` area of CTAN is concerned with providing mechanisms for using fonts that you have obtained *elsewhere*. The fonts may be built in to your printer, or may be purchased separately. There are a few freely available Type 1 fonts. In such cases there will be an additional directory, `type1`, which contains the font files (normally in 'pfb' format).

Standard PostScript Fonts

In addition to the above directories, the psfonts area contains two zip files. If you need the files and have not got unzip (or pkunzip or winzip or...) then you can get a copy of unzip from the CTAN support area.

lw35nfss This zip archive expands to the subset of the `psfonts/adobe` tree that corresponds to the 'Standard 35' PostScript fonts as used in Adobe Laserwriter printers. If you are only interested in using fonts built into your printer, and not in using downloaded fonts, then just get this file rather than the large collection of metrics in `psfonts/adobe`.

lw35pk This zip archive contains bitmap fonts for the 'Standard PostScript fonts' in the usual PK format understood by most dvi drivers. This enables documents using Type 1 fonts to be previewed with dvi previewers that can not use outline font formats. (For example xdvi or the emtex drivers).

Tools and Extra Packages

There are a few remaining directories in psfonts.

ts1 The LaTeX textcomp package and related utilities for accessing fonts in the 'text companion' encoding known as TS1 in LaTeX. These include the TC fonts that are distributed with the EC fonts, and suitably re-encoded fonts from the standard Type 1 font sets. This encoding contains many non alphabetic symbols that should match the current text font (rather than the math font). It includes currency symbols, superior digits, dagger signs, etc.

mathcomp A contributed package for using the text companion fonts in math mode.

tools The source for the scripts and utilities used for generating all these files.

3.2 Standard Psnfss Packages

By contrast to the packages and font descriptor files in the psfonts distribution, the psnfss distribution contains 'hand written' files. These are either used to set up popular *combinations* of the 'standard' fonts, or load alternative font sets for mathematics. Due to the nature of mathematics fonts, these latter packages are typically much more complicated internally than the one or two line packages that load text fonts. For the user, however, this complexity should not be apparent.

The first set of packages (all generated from the source file `psfonts.dtx`) load combinations of the Basic Adobe PostScript font set into LaTeX.

times As one might guess, this declares Times Roman as `\rmfamily`. For mainly historical reasons, this package also declares Helvetica as `\sffamily` and Courier as `\ttfamily`, so effectively ensuring that all text (but not mathematics) is set in the basic PostScript font set. This is a convenience for the user who wants to replace all the text fonts by references to the basic Adobe fonts. It is an advantage to do this if you want to produce device independent and small PostScript documents for distribution. The disadvantage is that Times Roman, Helvetica and Courier, despite being the 'standard PostScript combination' look particularly horrible if placed next to each other at the same nominal size, as done by this package. Helvetica has a much larger 'x-height' (the height of the lower case letters) than Times Roman, so if sans serif and Roman text are mixed

in-line, then the sans serif looks much too big. (This is not so much of a problem if the sans serif is only used for headings.) Courier is just too 'wide' when placed alongside Times Roman, which is a particularly compact font.
To partially compensate for these problems, the pslatex package (written by me, but currently distributed as a contributed package, not part of the core LaTeX distribution) is an alternative to the times package. It loads Helvetica scaled by 90% and loads Courier by way of a virtual font that condenses it by scaling the horizontal direction (only) by 85%. pslatex also contains a copy of the mathptm package (see below) so installs a Times-Italic based font set for use in mathematics.

palatino Declares Palatino as `\rmfamily`, and Helvetica and Courier as `\sffamily` and `\ttfamily`.

helvet Declares Helvetica as `\sffamily`. (Does not change the other families.)

avant Declares Avant Garde as `\sffamily`. (Does not change the other families.)

newcent Declares New Century Schoolbook as `\rmfamily`, Avant Garde as `\sffamily` and Courier as `\ttfamily`.

bookman Declares Bookman Roman as `\rmfamily`, Avant Garde as `\sffamily` and Courier as `\ttfamily`.

chancery Declares Zapf Chancery as `\rmfamily`.

The above packages only affect *text* fonts, not mathematics. `psfonts.dtx` contains one special package, written by Alan Jeffrey, which does affect the math setup.

mathptm This package uses a set of virtual files that use various built in or freely available fonts to make a set of fonts suitable for replacing the standard Computer Modern Math fonts. In the current release, bold fonts (and so the LaTeX `\boldmath` command) are not supported. The pslatex package referred to above contains an essentially verbatim copy of mathptm. One may use mathptm as an example of the coding needed to make virtual fonts for mathematics based on other text italic fonts. How successful this will be depends to a certain extent how visually compatible are the symbols that are gathered from the various 'real' fonts that are used by the virtual math fonts. There are often good reasons for making such fonts (the main one being that documents using freely available fonts may be more easily placed on the Web in PostScript form), however the result is never likely to be as good as using fonts that have symbols that are *designed* to be visually compatible. For mathematics use within TeX, that currently restricts use to Computer Modern, or the commercial MathTime or Lucida Bright font sets described below.

The `psfonts.dtx` source file contains one other package:

pifont This declares the Zapf Dingbats font which contains an assorted mixture of symbols, and also defines new user level commands to access these symbols. See the package documentation, or *The LaTeX Companion* for details.

3.3 Freely Available Type 1 Text Fonts

The next set of packages are contributed by Peter Dyballa. In fact these are just one-line packages loading the appropriate font. Most of the code is in the `fd` files which are generated from the same source file.

charter Defines `\rmfamily` to use Bitstream Charter.

nimbus Declares URW Nimbus Roman-Regular and URW Nimbus Sans-Regular as `\rmfamily` and `\sffamily`. These are essentially free clones of Times Roman and Helvetica.

utopia Defines `\rmfamily` to use Adobe Utopia-Regular.

3.4 Commercial Text Fonts

The following packages are generated from the source file `adobe.dtx`. They are a rather random selection from the large catalogue of fonts sold by Adobe.

garamond Garamond as `\rmfamily`, Optima as `\sffamily` and Courier as `\ttfamily`.

basker Baskerville as `\rmfamily`.

mtimes Monotype[2] Times as `\rmfamily`.

bembo Bembo as `\rmfamily`, Optima as `\sffamily` and the ever popular Courier as `\ttfamily`.

3.5 Adobe Lucida

The following two packages relate to the original Lucida font set, designed by Bigelow and Holmes and sold by Adobe. They are generated from the `alucida.dtx` source file.

lucid Declares Lucida Roman and Lucida Sans as the Roman and sans serif families, and Adobe Courier again as the monospaced font.

lucmath Lucida has a matching set of mathematics fonts suitable for TeX use. This package makes the required definitions to make these known to LaTeX.

2. Not sure why this is generated from *adobe* source file.

3.6 Lucida Bright

A newer and more extensive Lucida family, also designed by Bigelow and Holmes but in this case sold by Y&Y, is known as 'Lucida Bright' and 'Lucida New Math'. The LaTeX support described here was written by Sebastian Rahtz and myself.

lucidabr.dtx This package (replacing the earlier lucidbrb and lucidbry packages) changes the LaTeX defaults for both text and mathematics to use the Lucida Bright and Lucida New Math font collections. It has numerous options to control different aspects of the package and to control which of the fonts to use. (Lucida Bright contains several font families, including 'fax' and 'casual' etc, as well as variant forms of the math italic alphabet.)
The LaTeX package and the font descriptor files for the math fonts are generated from this source file. The font descriptor files for the Lucida text fonts in the standard LaTeX encodings are available from the psfonts area (in the bh) directory, after Bigelow and Holmes, the creators of these fonts.
The TeX support and font metrics are freely available, but the fonts themselves must be purchased separately.

lucidabr.ins LaTeX installation file for Lucida Bright using the standardised 'Karl Berry' font names.

lucidabr.yy Alternative installation file. Use this instead of `lucidabr.ins` if you plan to install the fonts with their original font names, as sold by Y&Y. (In this case you do *not* need the `fd` files from the psfonts area.)

lucidabr.txt Introduction and installation guide for this package.

3.7 MathTime

The MathTime fonts are produced by Michael Spivak 'TEXplorators'. They are sold by Y&Y. The LaTeX support was written by Frank Mittelbach and myself.

mathtime.dtx The mathtime package is mainly concerned with mathematics setup, although it selects Times, Helvetica and Courier as the text fonts if they have not already been set by another package. The MathTime mathematics fonts are specially designed to match Times Roman, but blend quite well with other text fonts that are of a similar weight. Computer Modern mathematics tends to look very 'light' if used with font families other than Computer Modern. The package has several options to control the font choices made.

mtfonts.fdd The source for the font descriptor files for MathTime mathematics fonts.

mathtime.ins Installation file. Note that this file may be edited in a couple of places depending on whether or not you have the extended 'MathTime Plus' font set which includes bold math support.

mathtime.txt Introduction and installation guide for this package.

3.8 Documentation and Other Files

readme.txt General introduction.

psnfss2e.tex User level documentation on the use of these packages.

test0.tex Testing accents and other encoding specific commands are working correctly using POSTSCRIPT fonts.

test1.tex Test document that uses most of the 'Standard 35' fonts.

pitest.tex Test of the pifont package.

mathtest.tex Test of the mathptm package.

makefile Unix 'make' utility to automate installation of the packages.

allpspk Unix script that makes a test document using a specified font family and then uses dvips and its associated scripts to generate 'pk' versions of the fonts.

makepk Unix script that calls allpspk on some common fonts.

3.9 Psnfssx

Recently the psnfss collection has aquired a close cousin, psnfssx, distributed as a contributed package from `macros/latex/contrib/supported/psnfssx`. This contains some lesser used or nonstandard packages, related to POSTSCRIPT support. Of particular interest might be the ly1 files (contributed by myself) in that directory which provide the LaTeX support for the 'texnansi' encoding promoted by Y&Y by way of an `LY1` option to the fontinst package.

This psnfssx collection also contains some obsolete versions of packages formerly in psnfss; this material is provided for historical interest only. Use at own risk!

4 The Babel Distribution

The babel package is distributed from `latex/packages/babel` and is supported via the LaTeX bug reporting address, but has origins predating the current LaTeX release. As well as supporting LaTeX it contains support for plain TeX (and formats such as AMSTeX or eplain that are based on plain). Primarily babel is the work of Johannes Braams, with contributions for specific language files by numerous people.

Babel consists of a 'kernel' that extends LaTeX with a mechanism for switching between specified languages. Part of this kernel (related to hyphenation) must be loaded when the LaTeX format is made to get the full benefit of hy-

phenation tables for multiple languages. For each language, or related group of languages, supported by babel there exists a language-specific code file. This will offer translations of the fixed text strings used in the standard LaTeX classes, such as 'Table of Contents', 'Figure', etc., and may also offer language-specific 'shorthands' that make typing common constructs easier (for example the german option provides the construct `"ff` to produce 'ff' that would hyphenate to 'ff-f' if it fell at the end of a line). The language file may also modify the typesetting to support the normal conventions of that language. For example the french option modifies the spacing around punctuation marks in text.

4.1 Babel Kernel

babel.sty The main interface to babel. The user specifies all languages to be used in a document as options to this package, the last option specified is the default language for the document. So for example

```
\usepackage[french,german]{babel}
```

would enable the use of French and German conventions within the document, with the default language being German.

hyphen.cfg The standard LaTeX interface to hyphenation. When the LaTeX format is being made, this file is input if it exists, to setup the required hyphenation patterns. In the base LaTeX distribution there is no such file, and so a default action is taken which loads the original TeX patterns for American English. The babel distribution provides this configuration file (generated from `babel.dtx`) which defines some core functionality, and then reads `language.dat` to specify which hyphenation files to load.

language.dat This file must be edited to specify which language hyphenation files to load, and the name of the external file which contains the hyphenation table for each such language (and optionally a second external file, typically containing hypenation exceptions). Note that hyphenation files *must* be specified here, and so loaded when the format is made. This is a restriction of the underlying TeX system. Documents using other languages not specified here may still be processed, and babel will translate any fixed text strings, but it will not be able to correctly hyphenate that language. A default hyphenation will be used (most likely English) which may or may not be suitable depending how far the language differs from English.

switch.def This file is also generated from the same `babel.dtx` source. If babel is used as a package but was not used when the format was made, then the core functionality normally provided by `hyphen.cfg` will not be present. The package will detect this, and so input this file to provide the necessary definitions.

4.2 Language-Specific Files

The implementation of the language-specific code for each language within babel is contained in files with extension '`.ldf`' (language definition files). These are not directly input by the user, but specified as options to the babel package. Normally the option name is the same as the file name, except where noted below. Some similar languages or dialects are supported by the same external file, and some options are available in more than one name; such aliases are noted in parentheses in the list below.

Most languages also have a file with extension `.sty`; however this is just offered for compatibility with older versions of Babel and of LaTeX, or for use with plain TeX based formats. In normal LaTeX usage only the `.ldf` file is used.

bahasa Support for the Bahasa language.
basque Support for the Basque language.[3]
breton Support for the Breton language.
catalan Support for the Catalan language.
croatian Support for the Croatian language.
czech Support for the Czech language.
danish Support for the Danish language.
dutch The dutch and afrikaans options.
english The american (USenglish) and british (UKenglish) options. The option english refers to either British or American English, depending on the local installation.
esperant The esperanto option.
estonian Support for the Estonian language.
finnish Support for the Finnish language.
frenchb Support for the French language (the corresponding options are french (frenchb) or francais. If the french option is used then `french.ldf` will be used (from the GUTenburg french package) if it is available.
galician Support for the Galician language.
germanb The austrian and german (germanb) options.
kannada Support for the Indian language, Kannada.[3]
irish Support for the Irish Gaelic language.
italian Support for the Italian language.
lsorbian The lowersorbian option.
magyar The magyar (hungarian) options.
norsk Support for the Norwegian languages with options norsk, nynorsk.
polish Support for the Polish language.
portuges The brazil (brazilian) and portuges (portuguese) options.
romanian Support for the Romanian language.

3. Not in the current release, planned for babel 3.7.

sanskrit Support for the Sanskrit language, transliterated to latin script.[3]
scottish Support for the Scottish Gaelic language.
slovak Support for the Slovakian language.
slovene Support for the Slovenian language.
spanish Support for the Spanish language.
swedish Support for the Swedish language.
turkish Support for the Turkish language.
usorbian The uppersorbian option.
welsh Support for the Welsh language

Babel version 3.6 sees the welcome (re)introduction of support for non-latin scripts. It is probably fair to say that this support is still more experimental than the support for latin scripts. One problem, not directly under babel 'control', is that the TeX encodings for Greek and Cyrillic (corresponding to T1 for European Latin scripts) have not yet been finalised or agreed. Currently babel uses two 'locally defined' encodings, LWN and LGR.

greek The greek option, which utilises the 'kd' Greek fonts.
russianb The russian option, which utilises the 'LH' fonts.
ukranian Support for the Ukranian language.[3]

Two separate packages are currently in preparation which will be distributed, together with suitable fonts and hypenation tables, from CTAN. These will extend babel with options for the Ethiopian and Ukrainian languages.

4.3 Compatibility Files

The distribution contains the following two source files which generate files which enable the use of babel with formats based on plain TeX (and also the old LaTeX 2.09 release).

bbcompat The source for compatibility mode files. Most languages are provided with a 'package' with extension `.sty`. This just inputs the corresponding language definition file and should never be needed using the normal LaTeX interface.
bbplain The source for the `plain.def` file allowing the use of babel with plain TeX.

4.4 Installation Script and Font Descriptor Files

babel.ins Unpacks the babel distribution from the documented source files
cyrillic.fdd Font descriptor files for Cyrillic fonts in 'LCY' encoding.
greek.fdd Font descriptor files for Greek fonts in 'LGR' encoding.

4.5 Documentation

ASCII Text Files

00readme.txt The distribution guide.
install.txt How to install Babel.
install.mac How to install Babel with OzTeX.
CyrillicFonts.txt Further notes on the Cyrillic installation.
GreekFonts.txt Further notes on the Greek installation.

TeX Documents

tb1202 The source of the original article that appeared in *TUGboat*, Volume 12 (1991), No. 2.
tb1401 The source of an update article that appeared in in *TUGboat*, Volume 14 (1993), No. 1.
tb1604 The source of an update article that never appeared in *TUGboat*, but was presented at EuroTeX 1995, Arnhem.

4.6 Example File

language.skeleton An example file that can be used to build new language definition files from scratch.

5 Coming Soon

Part 4 of this tour will describe the files of the amsfonts and amslatex distributions of packages produced by the American Mathematical society.

Bijlage 14

Conversie van anyTEX naar HTML met TEX4ht

Erik Frambach
Rijksuniversiteit Groningen
email: E.H.M.Frambach@eco.rug.nl

abstract

Gurari's TEX4ht-programmatuur maakt het mogelijk om in TEX geschreven teksten (plain TEX, LATEX, of wat dan ook) vrij eenvoudig te converteren naar HTML. Als voorbeeld nemen we de 4TEX-handleiding die in LATEX is geschreven.

keywords

conversie, html, www, TEX4ht

Introductie

Vrijwel iedereen zal inmiddels wel een idee hebben wat HTML is. Heel kort samengevat is HTML een standaard om teksten voor het World Wide Web in op te maken. Qua structuur lijkt het erg veel op (La)TEX zodat conversie niet al te moeilijk is.

In TEX schrijf je bv. '`{\it cursief}`', in HTML schrijft je '`<it>cursief</it>`'.

In LATEX schrijf je bv. '`\chapter{hoofdstuk}`', in HTML wordt dat '`<h1>hoofdstuk</h1>`'.[1] Ook in HTML kun je tekst over een willekeurig aantal regels verdelen net als in TEX, echter een nieuwe alinea moet expliciet aangegeven worden met '`<p>`'. Een lege regel heeft geen speciale betekenis zoals in TEX.

Uiteraard zijn er nog veel meer subtiele en minder subtiele verschillen maar die komen we vanzelf tegen in de rest van dit verhaal.

Principes

TEX4ht kan samenwerken met willekeurige TEX formats doordat hij zelf helemaal geen TEX-code interpreteert zoals bv. LATEX2html, maar vrijwel alles overlaat aan TEX zelf. De truuk hierbij is dat `tex4ht.sty` in de dvi-file '`\special`'s toevoegt die later door TEX4ht als besturing worden gebruikt voor de conversie naar HTML.

Vereenvoudigd komt het erop neer dat bv. aan het `\section`-commando '`\special{<h2>}`' en '`\special{</h2>}`' wordt toegevoegd. TEX4ht kan die later gemakkelijk uit de dvi-file vissen.

Uiteraard is het in werkelijkheid veel ingewikkelder omdat o.a. de inhoudsopgave ook goed moet komen. En natuurlijk wil je vanuit de inhoudsopgave meteen naar een sectie kunnen springen. Er komt dus heel wat boekhouding bij kijken.

Het grote voordeel van deze aanpak is dat nummering, referenties en (niet in de laatste plaats!) macro's door TEX al zijn verwerkt. TEX4ht kijkt uitsluitend naar wat TEX in de dvi-file stopt.

Noodzakelijke aanpasssingen

Om met TEX4ht te werken moet er om te beginnen een stukje code toegevoegd worden aan de tekst. In het geval van een LATEX-tekst gaat dat als volgt:

```
\documentclass{rapport1}

\input tex4ht.sty
\Preamble{htm,3,fonts}
\begin{document}
\EndPreamble

\end{document}
```

De style file `tex4ht.sty` wordt dus niet via `\usepackage` geladen. De reden daarvoor is dat de style file niet strikt LATEX is maar in principe met elk format kan samenwerken.

Het `\Preamble`-statement bepaalt dat de file extensie 'htm' wordt (ik had ook 'html' kunnen specificeren, maar MS-DOS-gebruikers hebben daar moeite mee). De '3' geeft aan dat ik niet één lange lap HTML wil, maar een splitsing tot op drie niveau's: 'chapter', 'section' en 'subsection'. Daardoor krijg ik dus een groter aantal HTML-bestanden die stuk voor stuk veel kleiner zijn en dus meer geschikt voor online publicatie. TEX4ht zorgt daarbij zelf voor de noodzakelijke structuur met verwijzingen naar de samenstellende delen.

De optie 'fonts' geeft aan dat ik wil dat TEX4ht zo veel mogelijk de gebruikte fonts (d.w.z. italic, bold, sansserif) in de HTML-versie respecteert.

1. Omgekeerd is conversie van HTML naar LATEX ook eenvoudig. HTML kan ook direct door TEX geïnterpreteerd worden, bv. met Carlisles LATEX-package 'typehtml'.

Met deze eenvoudige aanpassing kunnen we al meteen aan de slag, doordat `tex4ht.sty` al voorgeprogrammeerd is om met LATEX samen te werken.

TEX4ht draaien

Als we de TEX-file met de bovenbeschreven aanpassing compileren krijgen we een dvi-file die niet meer geschikt is om 'gewoon' te bekijken of afdrukken. Het is daarom verstandig om te beginnen met alles wat bij de tekst hoort te kopiëren naar een nieuwe directory en vandaaruit met TEX4ht te werken.

De nieuwe dvi-file kunnen we nu door TEX4ht halen. Die genereert nu in één keer de HTML-code, maar ook nog een paar extra bestanden. De belangrijkste daarvan is de 'ivd' file (het omgekeerde van 'dvi'). In dat bestand staan die stukken uit de dvi-file die niet door TEX4ht c.q. niet in puur HTML kunnen worden weergegeven. Dat zijn met name figuren, maar ook die geaccentueerde letters die in HTML niet gedefinieerd zijn. Die staan allemaal op afzonderlijke pagina's in de ivd-file.

Voor de interne administratie genereert TEX4ht ook bestanden met de extensie 'xre' (voor kruisverwijzingen) en 'otc' (voor de inhoudsopgave). Daar hoef je zelf verder niet naar om te kijken.

[next] [prev] [prev-tail] [tail] [up]

Chapter 3 Installing 4TeX

3.1 Introduction
3.2 Typography conventions
3.3 TeX and 4TeX
3.4 A Historical Note, or `What's in a Name?'
3.5 Principles of 4TeX installation from CD-rom
3.5.1 Installation from CD-rom
3.5.2 Installation on your hard disk
3.5.3 Fine-tuning 4TeX to your taste
3.5.4 Network setup
3.5.5 Directory set up
3.6 Starting 4TeX

[next] [prev] [prev-tail] [front] [up]

Een partiële inhoudsopgave

Die ivd-file kan vervolgens naar PostScript worden omgezet, bv. met DVIPS. Dat PostScript-bestand wordt op zijn beurt door Ghostscript verwerkt tot afzonderlijke PCX-plaatjes per pagina.

Nou ja plaatjes, het zijn echt hele pagina's, zij het op lage resolutie, bv. 110 dpi, want de uitvoer is bedoeld voor presentatie op computer-beeldschermen.

Die PCX-plaatjes moeten eerst van hun ongewenste witruimte worden ontdaan ('cropping'). Daarvoor kan het programma 'display' gebruikt worden. Dat programma kan meteen de conversie doen van PCX naar GIF, een graphics-formaat waar elke WWW-browser mee overweg kan.[2]

Dan volgt er nog een laatste slag waarbij de GIF-plaatjes 'transparant' worden gemaakt. Daardoor krijgen de plaatjes bij weergave door een WWW-browser toch de achtergrondkleur van de HTML-pagina. Het programma 'giftrans' zorgt daarvoor.

Uiteraard heeft TEX4ht zijn boekhouding netjes bijgehouden zodat hij weet welk GIF-plaatje op welke plek in een HTML-bestand moet worden gebruikt.

Overigens moet gezegd worden dat deze procedure niet echt eenvoudig is maar wel goed te automatiseren. In 4TEX versie 4 is de hele procedure met de druk op een knop in werking te stellen. De HTML-versie van de 4TEX-handleiding op de cdrom is direct door 4TEX geproduceerd zonder 'handwerk'.

Ook moet gezegd worden dat de procedure tijdrovend is. TEX4ht is zelf erg lang bezig met het verwerken van een dvi-file, maar ook het genereren van de plaatjes kan veel tijd kosten. Echter een HTML-versie genereren van een compleet boek doe je niet elke dag.

Verfijningen

Ofschoon de eerste resultaten lang niet slecht zijn kan er nog heel wat verbeterd worden.

In het geval van de 4TEX-handleiding heb ik daarom de volgende aanpassingen gedaan.

Na de `\Preamble` heb ik toegevoegd:

```
\Configure{centerline}%
  {\HCode{<CENTER>}}{\HCode{</CENTER>}}

\let\Xincludegraphics=%
  \includegraphics
\def\includegraphics[#1]#2{%
  \hbox{\Picture+[pict]{}%
  \Xincludegraphics[#1]{#2}%
  \EndPicture}}
```

Daarmee zorg ik ervoor dat `\centerline` ook in HTML wordt ondersteund. Verder breid ik `\includegraphics` uit zodat TEX4ht weet wat daarmee moet gebeuren: een plaatje ervan maken. Merk op dat ik met deze simpele definitie

2. Wanneer je het programma 'display' onder 4TEX onder Windows NT draait 'lekt' het geheugen. Daardoor had ik voor de conversie (56 plaatjes achter elkaar 'croppen') zo'n 150 MB geheugen nodig! Na verlaten van 4TEX is alles weer normaal. Wees gewaarschuwd.

mezelf verplicht tot het geven van opties in vierkante haken. Het kan mooier maar dat is slechts syntactische suiker.

Een volgende verfijning is het verwijderen van `\tableofcontents`. TEX4ht genereert die zelf al, en twee TOC's is te veel van het goede. Ook al begint het boek met een voorwoord, de inhoudsopgave komt altijd helemaal bovenaan. Het voorwoord komt daarom nu *wel* in de inhoudsopgave, wat ook logisch is.

Ook bibliografieën moeten anders verwerkt worden. In plaats van

```
\bibliographystyle{plain}
\bibliography{mybib}
```

moet je nu opnemen:

```
\input myfile.bbl
```

Uiteraard moet de bbl-file uit een eerdere gewone TEX- en bibTEX-run beschikbaar zijn.

Bibliography

[1] P.W. Abrahams, K.A. Hargreaves, and K. Berry. *TeX for the impatient*. Addison-Wesley, 1990.

[2] Adobe Systems Incorporated. *PostScript Language Tutorial and Cookbook*. Addison-Wesley, 1985.

[3] Adobe Systems Incorporated. *PostScript Language Reference Manual*. Addison-Wesley, 1990.

[4] S. von Bechtolsheim. *TeX in Practice: 1. Basics*. Springer-Verlag, 1993.

Een stukje bibliografie

Strikt genomen kan in HTML de verlaagde E uit het TEX-logo wel weergegeven worden, maar omdat exacte positionering (kerning) niet mogelijk is wordt het altijd lelijk. Dan maar liever gewoon 'TeX'. Voor het LATEX-logo geldt dat nog sterker. Dat is gelukkig ook precies wat TEX4ht doet. Voor de liefhebbers: het LATEX-logo zou in HTML ongeveer zo uit moeten zien (op 1 regel zonder spaties!):

```
L<SUP><FONT SIZE=-2>A</FONT></SUP>
T<SUB><FONT SIZE=+0>E</FONT></SUB>X
```

maar het hangt maar net van je browser en van het lettertype af of dit ergens op lijkt of niet.

Het LATEX-logo in html: mooi of niet?

Omdat we nu een interactief document maken is het aardig om email-adressen en verwijzingen naar WWW-pagina's meteen als 'link' weer te geven:

```
\def\email#1{%
  \HCode{<A HREF="mailto:#1">}%
  \texttt{#1}\HCode{</A>}}
\def\url#1{%
  \HCode{<A HREF="#1">}%
  \texttt{#1}\HCode{</A>}}
```

Daar is natuurlijk wat meer kennis van HTML voor nodig, maar het geeft aan hoe eenvoudig je zelf m.b.v. `\HCode` zelf HTML-code kan toevoegen. Een ander aardig voorbeeld is een andere definitie voor de toets-icoontjes die in de papieren handleiding worden gebruikt (bv. Esc). Van die icoontjes zou ik plaatjes kunnen maken, maar dat bleek ondoenlijk. TEX4ht beschouwt namelijk elke instantie van een plaatje als uniek en zou dus honderden plaatjes gegenereren terwijl er maar zo'n 30 unieke icoontjes zijn (dit geldt overigens ook voor bv. geaccentueerde letters). De volgende oplossing is daarom beter:

```
\def\toetsfont{\Large\sf}
\def\toets#1{%
  \HCode{<FONT COLOR="\#FF0000">}%
  {\toetsfont[#1]}%
  \HCode{</FONT>}}
```

In plaats van een zwart omkaderd teken krijg je nu de toets tussen vierkante haken, helemaal in het rood. Op het beeldscherm is dat heel effectief en natuurlijk veel sneller dan met plaatjes.

Een andere eigenaardigheid die verbeterd moest worden was de interpretatie van LATEX's '`\\`' commando. Dat bleek TEX4ht helemaal te negeren. Geen nood. Het HTML 'break'-commando kan zo toegevoegd worden:

```
\let\omslag=\\
\def\\{\omslag\HCode{<BR>}}
```

Merk op dat het commando's '`\\*`' nu niet meer werkt, maar die optie is voor de HTML-versie toch overbodig. Voor '`\\[xx]`' geldt dat in mindere mate omdat die wel eens misbruikt wordt om witruimte tussen 'alinea's' te maken. Die heb ik daarom vervangen door `\par`.

Een ander aspect dat juist opvallend gemakkelijk door TEX4ht wordt opgelost is tabellen.

Door in alle gevallen domweg standaard 'tabular'-omgevingen te gebruiken gaat alles vanzelf goed. TEX4ht genereert een HTML `<TABLE>` omgeving zodat die snel en betrouwbaar door een WWW-browser getoond kan worden. LATEX2html waagt zich daar niet aan en genereert een plaatje.

Verder kwam de conversie vooral neer op het verwijderen (voor zover nodig) van overbodige statements: `\clear(double)page`, `\parskip`, `\parindent`, `\textheight` en andere opmaakcommando's hebben geen zin meer.

	n^*	R^*	$EK^{[0,L]}(\pi^{R^*}(n^*))$
case 1	0	∞	0.8
case 2	1	11.4	1.7
case 3	2	0.6	5.0
case 4	4	1.2	6.4
case 5	6	1.7	9.8

Een willekeurige tabel

Met formules gaat TEX4ht heel zorgvuldig om. Voor zover mogelijk gebruikt hij HTML, en waar nodig laat hij TEX/DVIPS plaatjes genereren.

Wanneer we bv. $\sin x^2 \sum_1^\infty \delta x$ als volgt invoeren:

```
$\sin x^2 \sum_1^\infty \delta x$
```

dan genereert TEX4ht daarvoor de volgende HTML-code:

```
sin x<SUP >2</SUP>
<FONT FACE="SYMBOL">a</FONT>
<SUB >1</SUB>
<SUP ><FONT FACE="SYMBOL">Y</FONT>
</SUP><FONT FACE="SYMBOL">d</FONT>x
```

Keurig. Zetten we dezelfde formule in een equation-omgeving dan genereert TEX4ht een plaatje, waarin natuurlijk ook het formulenummer terugkomt.

Conclusie

De resultaten die met TEX4ht te bereiken zijn mogen gezien worden. Toch zijn er nog een paar punten die verbeterd zouden kunnen worden.

- Je hebt geen controle over de plaats (en vormgeving) van de inhoudsopgave.
- TEX4ht genereert zelf [up], [front], [tail], [prev-tail], [next] en [prev] links, zowel boven- als onderaan elke pagina. Onderaan de pagina komen ze echter soms direct achter de lopende tekst, dus zonder witruimte.
- De HTML-code is niet erg fraai. WWW-browsers hebben daar geen enkele moeite mee, maar als je zelf toch iets wilt aanpassen is het hinderlijk.
- Plaatjes worden niet hergebruikt ook al zijn ze volstrekt identiek.
- De fontdefinitie-bestanden die TEX4ht meelevert zijn goed bruikbaar maar geven soms toch onvoldoende informatie om de conversie goed te laten verlopen. Het beste kun je je helemaal beperken tot de Computer Modern familie of de 'mathptm' fonts.
- De uitvoer van MakeIndex wordt (nog) niet ondersteund.

Maar laten we niet vergeten dat dit programma nog maar heel kort bestaat. Bovendien heeft het ten opzichte van zijn meest direct concurrent LATEX2html ook duidelijke voordelen:

- Het systeem is LATEX-onafhankelijk: het kan samenwerken met elk TEX-systeem.
- Het systeem is niet Unix-afhankelijk. Dat is LATEX2html inmiddels ook niet meer zo erg, maar toch.
- Veel lastige zaken (macro's!) worden aan TEX overgelaten. Wie weet het nou beter dan TEX?
- Nummering van hoofdstukken, secties, formules, tabellen, figuren etc. blijft intact.
- Tabellen worden keurig in HTML vertaald.
- Input-encoding doet er niet toe, want TEX regelt dat.

De resultaten van de conversie van de 4TEX-handleiding staan op de 4allTEX cdroms (zie `\4texdoc\4texdoc.htm`) en (enigszins aangepast) op de NTG WWW-server: `http://www.ntg.nl/4allcd/4texdoc/4texdoc.htm`

Uiteraard staat het hele TEX4ht-pakket ook op de 4allTEX cdroms (zie disk2, `distrib\tex4ht`) en is het ingebouwd in 4TEX.

Via Gurari's WWW-pagina `http://www.cis.ohio-state.edu/~gurari/` is altijd de nieuwste versie van TEX4ht op te halen. Daar zijn ook verschillende mooie voorbeelden te zien van conversies. Kijk vooral eens naar `http://www.cis.ohio-state.edu/~gurari/TeX4ht/mn162.html` waarin wordt weergegeven hoe bekende public domain TEX- en LATEX-documenten naar HTML geconverteerd kunnen worden. Zeer leerzaam.

Bijlage 15

Visual TEX 5.10 for MS-Windows

Erik Frambach
Rijksuniversiteit Groningen
email: E.H.M.Frambach@eco.rug.nl

abstract

MicroPress's Visual TEX is a complete TEX implementation for MS-DOS and all flavors of MS-Windows. The Windows version is reviewed here. It supports several interesting features that go beyond standard TEX, such as outline fonts and conversion to HTML.

keywords

TEX implementation, Visual TEX, VTEX, MS-Windows, HTML

Introduction

MicroPress's Visual TEX 5.10 ('VTEX' for short) is a complete TEX implementation that includes TEX, Metafont, BibTEX and MakeIndex. A few more utilities are available that enable special features.

To quote the extensive online help file: "On the low end, basic VTEX package is a small no-frills implementation intended for smaller and older machines. Basic VTEX can be run on 512k machines without hard drive. Basic VTEX is the only version of VTEX that can be run on 8086- and 80286-class computers. Basic VTEX package is frozen at the version 3.5 and will not be further enhanced. This version of the software is not described in the manual.

VTEX386 is a 32-bit protected mode implementation of VTEX for DOS. While the VTEX386 interface is similar to the basic VTEX, it offers much greater capacity and speed. Furthermore, it does away with some limitations inherent in 16-bit programs.

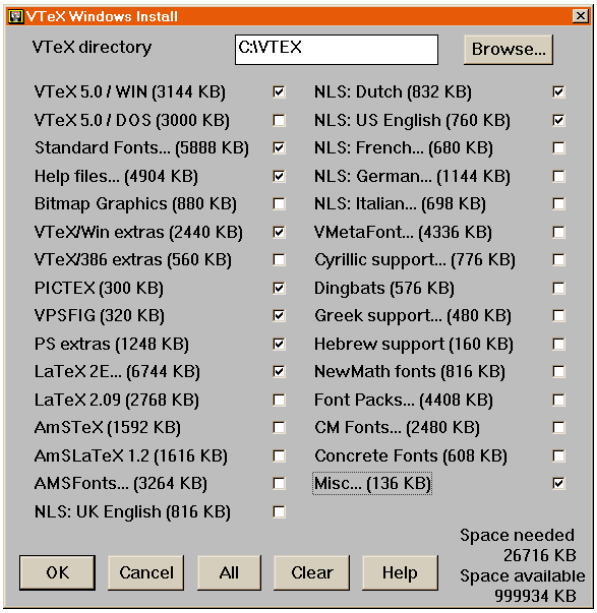
VTEX/Win is a 32-bit Windows application. VTEX/Win includes many capacities not present in DOS versions of the software; perhaps the most important enhancement being the Visual tools."

This article is a first impression of the software as it runs on MS-Windows (NT in this case).

Installation

VTEX is distributed on diskettes and on cd-rom. The cd-rom that I got was a 'gold' disk that contains no more than 31 megabytes of data. The whole system is packaged in compressed files of 360 kilobytes each, which brings back memories from the old days when everyone used 5.25 inch 'double density' floppy disks that couldn't hold more that 360 kb.

But don't let this packaging method fool you. The software is up-to-date and the Windows version comes with a full-fledged Windows setup program. This program allows you to select which parts of the software you want to install. A typical installation (whatever that is) will require around 30 megabytes, but even if you select all, you need no more than around 60 megabytes. As you select parts you want to install VTEX will tell you how much space you need. Nice.



Installing VTEX.

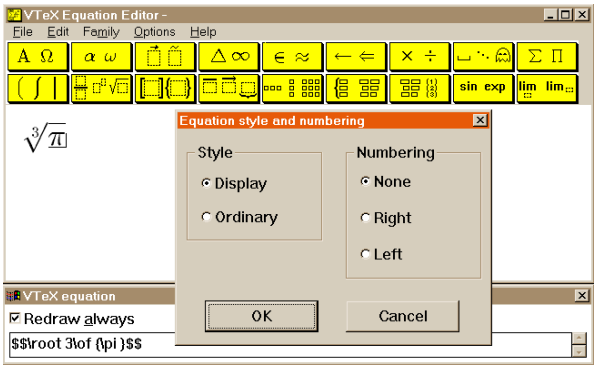
Some of the installation options will not be evident at first. Fortunately the online help information explains them in full detail. You should be able to make a reasonable selection if you read carefully, but if you are new to TEX you may well be dazzled by all the choices. In that case it's wise to start with a minimal set and install other parts later if you feel you need them. VTEX's installation program can be run again and again to add more and more.

Installation is always done completely on hard disk, so after installation the cd-rom is no longer required. It's not possible to run VTEX directly from cd-rom, like TEX-Live and 4allTEX. There is plenty of room left on the cd-rom, so I hope a future version will contain such a 'plug & play' system.

Special features

When you start VTEX you will find that it's a fully integrated TEX environment in which all standard TEX programs can be invoked by pressing a button or selecting them from a menu.

Of course an editor is part of the system. The editor has all the bells and whistles you can expect from a decent programmer's editor. Of course there is 'syntax high-lighting', and the editor can search for a matching brace if you put the cursor on a curly brace. Too bad that this trick doesn't work for '()[]<>'.

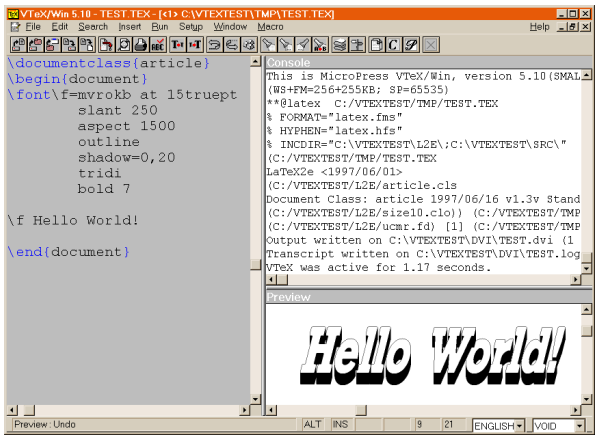


The equation editor.

The editor has a few more features that make it a fine TEX user's tool:

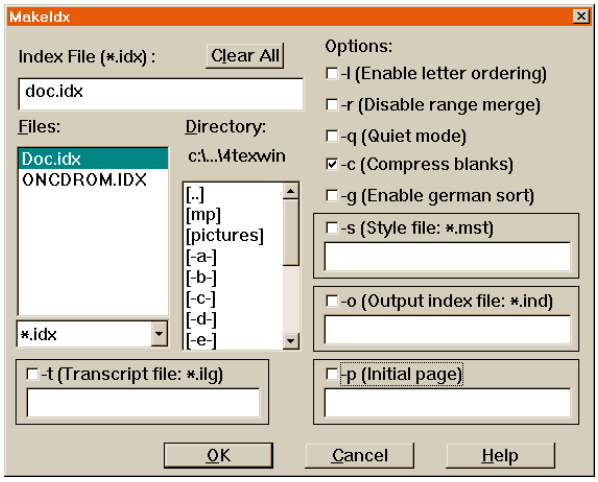
- You can start a WYSIWYG equation editor and paste the equation right into your text.
- You can include bitmap pictures for which automatically the correct TEX command is inserted (BMP, PCX, JPG, GIF, TIF, TGA and PNG are supported).
- You can include pictures in LATEX picture environment format. A simple graphic picture editor is included.
- You can consult the online help file on Plain TEX/LATEX if you're not certain about specific commands. Unfortunately the LATEX part is still based on LATEX 2.09, although VTEX itself fully supports LATEX 2_ε.
- You can convert a file written using a specific codepage to another on. If a file is in Latin2 format e.g. it's easy to make it PC850 or ISO8859 (Unicode). There are several other codepage conversions available.
- You can check the spelling of your text. Dictionaries for English (American, I presume), British, German, French, Italian and Dutch can be installed (the Dutch version is still in 'old' spelling, though).

In Visual TEX you can easily configure the window in such a way that you have full access to your text, compiler messages and the previewer. This 'topology' idea works rather well, especially if you are developing macros and need to test a lot. If you are 'only' writing a book, concentrating on content rather than programming, it may be of little use. Nevertheless, it's always convenient to have both a preview and your own text visible.



A possible screen topology that includes editor, TEX output messages and DVI preview.

Of course VTEX also supports BibTEX and MakeIndex, tools that are especially popular with LATEX users. The MakeIndex menu is a very user friendly system that makes it easy to do more advanced tricks.



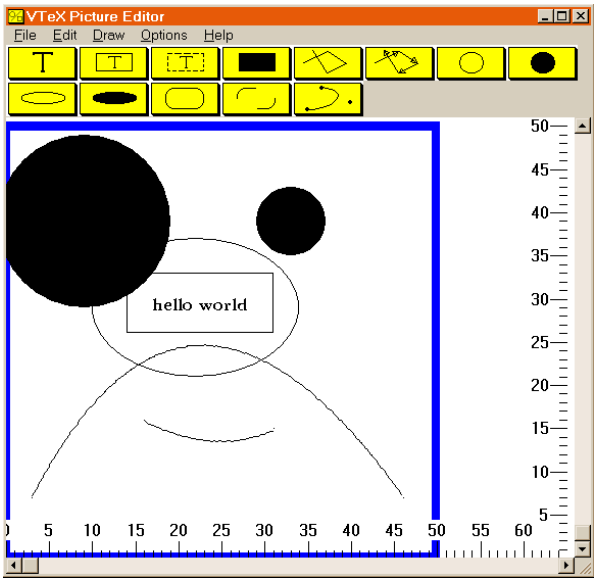
The MakeIndex menu.

In contrast the BibTEX menu is rather disappointing. In fact there is no menu. You can select a '.aux' file and then BibTEX is started. You can see BibTEX's screen messages just like the TEX compiler's messages and that't it. No BibTEX editor or anything else.

The compiler is not very fast. Compiling the 'LATEX 2_ε Kurzbeschreibung' (49 pages) took 25 seconds, whereas both emTEX and Web2c do the job in 10 seconds. A TEX

run can configured and automated in several ways:

- You can specify how many TEX runs you want, with or without BibTEX and MakeIndex runs;
- You can select ‘normal’, ‘scroll’, ‘non-stop’ or ‘batch’ mode;
- You can enable advanced debugging features. Depending on the level of detail you require the DVI file will grow a lot, but it’s very neat to be able to point to a place on the page in the previer, and then be guided to the correct spot in your text file to start editing...
- You can specify which hyphenation patterns you want to use *in runtime*. Unlike standard TEX VTEX does not depend on hyphenation patterns loaded during ‘iniTEX’. This approach obviously has advantages. For some reason Dutch hyphenation patterns are not included, so you if you need them you will have to get them from elsewhere.



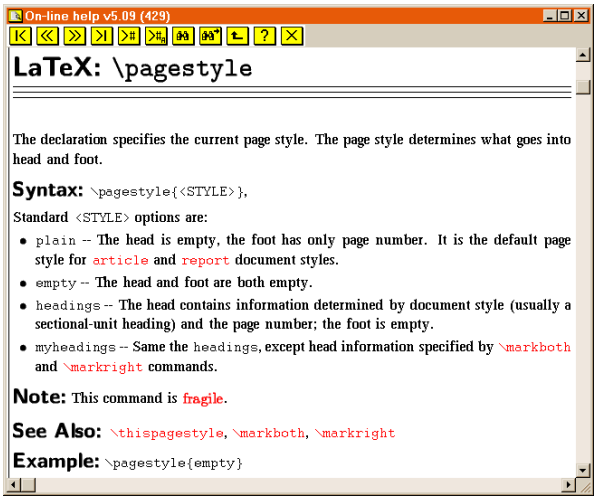
The picture editor.

Documentation

VTEX comes with about 300 paged of printed documentation. This documentation consists almost entirely of a Plain TEX course. The VTEX extensions to standard TEX are also covered in the manual. These extensions are mixed with other documentation, so they may be hard to find. E.g., did you know that all VTEX fonts support the Dutch ij ligature? That may not be much of an issue, but it would be a pity if you missed the following fancy extensions:

- Gray rules and patterns (`\gray`, `\grayl`, `\grayr`, `\graypattern`);
- Executing external programs from within TEX (`\command`, `\exec`, `\errno`);
- Extended font attributes (slant, aspect, bold, outline, shadow, fillpattern, reverse, smallcaps);
- Font rotation (`\sinecos`, `\sine`, `\cos`).

The printed documentation does not explain anything about the program itself. This is on purpose. Instead of quickly outdated printed documentation the program comes with truely extensive documentation in online help files.



Online help.

Conversion to HTML

A new feature in VTEX is direct conversion to HTML. They call it the ‘TeXpider’. Installing this feature is still a bit clumsy, but that will probably soon change. Documentation was not yet available in help files, but a LATEX documentation file and another sample should get you started.

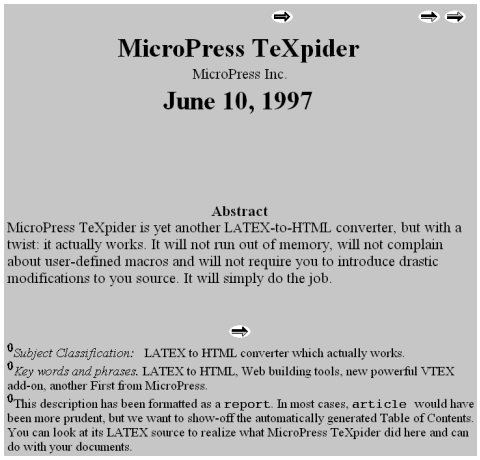
Programs like LATEX2html try to parse TEX code, which is very hard to do and by definition specific to a (subset of a) TEX dialect.

A different approach is taken by programs like TEX4ht that leave nearly all hard work to TEX. They enrich the DVI file with lots of `\specials` that will guide a DVI postprocessor in generating HTML.

VTEX takes yet another approach: it uses a modified TEX compiler that generates HTML directly instead of DVI. This compiler needs a specifically adapted LATEX format. Other formats are not (yet?) supported. An advantage of this approach is speed. Converting any LATEX file to HTML is only slightly slower than a normal compilation. No postprocessing is required.

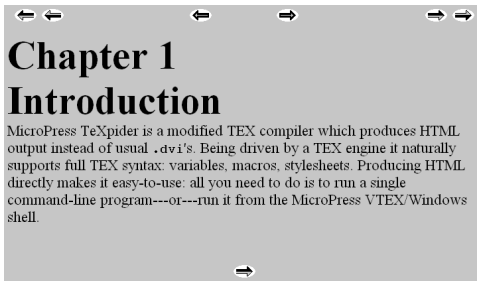
On the next page is a fragment of the TeXpider documentation and how it looks as an HTML page.

```
\maketitle
\begin{abstract}
MicroPress {\TeXspider} is yet another
\LaTeX-to-HTML converter, but with a twist:
it actually works. It will not run out of
memory, will not complain about user-defined
macros and will not require you to introduce
drastic modifications to you source. It will
simply do the job.
\end{abstract}
```



Example of HTML output rendered by Netscape Navigator.

It looks nice, doesn't it? But beware, if you want to change anything in the HTML version you're in for an unpleasant surprise. The HTML code that the TeXpider produces hardly shows anything of the original structure. No `<h1>` commands, not even a `<p>` can be found, whitespace is produced by multiple `<br>`s. The TeXpider produces pure 'visual' markup like in the next fragment:



Example of a section in HTML.

The actual HTML code looks like this:

```
<font face="times" size=4><b>
</div></font></b><font face="times"
size=7><b>Chapter 1 <br>
Introduction<br>
<a name="introduction"><a name="sec_intro">
</font></b><font face="times"
size=4>MicroPress TeXpider is a
modified TEX compiler which produces HTML
```

This is almost impossible to read, though WWW browsers will have no problem with that. However, if you want search engines to make any sense of your pages you have a problem. But then again maybe you don't if you only want to publish something quickly on the Internet. Fortunately, again, a few HTML editors that I tried understand the code, so editing is still possible, and you can avoid reading the code yourself.

A future version of the TeXpider will also support conversion to Winhelp files.

Conclusion

VTEX is a fine environment for using TEX on MS-Windows. It has a decent editor, excellent debugging features, some neat extensions to TEX, and its integration of editor, previewer and TEX console is very well done.

It's a pity that the manual is badly organized, and that the HTML conversion is not as powerful as it might be. VTEX's online help should also be updated to support LATEX 2ε: the current setup is rather confusing to new users.

All in all I think VTEX is a very powerful tool especially for TEX programmers, who will appreciate it's flexibilty and debugging features. Other TEX users may not need all that but they can simply ignore things they don't need. In this case the extra functions don't seem stand in the way of simple tasks. They never should, but in many other programs they do.

[Response from MicroPress]

While most of the concerns raised in Dr. Frambach's review have been resolved in 6.0 [for example, the online help has been totally reorganized and now covers LaTeX 2E; the TeXpider is now installed in an obvious fashion], we disagree with the reviewer's position about the importance of reediting the HTML which has been generated by TeXpider. The entire idea of this program has been for producing HTML that best approximates the original document and does not require reediting or testing with different browsers; if the code that has been generated needs correction, it should be done on the source level (very substantial part of the program comes as a macro package), or — when this is not sufficient — by requesting us to add the needed feature. High-level support for "meta" tags falls into the second category [it was indeed a serious omission on our part] and will be added by the time this review is published.

Bijlage 16
Windvi User's Manual

Fabrice Popineau
Fabrice.Popineau@supelec.fr

Editor's note
This article is an adaptation of the HTML file that comes with the current version (0.40) of windvi.

Why another Windows dvi viewer ?

Note that throughout this document, when I say 'Win32', this means Windows 9x and Windows NT.

There are many previewers for dvi files under Win32. The most popular is probably Dviwin by H. Sendoukas. However it lacks some important features:

- the ability to recursively search directories for font files,
- the ability to use .vf files or display PostScript fonts.

Unfortunately, the Dviwin sources were never put into the public domain; on the other side of the fence, Xdvi under Unix has these features, is widely used and its sources are available. Xdvi(k) uses the kpathsea library to search directories, already used in the Web2c-win32 port of TeX, so there was some interest in porting Xdvi(k) to Win32. As it turned out, this turned into far more than just a port, as X Windows is far from Win32. All the user interface and the graphical part has been rewritten.

For the future, when the base functionality of Xdvi(k) is available and stable under Win32, we can expect to add the interesting features of Dviwin to Windvi.

Features

The most important features of Windvi are as follows:

- monochrome or grey scale bitmaps (antialiasing) for fonts,
- easy navigation through the dvi file
  - page by page,
  - with different increments (by 5 or 10 pages at a time)
  - goto home, end, or any page within the document,
- different shrink factors to zoom page in and out,
- magnifying glass to show the page at the pixel level,
- use of .vf fonts
- display .pk .gf font files
- automatic generation of missing PK files even for PostScript fonts,
- tracking dvi file changes, and automatic reopening
- understanding Omega extended dvi files,
- drag-and-drop file from the Windows shell explorer,
- some color support (foreground and background),

Installation

The home of Windvi

Windvi is part of the Web2c package for win32. You will find the whole Web2C package on any CTAN archive, for example:

`ftp://ftp.tex.ac.uk/pub/tex/systems/win32/web2c`

Beta versions of Web2C for Win32 are available from:

`ftp://ftp.ese-metz.fr/pub/TeX/win32-beta`

If you want to retrieve only the windvi distribution, you should get:

`ftp://ftp.tex.ac.uk/pub/tex/systems/win32/web2c/windvixx.zip` for the standalone released version;

`ftp://ftp.ese-metz.fr/pub/TeX/win32-beta/windvixx.zip` for beta versions, xx being always as high as possible.

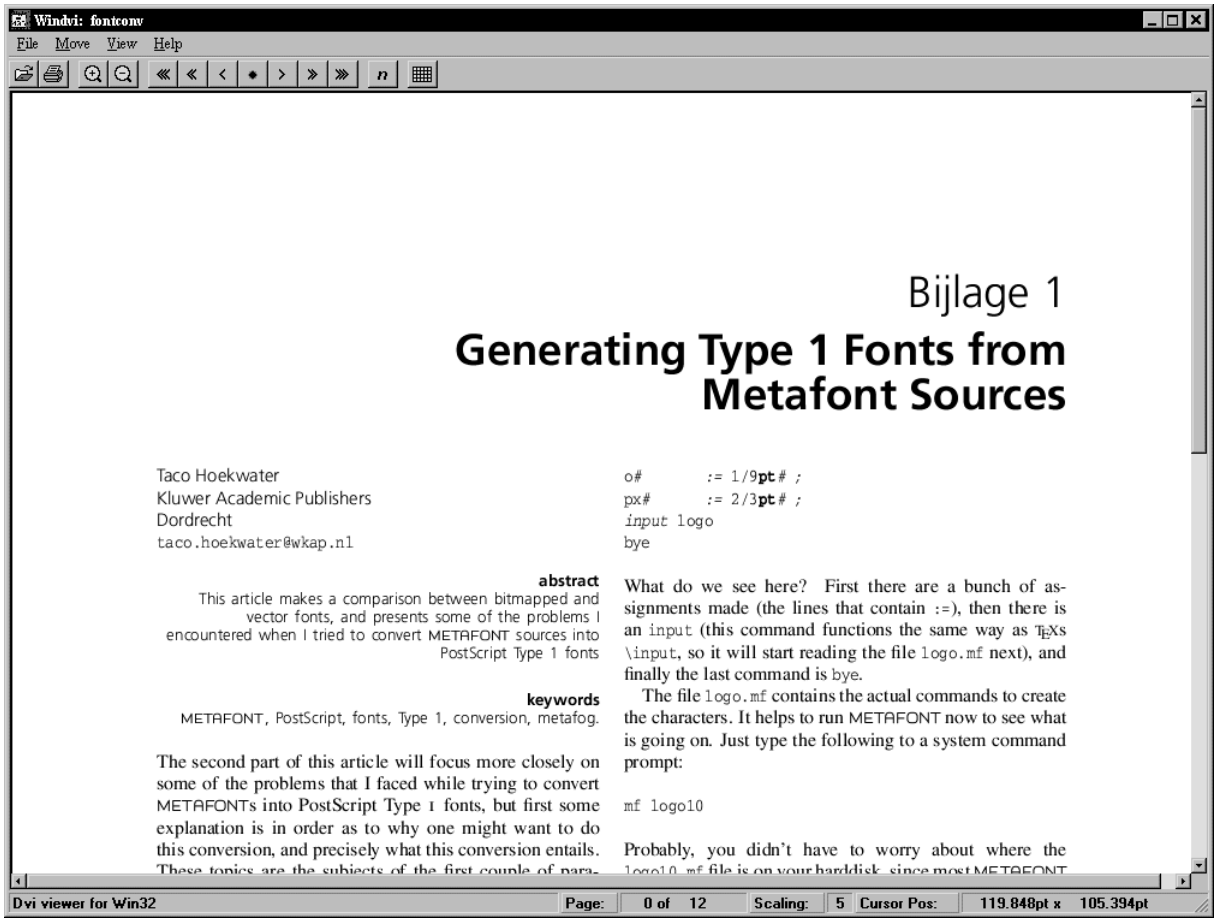
Announcements of beta version are made through the texk-win32 mailing-list. Subscriptions can be sent to `texk-win32-request@ese-metz.fr`. Send 'help' in the subject for more information.

Unpacking

Assuming you have retrieved one of the windvixx.zip files, you will have to unpack it at the ¡root¿ of some TDS conformant texmf tree like this one:

```
<root>/bin/win32
      /texmf/
                /dvips
                /tex
                     /latex
                /web2c
```

- owindvi.exe, windvi.exe, mktex*.exe and gsftopk.exe go to bin/win32
- render.ps (used by gsftopk) goes to texmf/dvips
- texmf.cnf and mktex.cnf go to texmf/web2c.



In order not to overwrite the texmf.cnf and mktex.cnf files of people who are already using Web2c for win32, those files are distributed as texmf.xam and mktex.xam. You must rename them if you don't already have .cnf files.

Next, add `<root>\bin\win32` to your PATH. This is done by modifying autoexec.bat under Windows 9x or the Control Panel/System under Windows NT.

Configuration

If you have respected the previous layout for the tree, ie the relative position of windvi.exe with respect to the texmf directory, you should not have anything more to configure than update your PATH environment variable.

If you want to use windvi.exe in another context, you may need to set the TEXMFMAIN and / or TEXMFCNF environment variable. TEXMFMAIN must point to the texmf directory and TEXMFCNF to the directory containing texmf.cnf.

Generation of PK files

The generation of PK files is under control of the Kpathsea library through the use of mktexpk.exe. You can choose the destination for generated files — the scheme is explained in detail in the Kpathsea documentation.

The main points are:

- any generated file will go in the same texmf tree as the one in which the font source has been found,
- if the source directory is not writable, the directory named by VARTEXFONTS will be used, so you had better set this variable to something meaningful in texmf.cnf
- MT_FEATURES can be set either in mktex.cnf or in your environment to control the naming of generated files: you can add 'dosnames', 'nomode', 'stripsupplier', 'striptypeface', 'fontmaps' and 'varfonts' to the features. Feel free to experiment with them by setting MT_FEATURES in the environment and checking with 'mktexnam cmr10' for the result you want.

Type1 fonts

In order to use gsftopk.exe to generate PK files for Type1 fonts, you will need to install Ghostscript. Ghostscript is used in the background for computing the bitmaps. Setting it up is quite easy; assuming that Ghostscript has been installed in `c:\gstools\gs5.10` and fonts in `c:\gstools\fonts`, add to your environment:

```
set GS_PATH=c:\gstools\gs5.10\gswin32c.exe
set GS_LIB=c:\gstools\gs5.10;c:\gstools\fonts
```

After that, Windvi should be able to generate PK files for your Type1 fonts, providing you have the corresponding vf and tfm files. It is however wise to try mktexnam.exe on some of your fonts ('mktexnam ptmr8r' for example) to check that the fonts will be put at the right place.

Quick startup

Create a shortcut to Windvi on your desktop:

- click right button on the desktop,
- New Shortcut,
- browse and find windvi.exe
- OK.

Next, explore your computer, drag and drop some dvi file onto the Windvi icon. If everything goes well, Windvi will open your dvi file and choose a suitable shrink factor for the page to be fully displayed.

Reference guide

Command line options

`+<page>` Specifies the first page to show. If + is given without a number, the last page is assumed; the first page is the default.

`-altfont <font>` Declares a default font to use when the font in the dvi file cannot be found. This is useful, for example, with PostScript fonts. Defaults to cmr10

`-background <color>` uses ¡color¿ as background color

`-bg <color>` same as -background

`-debug <bitmask>` If nonzero, prints additional information on standard output. The number is taken as a set of independent bits. The meaning of each bit follows. 1=bitmaps; 2=dvi translation; 4=pk reading; 8=batch operation; 16=events; 32=file opening; 64=PostScript communication; 128=Kpathsea stat(2) calls; 256=Kpathsea hash table lookups; 512=Kpathsea path definitions; 1024=Kpathsea path expansion; 2048=Kpathsea searches. To trace everything having to do with file searching and opening, use 4000. Some of these debugging options are actually provided by Kpathsea. See the 'Debugging' section in the Kpathsea manual.

`-density <density>` Determines the density used when shrinking bitmaps for fonts. A higher value produces a lighter font. The default value is 40. For monochrome displays; for color displays, use -gamma. Same as -S.

`-foreground <color>` uses ¡color¿ as foreground color

`-fg <color>` same as -foreground

`-gamma <gamma>` Controls the interpolation of colors in the greyscale anti-aliasing color palette. Default value is 1.0. For 0 ¡ gamma ¡ 1, the fonts will be lighter (more like the background), and for gamma ¿ 1, the fonts will be darker (more like the foreground). Negative values behave the same way, but use a slightly different algorithm. For color and grayscale displays; for monochrome, see -density.

`-hush` Causes Windvi to suppress all suppressible warnings.

`-hushchars` Causes Windvi to suppress warnings about references to characters which are not defined in the font.

`-hushchecksums` Causes Windvi to suppress warnings about checksum mismatches between the dvi file and the font file.

`-hushspecials` Causes Windvi to suppress warnings about special strings that it cannot process.

`-keep` Sets a flag to indicate that Windvi should not move to the home position when moving to a new page. See also the ' k ' keystroke.

`-margins <dimen>` Specifies the size of both the top margin and side margin. This should be a decimal number optionally followed by " cm ", e.g. , 1.5 or 3cm , giving a measurement in inches or centimeters. It determines the "home" position of the page within the window as follows. If the entire page fits in the window, then the margin settings are ignored. If, even after removing the margins from the left, right, top, and bottom, the page still cannot fit in the window, then the page is put in the window such that the top and left margins are hidden, and presumably the upper left-hand corner of the text on the page will be in the upper left-hand corner of the window. Otherwise, the text is centered in the window. See also -sidemargin, -topmargin , and the keystroke ' M .'

`-mfmode <mode-def>` Specifies a mode-def string, which can be used in searching for fonts. It is also passed to Metafont during automatic creation of fonts.

`-mgs <size>` Same as -mgs1 .

`-mgs[n <size>]` Specifies the size of the window to be used for the "magnifying glass" for Button n . The size may be given as an integer (indicating that the

magnifying glass is to be square), or it may be given in the form width x height. Defaults are 200x150, 400x250, 700x500, 1000x800, and 1200x1200.

`-nogrey` Turns off the use of greyscale anti-aliasing when printing shrunken bitmaps.

`-nomakepk` Turns off automatic generation of font files that cannot be found by other means.

`-offsets` Specifies the size of both the horizontal and vertical offsets of the output on the page. This should be a decimal number optionally followed by " cm ", e.g. , 1.5 or 3cm , giving a measurement in inches or centimeters. By decree of the Stanford TeX Project, the default TeX page origin is always 1 inch over and down from the top-left page corner, even when non-American paper sizes are used. Therefore, the default offsets are 1.0 inch. See also -xoffset and -yoffset .

`-p <dpi>` Defines the size of the fonts to use, in pixels per inch. The default value is 600.

`-qpaper <papertype>` Specifies the size of the printed page. This may be of the form widthheight (or widthheightcm), where width is the width in inches (or cm) and height is the height in inches (or cm), respectively. There are also synonyms which may be used: us (8.5x11), usr (11x8.5), legal (8.5x14), foolscap (13.5x17), as well as the ISO sizes a1 - a7 , b1 - b7 , c1 - c7 , a1r - a7r (a1 - a7rotated), etc. The default size is 21 x 29.7 cm.

`-rv` Causes the page to be displayed with white characters on a black background, instead of vice versa.

`-s <shrinkfactor>` Defines the initial shrink factor. The default value is to choose en appropriate factor.

`-S <density>` Same as -density, q.v.

`-sidemargin <dimen>` Specifies the side margin (see -margins).

`-topmargin <dimen>` Specifies the top and bottom margins (see -margins).

`-xoffset <dimen>` Specifies the size of the horizontal offset of the output on the page. See -offsets .

`-yoffset <dimen>` Specifies the size of the vertical offset of the output on the page. See -offsets .

Shortcut keys

`Home, '~'` goto the upper left corner of the page. If margins are active, use them.

`Next, 'n', Enter` goto next page.

`Prior, 'b', Backspace` goto previous page.

`Ctrl-Home, Ctrl-End` goto first (resp. last) page.

`Numpad +, Numpad -` zoom in (resp. out).

`Arrow keys, 'l', 'r', 'u', 'd'` move in the corresponding direction (left, right, up, down).

`'k'` Normally when Windvi switches pages, it moves to the home position as well. The ' k ' keystroke toggles a 'keep-position' flag which, when set, will keep the same position when moving between pages.

`'M'` set margins at the cursor.

`'t'` change tick units (cursor position).

FAQ

1. Windvi opens and closes immediately. You should check your installation:

   - did you rename the .xam files into .cnf files ?
   - have you .cnf files ?
   - what mktexnam cmr10 does report ?
   - in case of trouble, do the following:

     ```
     set KPATHSEA_DEBUG_OUTPUT=err.log
     mkktexnam --debug=1536 cmr10
     ```

     and send the err.log file to mailto:Fabrice.Popineau@supelec.fr

2. Windvi is stuck with the hour glass cursor, displaying some font name in the status bar. Currently, when kpathsea is generating fonts, Windvi is blocked. You can't see any progress status. This is because kpathsea-based programs are inherently console mode programs and Windvi is a GUI program. If it takes too long time and the status bar doesn't change, there is the chance of an improper installation. Check with the previous question. In this case, you will need to kill Windvi by hand, and any process named mf.exe or mktexpk.exe too.

Known bugs and TODO list

- Windvi is uninterruptible during font loading;
- The option –version makes Windvi abort;
- We need to add the preview of ps inclusions by using Ghostscript;
- We need to add more support for 'specials';
- Windvi is resource consuming;
- There are probably some other bugs left.

Color naming

You can use 'rgb:/rr/gg/bb/' where rr, gg and bb are the hexadecimal (00-FF) intensities of red, green and blue component, or any of the predefined symbolic names.[1]

1. editor's note: the HTML file lists alle of the defined names at this point in the text.

Bijlage 17

PPCHTEX

a macropackage for typesetting chemical structure formulas with TEX — release 2

Ton Otten
PRAGMA ADE
Ridderstraat 27
8061GH Hasselt NL
ntg-ppchtex@ntg.nl

Keywords
PPCHTEX, chemical formulas, graphics

abstract
A few years ago PPCHTEX was introduced. This generic macro package can be used to typeset chemical structure formulas. Currently the NTG supports a dedicated mailing list, to which quite some users subscribed.
Although PPCHTEX is used all over the world, the main body of users lives in Germany. Thanks to the input of Tobias Burnus and Dirk Kuypers the second release of PPCHTEX about doubles the functionality of the first and also supports PSTricks. Currently Tobias is coordinating the wish–list for the third release and Tobias and Dirk are collecting predefined structure formulas for a future database. We're also considering a METAPOST backend.
Here we present the long awaited for english update of the manual. The dutch and german manuals were already available for about a year and can be uploaded from the NTG server.

Before we go to the the manual, we present the content of `readme.en`. This file shows how the current distribution is packaged.

PPCHTEX

This macropackage can be used to typeset chemical structure formulas. PPCHTEX is applicable withing PLAIN TEX, CONTEXT, LATEX etc. The next files are part of this distribution

ppchtex.tex	the typesetting macros
ppchtex.noc	the interface to CONTEXT
m-ch-nl.tex, m-ch-nl.sty	dutch version
m-ch-de.tex, m-ch-de.sty	german version
m-ch-en.tex, m-ch-en.sty	english version

For historic reasons we still got:

m-chemie.sty	dutch version
m-chemic.sty	english version

Within CONTEXT one needs:

m-chemie.tex	automatic interface

The package PPCHTEX falls back on one of the system modules of CONTEXT.

syst-gen.tex	general interface macros

PPCHTEX

a macropackage for typesetting
chemical structure formulas

J. Hagen & A.F. Otten
Pragma ADE, Hasselt NL
October 2001

Contents

Introduction

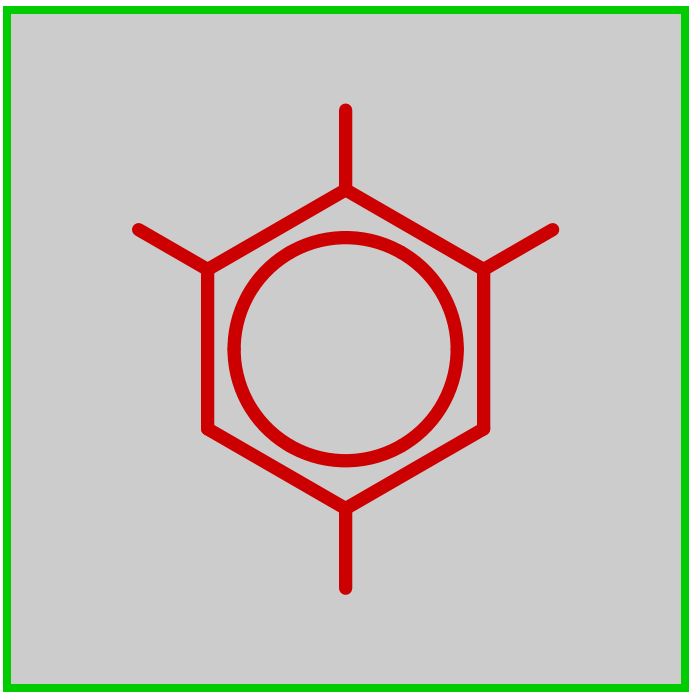
PPCHTEX is a set of coherent macros that can be used to typeset chemical structure formulas. The macros fall back on PICTEX, a public domain drawing package written by Michael Wichura. Although originally written on top of PICTEX, the second release can cooperate with PSTRICKS of Timothy Van Zandt, that is, with some limitations. Personally we still prefer the quality of the PICTEX output.

The macros can be used from within several macro packages and fall back on a few generic CONTEXT modules. The macros are written in a way that permits relatively easy upward compatible extensions. The interface is conform the CONTEXT interface.

PPCHTEX was originally meant for typesetting chemical structure formulas like sixrings. At the moment there is also support for reaction mechanisms. Formulas can be typeset at different sizes. Common elements or frequently used formula components can be reused.

Flexibility, simplicity and quality have been prefered over speed. We don't use the PICTEX option to save pictures, because timing showed that the gain was minimal.

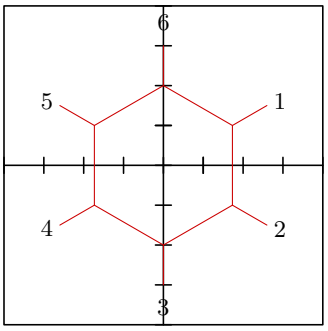
The first version of PPCHTEX was ready for use in 1995. This manual describes the second major release. Thanks to the many suggestions of Tobias Burnus, Dirk Kuypers and Ton Otten, the functionality was extended considerably.



Part 1
Explanation

1 Structures

The number of commands that is used to typeset chemical structure formulas is, apart from some bell and whistle commands, limited to four.[1] In the following example all of these commands are used.

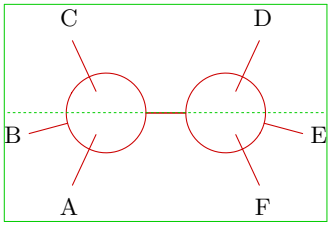


Example 1.1

```
\setupchemical[axis=on,frame=on]

\startchemical
  \chemical[SIX,B,R,RZ][1,2,3,4,5,6]
\stopchemical
```

With `\setupchemical` we can influence the makeup of the formulas. These setups influence all the following formulas, unless they are superceded by local setup variables.[2]

The set up variables can be defined right after `\startchemical`. In that case the set up is only applied to one structure formula.



Example 1.2

```
\startchemical[frame=on,width=fit,height=fit]
  \chemical[CARBON,CB1][A,B,C,D,E,F]
\stopchemical
```

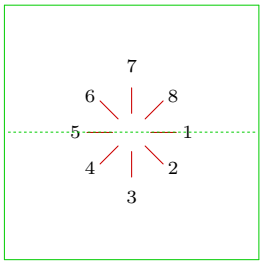
Both examples show that `\chemical` is the essential command. This command that may be used more than once within a `\start`–`\stop`–pair, is accompanied with two arguments. These arguments are written between `[ ]`. The first argument is used for defining the chemical bonds, the second argument for the atoms and molecules that make up the structure.

Text is typeset in mathematical mode, this means that you may type anything that normally is allowed between `$ $`.

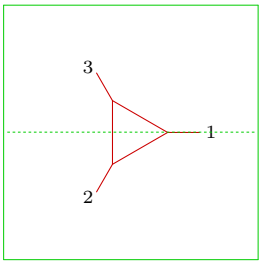
[1] The concept structure in this manual only refers to the chemical structure. It is not related to the document structure.

[2] One can of course limit the scope of the variables by using `{ }` and/or the grouping macros `..group`.

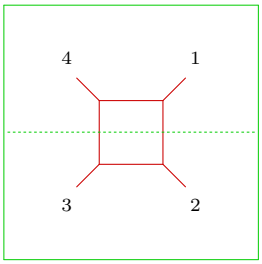
We will explain the first example in more detail. The key `SIX` means that we want to draw a sixring. In analogy we could type `ONE`, `THREE`, `FOUR` and `FIVE`, `EIGHT`, `CARBON`, `NEWMAN`, `CHAIR`, some alternatives on these keys and some symbols.



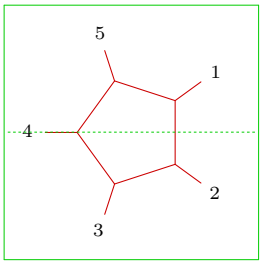
One



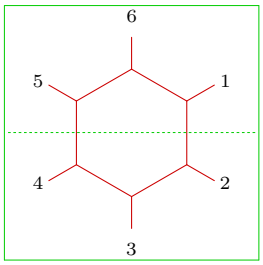
Three



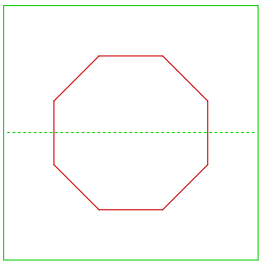
Four



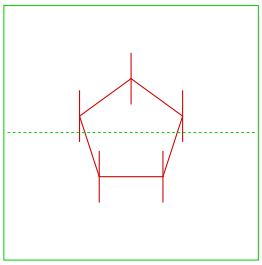
Five



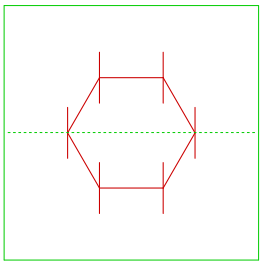
Six



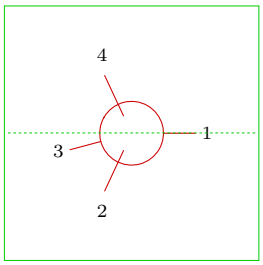
Eight



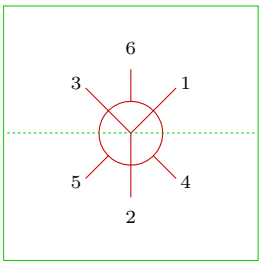
Five Front



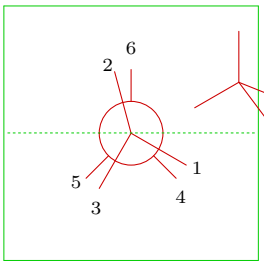
Six Front



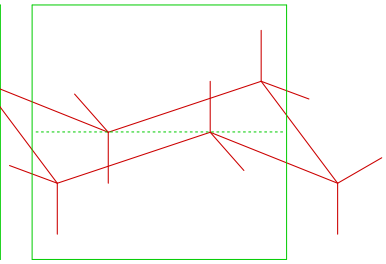
Carbon



Newman Stacker

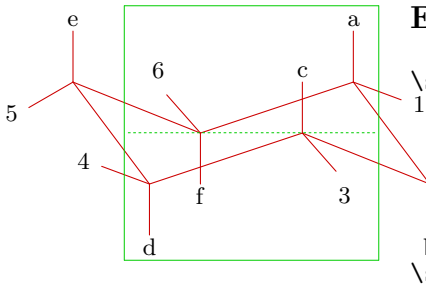


Newman Eclipse



Chair

The dimensions of `CHAIR` are somewhat different from the others. This structure is also different in other means. Rotation for example is not possible.



Example 1.3

```
\startchemical[frame=on,scale=small]
 \chemical
   [CHAIR,B,+R,-R,
    +RZ1,+RZ2,+RZ3,+RZ4,+RZ5,+RZ6,
    -RZ1,-RZ2,-RZ3,-RZ4,-RZ5,-RZ6]
   [a,b,c,d,e,f,1,2,3,4,5,6]
\stopchemical
```

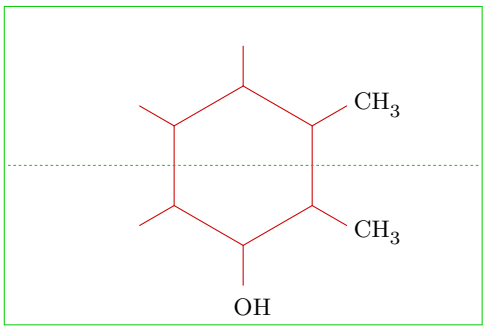
Within a structure the chemical bonds between the C–atoms are defined in the same way. In this example we use `B` and `R`. Bonds within a structure are numbered and can be defined by:

```
\chemical[SIX,B1,B2,B3,B4,B5,B6]
\chemical[SIX,B135]
\chemical[SIX,B1..5]
```

These keys draw parts of a sixring. With `R` and `RZ` we place substituents on the ring. The key `R` draws the bond from a ring corner to the substituent ($\angle$ 120°). The corner is also identified with a number.

```
\chemical[SIX,B1..6,R1..6]
```

The defininition above draws the six bonds in the sixring and the bonds to the substituents. The substituents are identified by the key `RZ`. Again numbers are used to mark the position. The substituents themselves are defined as text in the second argument.



Example 1.4

```
\startchemical[frame=on,width=6000]
  \chemical[SIX,B1..6,R1..6,RZ1..3][CH_3,CH_3,OH]
\stopchemical
```

When the second argument is left out no text (substituents) are placed on the ring and the key `RZ1..3` has no effect.

2 Bonds

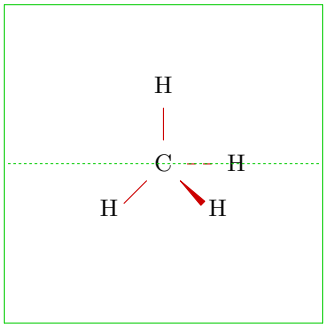
This chapter gives an overview of the bonds you can use in structures. From the examples throughout this manual the use of the different keys will become more meaningful.

Bonds always have two alternatives: a long and a short version. The shortened bonds leave room to place atoms within a structure. A number of bonds can be shortened on both sides left (`-`) or right (`+`).

`B`	Bond	`SB`	Single Bond
`BB`	Bold Bond	`-SB`	Left Single Bond
`HB`	Hydrogen Bond	`+SB`	Right Single Bond

Table 2.1 Single bonds.

The example below shows a number of bonds combined within one structure:



Example 2.1

```
\startchemical[frame=on]
  \chemical[ONE,SD1,SB4,BB2,SB7,Z01247][C,H,H,H,H]
\stopchemical
```

A bond can be followed by one or more numbers or a range, for example: `B1`, `B135` and `B1..5`. When you want to draw all bonds you can type `B`.

Within a ring structure you can define extra bonds between atoms, for example a double or triple bond.

`EB`	Extra Bond	`DB`	Double Bond
		`TB`	Triple Bond

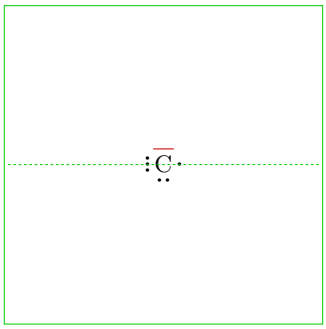
Table 2.2 Multiple bonds.

Free electrons and electron pairs can be defined in different ways. The accompanying keywords start with an `E`.

The example below shows a carbon atom with 8 outer electrons arranged in a chemically very peculiar way.

ES	Extra Single	ED	Extra Double
EP	Extra Pair	ET	Extra Triple

Table 2.3 Free electrons and electron pairs.



Example 2.2

```
\startchemical[frame=on]
  \chemical[ONE,Z0,ES1,ED3,ET5,EP7][C]
\stopchemical
```

Within a ring structure you can make a shortcut from one atom to another. In that case the atom that you want to skip has to be identified. As a replacement of the double bonds in an aromatic sixring a circle can be drawn and charges can be placed within the ring.

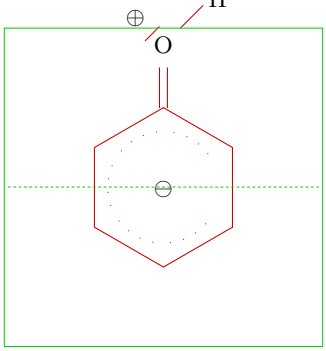
SS	Short Shortcut	S	Shortcut
-SS	Left Short Shortcut	MID	Open Mid Shortcut
+SS	Right Short Shortcut	MIDS	Closed Mid Shortcut

Table 2.4 Special bonds.

C	Circle	CD	Dashed Circle
CC	Shifted Circle	CCD	Dashed Shifted Circle

Table 2.5 Circle bonds.

An example will explain the use of the circular bond and the use of displaced charges.



Example 2.3

```
\startchemical[frame=on]
  \chemical
    [SIX,B,ER6,CCD12346,Z0,PB:RZ6,ONE,SB8,EP6,Z0,ZT6,Z8,PE]
    [\ominus,O,\oplus,H]
\stopchemical
```

Substituents can be connected to all corners of a structure. A substituent can be anything you want. It depends on the presence of atoms or molecules whether the bonds are long or short. In the examples you will see a great number of the keys that are used to define substituents.

R	Radical	SR	Single Radical
-R	Left Radical	-SR	Single Left Radical
+R	Right Radical	+SR	Single Right Radical

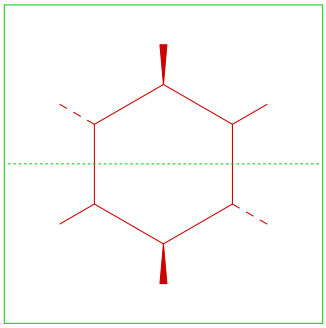
Table 2.6 Bonds to substituents.

There are a few alternatives to draw bridges.

RD	Radical Dashed	RB	Radical Bold
-RD	Left Radical Dashed	-RB	Left Radical Bold
+RD	Right Radical Dashed	+RB	Right Radical Bold

Table 2.7 Special bonds to substituents.

Radicals can be drawn in three ways.[3] Some alternatives are seldom used.



Example 2.4

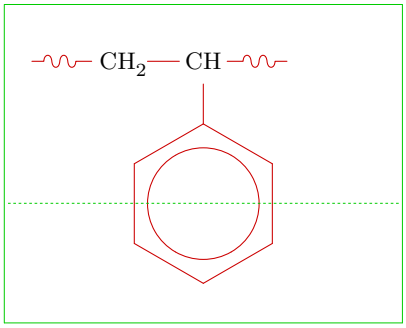
```
\startchemical[frame=on]
  \chemical[SIX,B,R14,RD25,RB36]
\stopchemical
```

SD	Single Dashed	LDD	Left Double Dashed
OE	Open Ended	RDD	Right Double Dashed

Table 2.8 More special bonds to substituents.

An example of an *Open Ended* is defined below. We see a sixring (SIX) with a number of consecutive ONEs. The use of PB is explained later.

[3] The word radical schould not be interpreted chemically, but typographically.



Example 2.5

```
\startchemical
    [width=5000,top=2500,bottom=1500,frame=on]
  \chemical
    [SIX,B,C,R6,
     PB:RZ6,ONE,CZ0,OE1,SB5,MOV5,CZ0,OFF5,OE5,PE]
    [CH,CH_2]
\stopchemical
```

It's obvious that substituents can be attached to the structure by means of double bonds.

ER	Extra Radical	DR	Double Radical

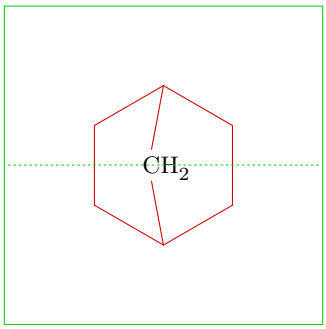
Table 2.9 Double bonds to substituents.

You can comment on a bond. Text is typed in the second argument of `\chemical`.

Z	Atom	RZ	Radical Atom
CRZ	Center Atom	-RZ	Left Radical Atom
MIDZ	Mid Atom	+RZ	Right Radical Atom

Table 2.10 Atoms and molecules (radicals).

From these keys `RZ` is an addition to the key `R`. The key `MID` is only available in combination with a sixring (`SIX`). In the example below we see the effects of `MID` and `MIDZ`. These keys have no positioning parameter.



Example 2.6

```
\startchemical[frame=on]
  \chemical[SIX,B,MID,MIDZ][\SL{CH_2}]
\stopchemical
```

Atoms and molecules are numbered clockwise. Combinations are also allowed. Position `0` (zero) is the middle of a structure.

We can attach labels and numbers to an atom or a bond. This is done with `ZN` and `ZT`:

ZN	Atom Number	ZT	Atom Text

Table 2.11 Labels and numbers.

In case of a SIX and a FIVE we can also attach text to radicals. We use RN and RT.

RN	Radical Number	RT	Radical Text
RTN	Radical Top Number	RTT	Radical Top Text
RBN	Radical Bottom Number	RBT	Radical Bottom Text

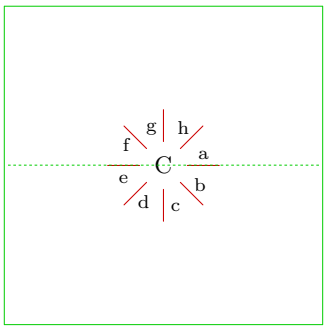
Table 2.12 Labels and numbers.

The structure ONE has also a top and bottom alternative.

ZTN	Atom Top Number	ZTT	Atom Top Text
ZBN	Atom Bottom Number	ZBT	Atom Bottom Text

Table 2.13 Extra labels and numbers.

With the keys ZTN and ZBN numbers are generated automatically. The other keys will use the typed text of the second argument of chemical.



Example 2.7

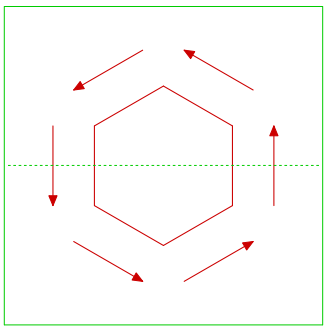
```
\startchemical[frame=on]
  \chemical[ONE,SB,ZO,ZTT][C,a,b,c,d,e,f,g,h]
\stopchemical
```

You can also add some symbols to the structure.

AU	Arrow Up	AD	Arrow Down

Table 2.14 Indications.

The arrows are positioned between the atoms in a structure.



Example 2.8

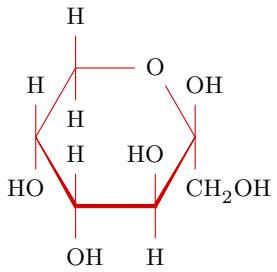
```
\startchemical[frame=on]
  \chemical[SIX,B,AU]
\stopchemical
```

We want to add that while typesetting atoms and molecules the dimensions of these atoms and molecules are taken into account. The width of C and the height of C_m^n play an important role during positioning. This mechanism may be refined in a later stage.

3 Frontviews

Structures `FIVE` and `SIX` can be displayed in a frontview. However there are some limitations. Frontviews can not be rotated. Also the coupling of several structures is limited.

We illustrate the frontview keys in two examples.

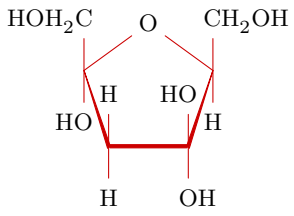


β-D-Fructopyranose

Example 3.1

```
\startchemical[height=4500,bottom=2500]
  \bottext{$\beta$-D-Fructopyranose}
  \chemical
    [SIX,FRONT,BB1236,+SB4,-SB5,Z5][O]
  \chemical
    [SIX,FRONT,+R12346,+RZ12346][\SR{HO},H,H,H,OH]
  \chemical
    [SIX,FRONT,-R12346,-RZ12346][H,OH,\SR{HO},H,CH_2OH]
\stopchemical
```

Positioning the radicals is an optimization of feasability and quality. The next example will illustrate this.



α-D-Fructofuranose

Example 3.2

```
\startchemical[height=4500,bottom=2500]
  \bottext{$\alpha$-D-Fructofuranose}
  \chemical
    [FIVE,FRONT,BB125,+SB3,-SB4,Z4][O]
  \chemical
    [FIVE,FRONT,+R1235,+RZ1235][\SR{HO},H,\SR{HOH_2C},CH_2OH]
  \chemical
    [FIVE,FRONT,-R1235,-RZ1235][OH,H,\SR{HO},H,CH_2OH]
\stopchemical
```

4 Definitions

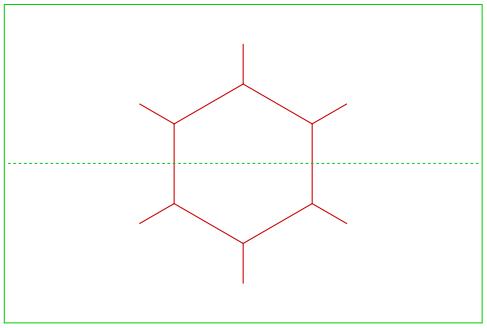
It is possible to build a library of structures. These predefined structures can be used in a later stage, for example as a component of a more complex structure. Predefinition can be done with the TEX–primitive `\def`.

```
\def\sixring{\chemical[SIX,B,R,RZ]}
```

However it is better to use the command `\definechemical`. In that case a message will occur during processing if a duplicate name is found.

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ]}
```

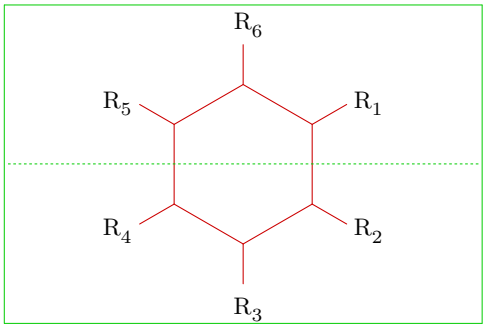
Recalling `\chemical[sixring]` will display a bare sixring without substituents.



Example 4.1

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ]}

\startchemical[frame=on,width=6000]
  \chemical[sixring]
\stopchemical
```

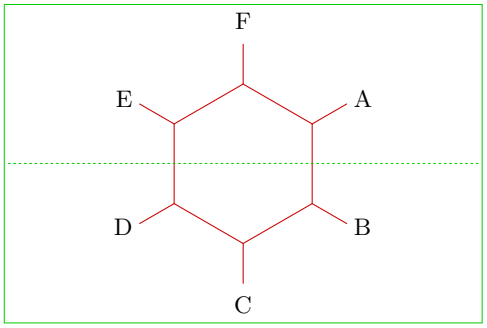
If we want to attach six substituents in a later stage to a sixring we could type:



Example 4.2

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ]}

\startchemical[frame=on,width=6000]
  \chemical[sixring][R_1,R_2,R_3,R_4,R_5,R_6]
\stopchemical
```

The structure `sixring` can be defined without substituents (`RZ`). We could attach them after recalling `\chemical[sixring]`.



Example 4.3

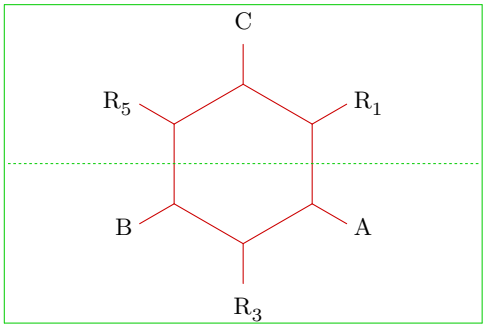
```
\definechemical[sixring]
  {\chemical[SIX,B,R]}

\startchemical[frame=on,width=6000]
  \chemical[sixring,RZ][A,B,C,D,E,F]
\stopchemical
```

In principal the possibilities are unlimited. However, you should remember that atoms and molecules are selected from the second argument in the order of definition in the first argument.

A definition may contain atoms and molecules (texts).

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ135][R_1,R_3,R_5]}
```

In the example above there will always be three substituents. If we want to attach more substituents we have to indicate explicitly that we want to continue with the sixring (`SIX`).

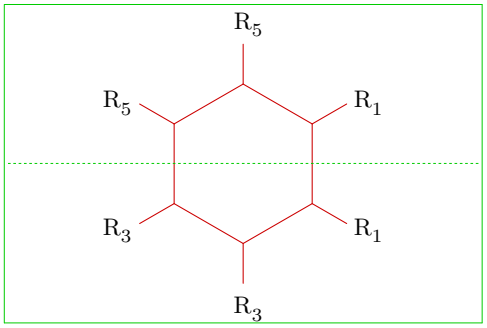


Example 4.4

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ135][R_1,R_3,R_5]}

\startchemical[frame=on,width=6000]
  \chemical[sixring,SIX,RZ246][A,B,C]
\stopchemical
```

In a definition `\chemical[]` has a global scope (this means that `SIX` is remembered) and `\chemical[][]` has a local scope. The idea behind this is that in the first case a range of keys can be added and in the second case a complete structure.

In a definition `\chemical` may be used more than once. The last example could have been defined thus:



Example 4.5

```
\definechemical[sixring]
  {\chemical[SIX,B,R,RZ135][R_1,R_3,R_5]
   \chemical[SIX,RZ246]}

\startchemical[frame=on,width=6000]
  \chemical[sixring][A,B,C]
\stopchemical
```

When TEX announces that an `unknown command` has occurred, you may have forgotten to type `SIX`, `FIVE` or a comparable key.

5 Combinations

Structures can be combined to more complex compounds. Moving one structure in relation to another structure is done by `MOV`, `ROT`, `ADJ` and `SUB`.

`MOV`	Move	moving a comparable structure structure in the direction of a bond
`ADJ`	Adjace	moving another structure in the direction of the x– or y–axis, adjacent to a bond
`SUB`	Substitute	moving one structure in relation to another in the direction of the x– or y–axis
`ROT`	Rotate	rotating a structure

Table 5.1 Moving and rotating.

The four keys mentioned above have different effects when they are applied to different structures. The angle of rotation in `\chemical[FIVE,ROT1,B]` differs from that in `\chemical[SIX,ROT1,B]`.

With the structure `ONE` you can use `MOV` but the key `DIR` is also available. Both keys have the same effect but differ in spacing. Small adjustments are possible with `OFF`.

`DIR`	Direction	moving a structure in a diagonal direction
`OFF`	Offset	moving atoms and molecules over small distances

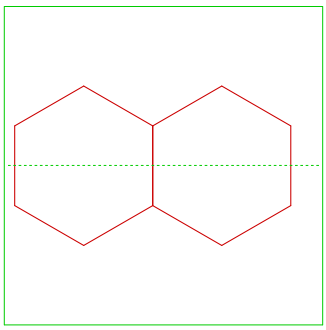
Table 5.2 Moving and rotating.

The structure `CARBON` can be mirrored with `MIR`.

`MIR`	Mirror	mirroring a structure

Table 5.3 Mirroring.

We use a number to indicate the direction of a movement or the level of rotation. These keys are closely related with the structure. Therefore they have to be defined before bonds are drawn and texts are placed. So definition `\chemical[FIVE,B,ROT1,R]` and `\chemical[FIVE,ROT1,B,R]` will not have the same result. The first definition will give an undesirable result.



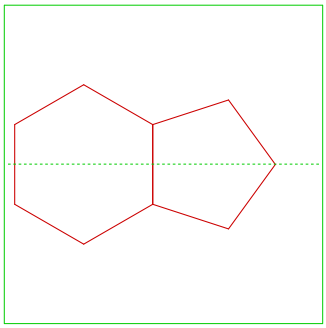
Example 5.1

```
\startchemical[frame=on,width=4000,right=3000]
  \chemical[SIX,B,MOV1,B]
\stopchemical
```

In this example a sixring is drawn because of `SIX,B`. Then a movement in the direction of bond 1 takes place and a second sixring is drawn: `B` (`SIX` is stil in effect).

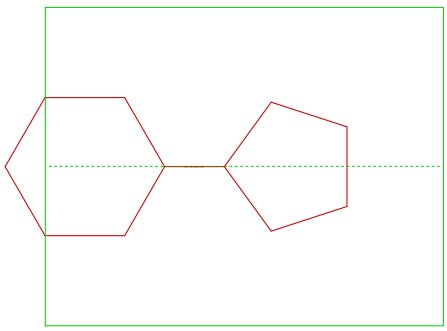
A movement with `MOV` in a sixring can occur in six directions. A movement with `ADJ` will take place in only four axis–directions (x, $-x$, y, $-y$). It is a coincidence that in a sixring some of these movements have the same effect. The example above could have been drawn with: `[SIX,B,ADJ1,B]`.

Structures can be combined. It is possible for example to combine structure `FIVE` with structure `SIX` in such a way that they have one mutual bond. Luckily the mechanism that takes care of these kinds of combinations is hidden for the user. In the next example you will see a sixring drawn by `SIX,B`. Then a movement in the positive x–direction is done by `ADJ1`. At last a rotated fivering is drawn: `FIVE,ROT3,B`.



Example 5.2

```
\startchemical[frame=on,width=4000,right=3000]
  \chemical[SIX,B,ADJ1,FIVE,ROT3,B]
\stopchemical
```

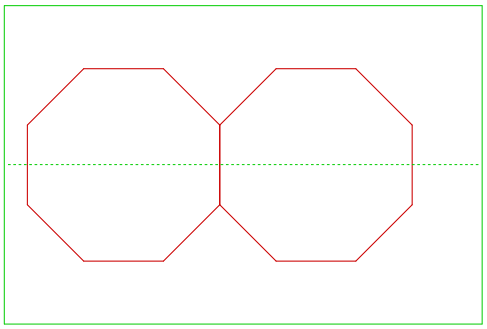


Example 5.3

```
\startchemical[frame=on,width=5000,right=4500]
  \chemical[SIX,ROT2,B,R6,SUB1,FIVE,B,R4]
\stopchemical
```

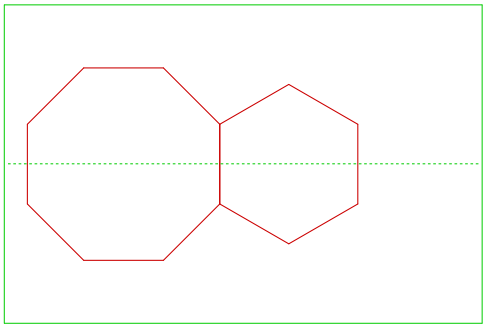
To go from one structure to an adjacent one is done with `ADJ`. Most of the time one of these structures will have to be rotated to obtain a good attachment. This is done by `ROT`. Rotations are always clockwise in steps of 90°. When a structure is attached with a bond you will have to use `SUB`. Movements with `ADJ` and `SUB` take place in the four directions of the x- and y–axis.

The next examples illustrate that the dimensions of the smaller structures are determined by the larger structures, especially `SIX`. You will notice that `EIGHT` has fewer possibilities than `SIX`.



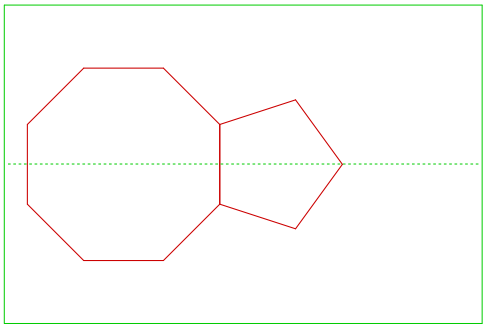
Example 5.4

```
\startchemical[width=6000,left=1500,frame=on]
  \chemical[EIGHT,B,MOV1,B]
\stopchemical
```



Example 5.5

```
\startchemical[width=6000,left=1500,frame=on]
  \chemical[EIGHT,B,ADJ1,SIX,B]
\stopchemical
```



Example 5.6

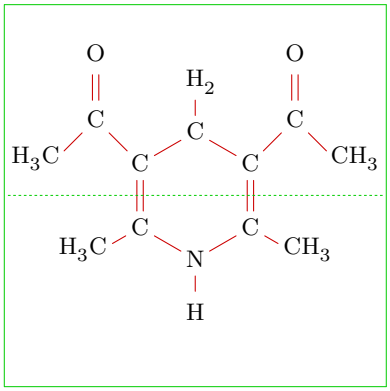
```
\startchemical[width=6000,left=1500,frame=on]
  \chemical[EIGHT,B,ADJ1,FIVE,ROT3,B]
\stopchemical
```

It will be clear by now that the order in which the keys are defined makes a lot of difference. The order should be:

```
\chemical
  [structure,                                % SIX, FIVE, ...
   bonds within the structure,               % B, C, EB, ...
   bonds pointing to substituents,           % R, DR, ...
   atoms within the structure,               % Z
   subsituents attached to the structure]    % RZ, -RZ, ...
  [atoms,
   substituents]
```

Most of the time putting structures together is done by translating and rotating. You could automate this process. In earlier versions this was done automatically, however this led to misinterpretations of users concerning the positions of bonds, atoms and substituents. A structure that consists of more than one component can best be defined per component, translations included. Rotations should wait until the last step.

A sixring may have substituents consisting of a carbon chain. In those situations we use `DIR` to build the chain.



Example 5.7

```
\startchemical
  [scale=small,width=6000,height=6000,frame=on]
  \chemical[SIX,SB2356,DB14,Z2346,SR36,RZ36]
    [C,N,C,C,H,H_2]
  \chemical[PB:Z1,ONE,Z0,DIR8,Z0,SB24,DB7,Z27,PE]
    [C,C,CH_3,O]
  \chemical[PB:Z5,ONE,Z0,DIR6,Z0,SB24,DB7,Z47,PE]
    [C,C,H_3C,O]
  \chemical[SR24,RZ24]
    [CH_3,H_3C]
\stopchemical
```

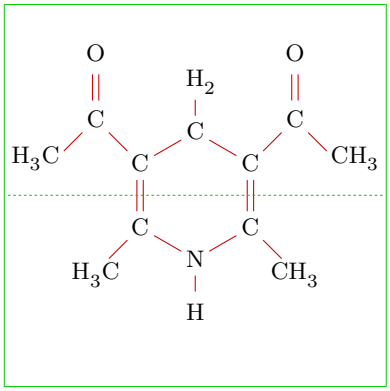
Because chains have no predefined format the chains are build and positioned as a substructure. For positioning we use the keys `PB` and `PE`.

`PB:..`	Picture Begin	beginning a substructure
`PE`	Picture End	ending a substructure

Table 5.4 Positioning.

Directly after `PB` you will have to define the location where the substructure is positioned. The first following atom is centered on that location. Always use a central atom on this location.

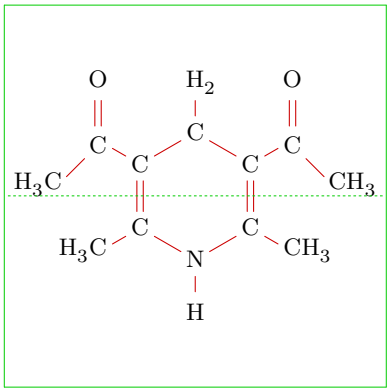
These keys were introduced after trying to obtain structures that are typeset in an acceptable quality. There are different ways to define structures. The following alternative would have resulted in:



Example 5.8

```
\startchemical
    [scale=small,width=6000,height=6000,frame=on]
  \chemical
    [SIX,SB2356,DB14,Z36,SR36,RZ36][N,C,H,H_2]
  \chemical
    [PB:Z1,ONE,Z0,DIR8,Z0,SB24,DB7,Z27,PE][C,C,CH_3,O]
  \chemical
    [PB:Z5,ONE,Z0,DIR6,Z0,SB24,DB7,Z47,PE][C,C,H_3C,O]
  \chemical
    [PB:Z2,ONE,Z0,DIR2,SB6,CZ0,PE][C,CH_3]
  \chemical
    [PB:Z4,ONE,Z0,DIR4,SB8,CZ0,PE][C,H_3C]
\stopchemical
```

The most efficient way to define such a structure would be like the example below. Typographically you wouldn't be satisfied.

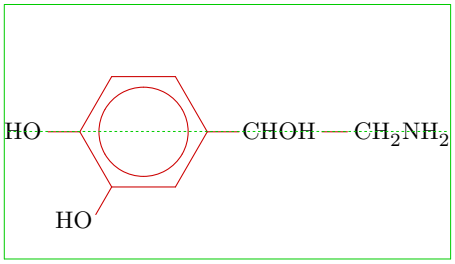


Example 5.9

```
\startchemical
    [scale=small,width=6000,height=6000,frame=on]
  \chemical[SIX,SB2356,DB14,Z,SR36,RZ36,SR1245,RZ24]
    [C,C,N,C,C,C,H,H_2,CH_3,H_3C]
  \chemical[PB:RZ1,ONE,Z0,SB2,DB7,Z27,PE]
    [C,CH_3,O]
  \chemical[PB:RZ5,ONE,Z0,SB4,DB7,Z47,PE]
    [C,H_3C,O]
\stopchemical
```

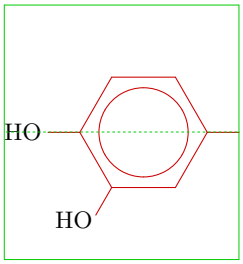
You may have noticed that the measurements of the structure is determined by the substituents. The chains are not taken into account. This leads to a consistent build-up of a structure.

The differences in outcome when using `SUB` in stead of `PB` are very small. However compare the following formulas.



Example 5.10

```
\startchemical
  [width=fit,frame=on,scale=small]
  \chemical
    [SIX,ROT2,B,C,R236,RZ23,
     SUB1,ONE,OFF1,Z0,4OFF1,SB1,Z1]
    [HO,HO,CHOH,CH_2NH_2]
\stopchemical
```

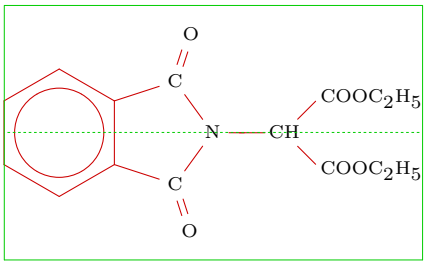


Example 5.11

```
\startchemical
  [width=fit,frame=on,scale=small]
  \chemical
    [SIX,ROT2,B,C,R236,RZ23,
     PB:RZ6,ONE,Z0,3OFF1,SB1,Z1,PE]
    [HO,HO,CHOH,CH_2NH_2]
\stopchemical
```

The use of the key `PB:` might be somewhat more difficult, but the results are much better. In that case you should define the width yourself, because the substituents are not taken into account when determining the dimensions.

First we will go into the key `OFF`. In some cases atom (`Z0`) in `ONE` can consist of more than one character. The reserved space for these characters would be insufficient and character and bond would overlap. When you need more space for `Z0` we can move bond 1, 2 and 8 by means of the key `OFF` ('offset'). The example below will illustrate its use.



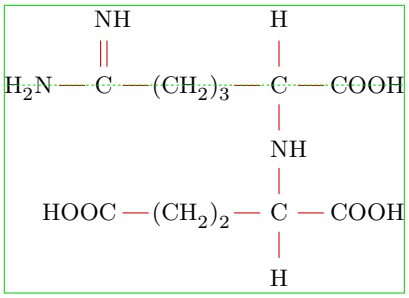
Example 5.12

```
\startchemical
  [width=fit,size=small,scale=small,frame=on]
  \chemical
    [SIX,B,C,ADJ1,
     FIVE,ROT3,SB34,+SB2,-SB5,Z345,DR35,SR4,CRZ35,SUB1,
     ONE,OFF1,SB258,Z0,Z28]
    [C,N,C,O,O,
     CH,COOC_2H_5,COOC_2H_5]
\stopchemical
```

Moving the bonds makes room for an extra character. More space was obtained when we would have typed `3OFF1`. The example looks rather complex but you can define it rather easy by defining its components first. Rotating should be done in the last stage.

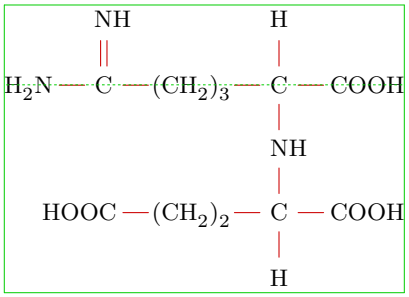
You see a new key: `CRZ`. This key is used to place the atom or molecule in one line with the bond. You could have used `RZ`, because you can influence spacing in the second argument with `{\,O}` in stead off `O` (spacing in mathematical mode).

We will show another example, produced in two ways. When choosing a method you should take into account the consistency throughout your document.



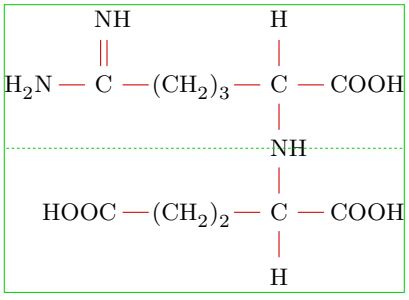
Example 5.13

```
\startchemical
  [width=fit,height=fit,frame=on,
   scale=small]
  \chemical
    [ONE,SB15,DB7,Z057,3OFF1,MOV1,Z0,3OFF1,MOV1,
     Z017,SB1357,MOV3,Z0,MOV3,SB1357,Z013,3OFF5,
     MOV5,Z0,3OFF5,SB5,Z5]
    [C,H_2N,NH,(CH_2)_3,C,COOH,H,\SL{NH},C,COOH,H,
     (CH_2)_2,HOOC]
\stopchemical
```



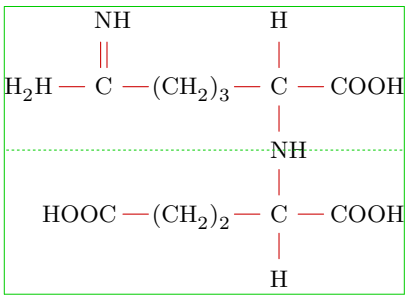
Example 5.14

```
\startchemical
  [width=fit,height=fit,frame=on,
   scale=small]
  \chemical [ONE,SB15,DB7,Z057,3OFF1][C,H_2N,NH]
  \chemical [MOV1,Z0,3OFF1]          [(CH_2)_3]
  \chemical [MOV1,Z017,SB1357]       [C,COOH,H]
  \chemical [MOV3,Z0]                [\SL{NH}]
  \chemical [MOV3,SB1357,Z013,3OFF5] [C,COOH,H]
  \chemical [MOV5,Z0,3OFF5,SB5,Z5]   [(CH_2)_2,HOOC]
\stopchemical
```



Example 5.15

```
\startchemical
  [width=fit,height=fit,frame=on,
   scale=small]
  \chemical
    [ONE,Z0,SAVE,MOV7,SB1357,Z017,3OFF5,MOV5,Z0,
     3OFF5,MOV5,SB15,DB7,Z057,RESTORE,
     MOV3,SB1357,Z013,MOV5,3OFF5,Z0,6OFF5,SB5,Z5]
    [\SL{NH},C,COOH,H,(CH_2)_3,C,H_2N,NH,C,COOH,H,
     (CH_2)_2,HOOC]
\stopchemical
```

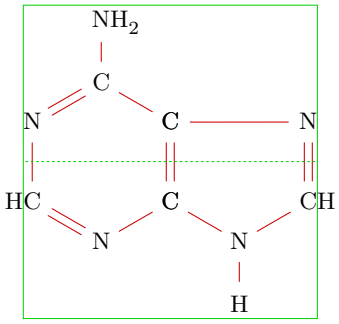


Example 5.16

```
\startchemical
  [width=fit,height=fit,frame=on,scale=small]
  \chemical
    [ONE,Z0,MOV7,SB1357,Z017,3OFF5,MOV5,Z0,
     3OFF5,MOV5,SB15,DB7,Z057,MOV0,MOV3,SB1357,
     Z013,MOV5,3OFF5,Z0,6OFF5,SB5,Z5]
    [\SL{NH},C,COOH,H,(CH_2)_3,C,H_2H,NH,C,COOH,H,
     (CH_2)_2,HOOC]
\stopchemical
```

Notice the use of `SAVE` and `RESTORE`. These keys enable you to save a location in a structure and return to that location in another stage.

As an extra we will show you a combination of `SIX` and `FIVE`. Be aware of the use of `SS`.

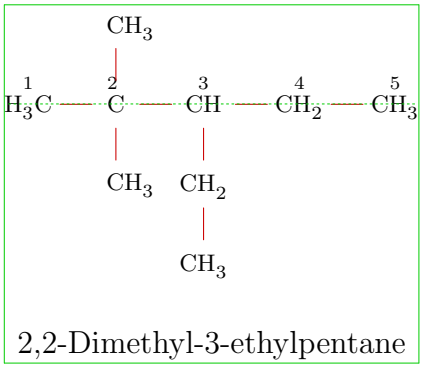


Example 5.17

```
\startchemical
    [width=fit,height=fit,frame=on]
  \chemical
    [SIX,DB135,SB246,Z,SR6,RZ6][C,C,N,\SR{HC},N,C,NH_2]
  \chemical
    [SIX,MOV1,DB1,SB23,SS6,Z1..5,SR3,RZ3][N,\SL{CH},N,C,C,H]
\stopchemical
```

6 Extra text

We can add text and symbols in and around structures. For example:



2,2-Dimethyl-3-ethylpentane

Example 6.1

```
\startchemical
  [height=4500,top=1250,width=fit,frame=on]
  \bottext
    {2,2-Dimethyl-3-ethylpentane}
  \chemical
    [ONE,Z3570,SB1357]
    [CH_3,\T{1}{H_3C},CH_3,\SR{\LT{2}{C}}]
  \chemical
    [MOV1,OFF1,Z0,SB3]
    [\T{3}{CH}]
  \chemical
    [MOV3,Z0,SB3,MOV3,Z0,MOV7,MOV7]
    [CH_2,CH_3]
  \chemical
    [OFF1,SB1,MOV1,OFF1,Z0,2OFF1,SB1,Z1]
    [\T{4}{CH_2},\T{5}{CH_3}]
\stopchemical
```

There is a range of keys like \T. In a number of cases the arguments are optional. Charges can be displayed in Roman by means of \+ and \- or directly by means of \1 up to \7.

\+{number}	positive charge in Roman
\-{number}	negative charge in Roman
\1	I (without sign)
\7	II, III, IV, V, VI and VII

Table 6.1 Text: charges.

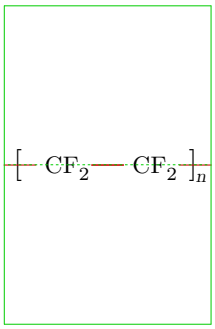
A charge is centered above the atom. For example:

```
\placeformula
  \startformula
    \chemical{S}+\chemical{O_2}
    \chemical{GIVES}
    \chemical{\+{4}{S}\-{2}{O_2}}
  \stopformula
```

will result in:

$$\mathrm{S} \quad + \quad \mathrm{O_2} \qquad \longrightarrow \qquad \overset{+\mathrm{IV}\,-\mathrm{II}}{\mathrm{S\,O_2}} \tag{6.1}$$

If we want to repeat a number of atoms or molecules we can define an (endless) range with `\[` and `\]`. Both arguments are optional as is shown in the example of PTFE of Polytetrafluorethane, better knowns as Teflon.



Example 6.2

```
\startchemical[width=fit,frame=on]
  \chemical
    [ONE,ZT5,SB5,OFF1,Z0,OFF1,SB1,MOV1,SB5,OFF1,Z0,OFF1,SB1,ZT1]
    [\[,CF_2,CF_2,\]{\sl n}]
\stopchemical
```

`\[{bottom}`	`\[{top}{bottom}`	right repeating sign
`\]{bottom}`	`\]{top}{bottom}`	left repeating sign

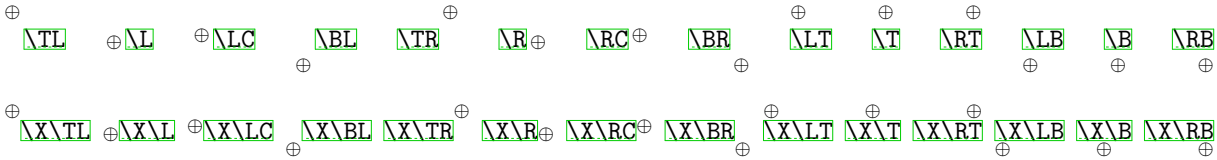
Table 6.2 Text: repeating.

There is no problem of placing texts on the left, right, top or bottom of the atoms or molecules. If we preceed the keys `\L`, `\R`, `\T` and `\B` by `\X` the distance from text to atoms is somewhat smaller.

`\L{text}`	text left
`\R{text}`	text right
`\T{text}`	text top
`\B{text}`	text bottom

Table 6.3 Text: around an atom.

Logical combinations of these keys are also possible. A key to centre text is also available.

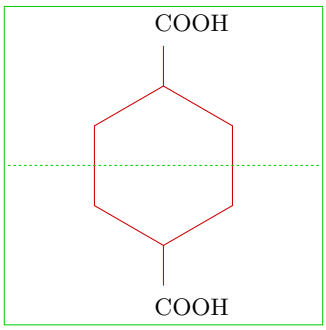


In some cases you will need what we may call *smashed* text.

\SL{text}	left align
\SM{text}	centre
\SR{text}	right align

Table 6.4 Text: smashed text.

An example is given below. The text is centred around the first character.



Example 6.3

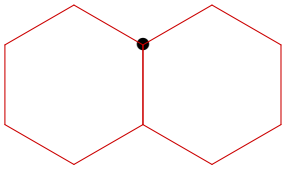
```
\startchemical[frame=on]
  \chemical[SIX,B,R36,RZ36][\SL{COOH},\SL{COOH}]
\stopchemical
```

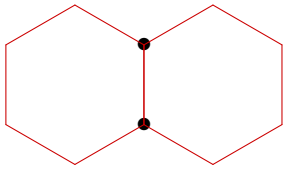
We can place text above, under or in the middle of structures with the keys \toptext, \midtext and \bottext. The exact position is determined by the height and depth of the structure.

```
\placeformula
  \startformula
    \startchemical[width=fit]
      \chemical[SIX,B,Z1,MOV1,B][\hbox{$\bullet$}]
      \toptext{{\sl trans}-Decalin}
    \stopchemical
    \hskip 24pt
    \startchemical[width=fit]
      \chemical[SIX,B,Z12,MOV1,B][\hbox{$\bullet$},\hbox{$\bullet$}]
      \bottext{{\sl cis}-Decalin}
    \stopchemical
  \stopformula
```

Both *Decalin* formulas look like this:

trans-Decalin



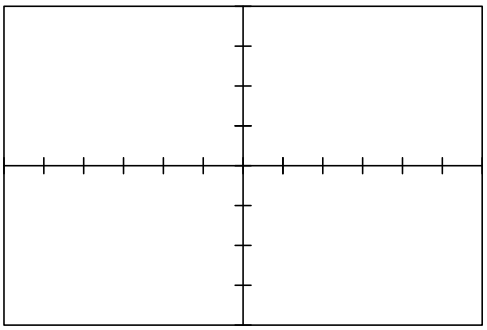


(6.2)

cis-Decalin

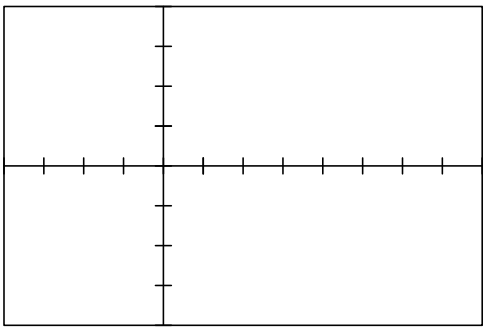
7 Axis

Structures can be typeset in a frame that is divided by axis. The dimensions of the axis and the location of the origin can be defined in the set up. The axis and the frame can be made visible.



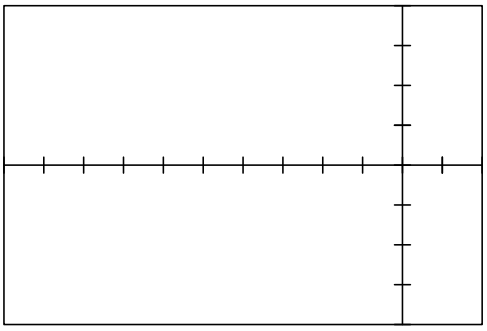
Example 7.1

```
\startchemical
  [axis=on,
   width=6000,height=4000]
\stopchemical
```



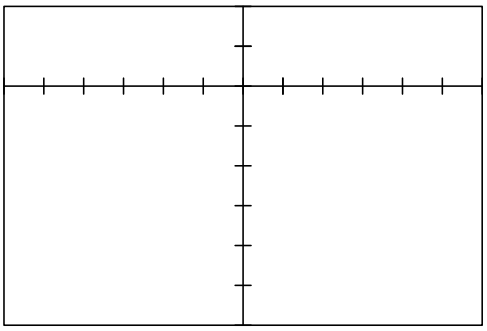
Example 7.2

```
\startchemical
  [axis=on,
   left=2000,right=4000]
\stopchemical
```



Example 7.3

```
\startchemical
  [axis=on,
   width=6000,right=1000]
\stopchemical
```



Example 7.4

```
\startchemical
  [axis=on,
   width=6000,top=1000,bottom=3000]
\stopchemical
```

The dimensions of the total structure determine the dimensions of the axis. When `width=fit` and/or `height=fit` is typed the dimensions are determined by the real dimensions. Your choice will depend on how you want to place the structure in the text. **Example 7.1** shows the default set up. Within a `\start–\stop`–pair you can use PICTEX–macros. However, be careful.

8 Set ups

After \startchemical and \setupchemical you can type the set up.

parameter	values	default
width	number	4000
height	number	4000
left	number	
right	number	
top	number	
bottom	number	
resolution	number	\outputresolution
bodyfont	8pt 9pt 10pt etc.	\bodyfontsize
character	\rm \bf etc.	\rm
scale	number	medium
size	small medium big	medium
state	start stop	start
option	test	
axis	on off	off
frame	on off	off
alternative	1 2	1
offset	HIGH LOW	LOW

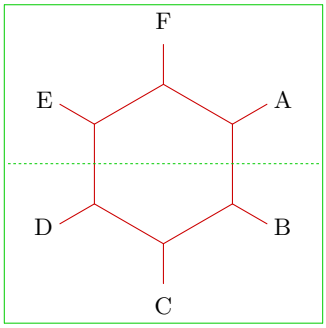
Table 8.1 Set ups for structures.

The axis range from −2000 upto +2000, height as well as width. The parameter Z0 is at $(0, 0)$. Other divisions can be set up with left, right, top and/or bottom in combination with width and height.

You can use the key size to set up the bodyfont. In doing so the TEX–primitives \textsize, \scriptsize and \scriptscriptsize are used. With scale you can set up the dimensions of the structure itself. The scale is determined by the parameter bodyfont. The values small, medium and big are proportionally related.

The set up of the parameter `bodyfont` is used for calculations and has no consequences for the text. In ConTeXt and LaTeX this set up parameter is coupled to the mechanism that sets the bodyfont.

In TeX and ConTeXt you can use commands like `\rm`, `\bf` and `\sl` in mathematical mode. PPCHTeX uses default `\rm`. With the parameter `character` another alternative can be chosen. In **example 8.1** the substituents are typeset *slanted*.

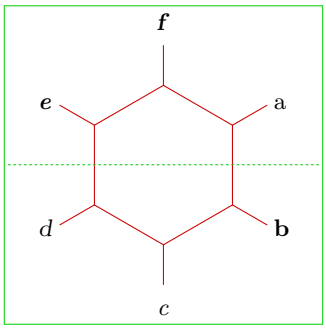


Example 8.1

```
\startchemical[frame=on,character=\sl]
  \chemical[SIX,B,R,RZ][A,B,C,D,E,F]
\stopchemical
```

The set up of `character` is valid for chemical structures in a picture and in the text. The sub- and superscripts are changed accordingly. This is illustrated in CH_4, CH_4 and CH_4, in which the set ups are `\rm`, `\bf` and `\sl`. Italic `\it` formulas lead to a bigger linewidth. In ConTeXt default bold–slanted (`\bs`) and bold–italic (`\bi`) are available. These commands adjust automatically to the actual fontstyle: CH_4, CH_4, CH_4 etc. (`\ss`, `\rm`, `\tt`).

It is also possible to set the characters at the instant you type them in the argument.



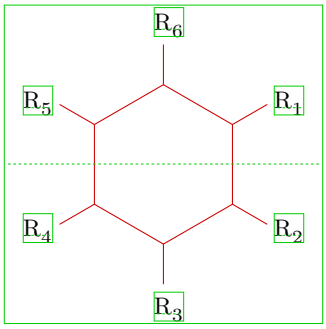
Example 8.2

```
\startchemical[frame=on]
  \chemical[SIX,B,R,RZ][\tf a,\bf b,\it c,\sl d,\bi e,\bs f]
\stopchemical
```

With parameter `state` calculations can be shortcut. The parameters `frame` and `axis` need no further explanation. With `option=test` frames are drawn around the texts in a structure. In this way you can see how the text is aligned.[4] With the parameter

[4] In ConTeXt you can activate the visual debugger. When activated the baseline is a dotted line. The module `supp-vis` can be used in other systems.

`alternative` you can set up the quality of the lines. Default PICTEX uses a 5 point . to draw lines. When option 2 is chosed a thinner line is drawn.

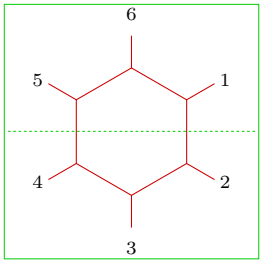


Example 8.3

```
\startchemical[frame=on,option=test,alternative=2]
  \chemical[SIX,B,R,RZ][R_1,R_2,R_3,R_4,R_5,R_6]
\stopchemical
```

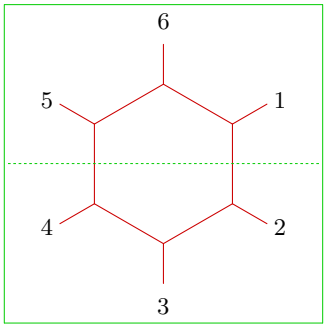
The offset relates to the position of the sub- and superscripts. With `HIGH` the subscripts are placed high (H_2O) and with `LOW` somewhat lower (H_2O).

A structure can be displayed in different sizes. This is done with `formaat` and `scale`.



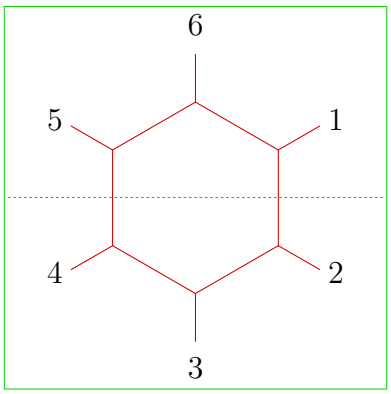
Example 8.4

```
\startchemical[frame=on,scale=small,size=small]
  \chemical[SIX,B,R,RZ][1,2,3,4,5,6]
\stopchemical
```



Example 8.5

```
\startchemical[frame=on,scale=medium,size=medium]
  \chemical[SIX,B,R,RZ][1,2,3,4,5,6]
\stopchemical
```



Example 8.6

```
\startchemical[frame=on,scale=big,size=big]
  \chemical[SIX,B,R,RZ][1,2,3,4,5,6]
\stopchemical
```

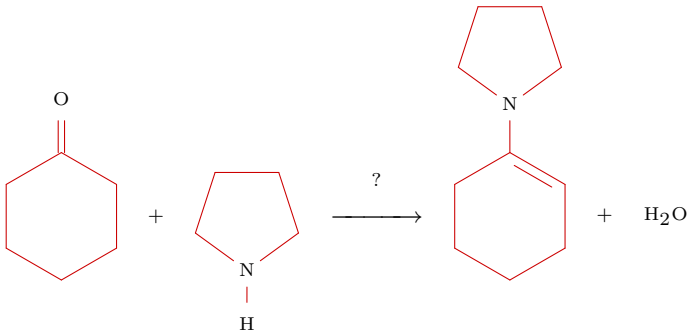
You can also type a number between 1 and 1000 in `scale`. The values belonging to `small`, `medium` or `big` are proportionally related.

9 Symbols

There are some symbols that can be used to display reactions. The reaction below is typed by:

```
\setupchemical
  [size=small,
   scale=small,
   width=fit,
   height=5500,
   bottom=1500]

\hbox
  {\startchemical
     \chemical[SIX,B,ER6,RZ6][O]
   \stopchemical
   \startchemical
     \chemical[SPACE,PLUS,SPACE]
   \stopchemical
   \startchemical
     \chemical[FIVE,ROT4,B125,+SB3,-SB4,Z4,SR4,RZ4][N,H]
   \stopchemical
   \startchemical
     \chemical[SPACE,GIVES,SPACE][?]
   \stopchemical
   \startchemical
     \chemical[SIX,B,EB6,R6,SUB4,FIVE,ROT4,B125,+SB3,-SB4,Z4][N]
   \stopchemical
   \startchemical
     \chemical[SPACE,PLUS,SPACE,CHEM][H_2O]
   \stopchemical}
```

The `\hbox` is necessary to align the structures. The symbols `GIVES` and `PLUS` need no further explanation. With `SPACE` more room can be created between the structures and symbols.



An equilibrium can be displayed with `EQUILIBRIUM`. Over `GIVES` and `EQUILIBRIUM` you can place text. In the example the text is just a '?'. In addition `MESOMERIC` is also available. Braces used for displaying complexes can be created with `OPENCOMPLEX` and `CLOSECOMPLEX`.

10 Positioning

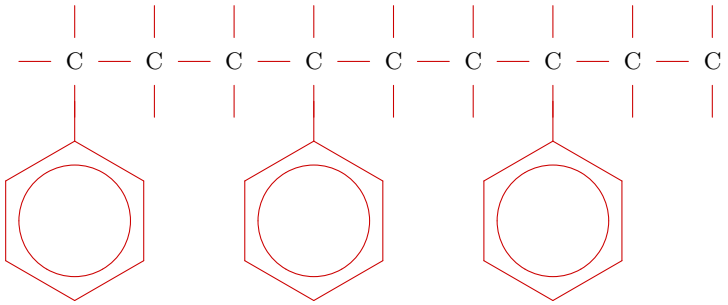
When you are combining atoms or molecules, for example with SUB, some positions and dimensions change their value. To overcome this problem it is possible to save a location with SAVE and return to that location with RESTORE.

SAVE	Save Status	save actual status
RESTORE	Restore Status	restore actual status

Table 10.1 Positioning.

The keys SAVE and RESTORE are used with substituents. When placing radicals we use PB and PE. This example also illustrates the possibility to create chains.

```
\definechemical[molecule]
  {\chemical
     [ONE,Z0,SB1357,
      SAVE,SUB2,SIX,B,R6,C,RESTORE,
      MOV1,Z0,SB137,
      MOV1,Z0,SB37,
      MOV1]
     [C,C,C]}

\startchemical[width=fit,height=fit]
  \chemical[molecule,molecule,molecule]
\stopchemical
```

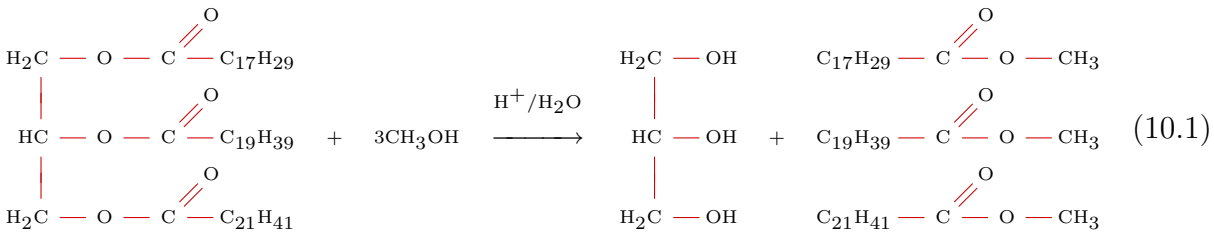


The example below is more complicated and show a complete reaction. The set up of bottom and top is essential in this example.

```
\placeformula
  \startformula
    \setupchemical
      [width=fit,top=2000,bottom=2000,
       scale=small,size=small]
    \startchemical
      \chemical
        [ONE,
         SAVE,
           Z0,SB7,SB3,SB1,MOV1,Z0,SB1,MOV1,Z0,DB8,CZ8,SB1,Z1,
         RESTORE,
         SAVE,
           SUB4,ONE,Z0,SB3,SB1,MOV1,Z0,SB1,MOV1,Z0,DB8,CZ8,SB1,Z1,
         RESTORE,
         SUB2,ONE,Z0,SB7,SB1,MOV1,Z0,SB1,MOV1,Z0,DB8,CZ8,SB1,Z1]
        [\SR{HC},O,C,O,C_{19}H_{39},
         \SR{H_{2}C},O,C,O,C_{17}H_{29},
         \SR{H_{2}C},O,C,O,C_{21}H_{41}]
    \stopchemical
    \startchemical
      \chemical[SPACE,PLUS,SPACE]
    \stopchemical
    \startchemical[right=600]
      \chemical[ONE,CZ0][3CH_{3}OH]
    \stopchemical
    \startchemical
      \chemical[SPACE,GIVES,SPACE,SPACE][H^+/H_2O]
    \stopchemical
    \startchemical
      \chemical
        [ONE,
         SAVE,
           Z0,SB7,SB3,SB1,Z1,
         RESTORE,
         SAVE,
           SUB4,ONE,Z0,SB3,SB1,Z1,
         RESTORE,
         SUB2,ONE,Z0,SB7,SB1,Z1]
        [\SR{HC},OH,
         \SR{H_{2}C},OH,
         \SR{H_{2}C},OH]
```

```
    \stopchemical
    \startchemical
      \chemical[SPACE,PLUS,SPACE]
    \stopchemical
    \startchemical
      \chemical
        [ONE,
         SAVE,
           Z0,DB8,CZ8,SB1,SB5,Z5,MOV1,Z0,SB1,Z1,
         RESTORE,
         SAVE,
           SUB4,ONE,Z0,DB8,CZ8,SB1,SB5,Z5,MOV1,Z0,SB1,Z1,
         RESTORE,
         SUB2,ONE,Z0,DB8,CZ8,SB1,SB5,Z5,MOV1,Z0,SB1,Z1]
        [C,O,C_{19}H_{39},O,CH_{3},
         C,O,C_{17}H_{29},O,CH_{3},
         C,O,C_{21}H_{41},O,CH_{3}]
    \stopchemical
  \stopformula
```

This definition might have been more compact if we would have typed `SB731` in stead of `SB7,SB3,SB1`. But in this way the definition is readable. Complex structures can best be defined in its respective components.

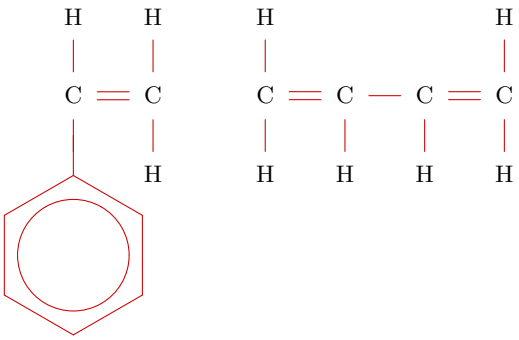


(10.1)

Just two more examples where we place text under a structure.

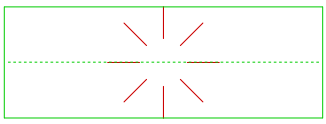
```
\placeformula
  \startformula
    \setupchemical
      [width=fit,top=1500,bottom=3500]
    \startchemical
      \chemical
        [ONE,Z0,DB1,SB3,SB7,Z7,MOV1,Z0,SB3,SB7,Z3,Z7,
         MOV0,SUB2,SIX,B,R6,C]
        [C,H,C,H,H]
```

```
    \bottext{styreen}
  \stopchemical
  \quad\quad\quad
  \startchemical
    \chemical
      [ONE,Z0,DB1,SB3,SB7,Z3,Z7,
       MOV1,Z0,SB1,SB3,Z3,
       MOV1,Z0,DB1,SB3,Z3,
       MOV1,Z0,SB3,SB7,Z3,Z7]
      [C,H,H,C,H,C,H,C,H,H]
    \bottext{1,3-butadieen}
 \stopchemical
\stopformula
```



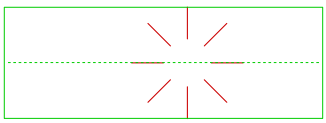
styreen 1,3-butadieen (10.2)

The use of `OFF` can be very subtle. The examples below illustrate this and show minor shifts of `ONE`.



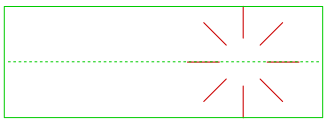
Example 10.1

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,SB]
\stopchemical
```



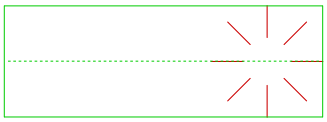
Example 10.2

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,3OFF1,SB]
\stopchemical
```



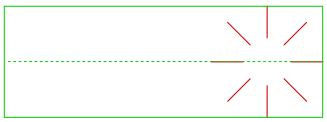
Example 10.3

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,MOV1,SB]
\stopchemical
```



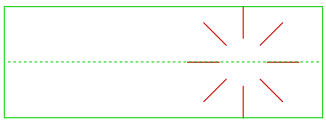
Example 10.4

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,3OFF1,MOV1,SB]
\stopchemical
```



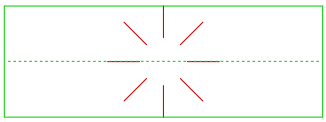
Example 10.5

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,MOV1,3OFF1,SB]
\stopchemical
```



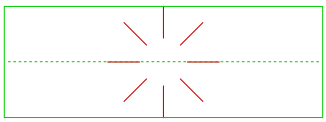
Example 10.6

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,MOV1,3OFF1,OFF0,SB]
\stopchemical
```



Example 10.7

```
\startchemical[height=fit,frame=on]
  \chemical[ONE,MOV1,3OFF1,MOV0,SB]
\stopchemical
```



Example 10.8

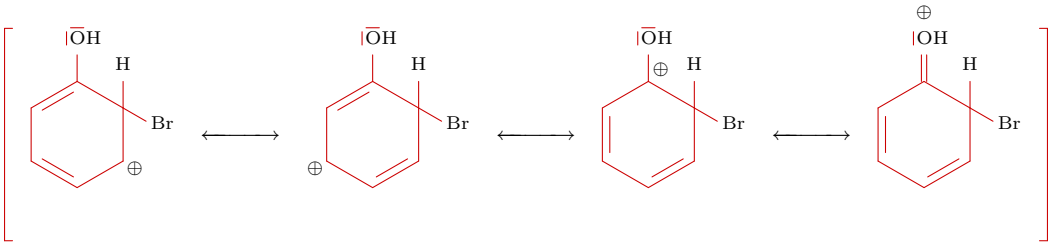
```
\startchemical[height=fit,frame=on]
  \chemical[ONE,MOV1,MOV0,SB]
\stopchemical
```

The next example shows the definition of complexes. Pay special attention to the use of `RBT`. Normally an extra spacing is not necessary but we use here —the command is not visible— a smaller bodyfont to prevent the structure to run in the margin.

```
\startformula
\setupchemical[scale=small,width=fit]
\startchemical
```

```
  \chemical[OPENCOMPLEX,SPACE]
\stopchemical
\startchemical
  \chemical[SIX,B,EB35,R6,-R1,+R1]
  \chemical[SIX,PB:RZ6,ONE,OFF1,Z0,EP57,PE][\SL{OH}]
  \chemical[SIX,-RZ1,+RZ1,RT2][H,Br,\oplus]
\stopchemical
\startchemical
  \chemical[SPACE,MESOMERIC,SPACE]
\stopchemical
\startchemical
  \chemical[SIX,B,EB25,R6,-R1,+R1]
  \chemical[SIX,PB:RZ6,ONE,OFF1,Z0,EP57,PE][\SL{OH}]
  \chemical[SIX,-RZ1,+RZ1,RT4][H,Br,\oplus]
\stopchemical
\startchemical
  \chemical[SPACE,MESOMERIC,SPACE]
\stopchemical
\startchemical
  \chemical[SIX,B,EB24,R6,-R1,+R1]
  \chemical[SIX,PB:RZ6,ONE,OFF1,Z0,EP57,PE][\SL{OH}]
  \chemical[SIX,-RZ1,+RZ1,RBT6][H,Br,\ \oplus]
\stopchemical
\startchemical
  \chemical[SPACE,MESOMERIC,SPACE]
\stopchemical
\startchemical
  \chemical[SIX,B,EB24,ER6,-R1,+R1]
  \chemical[SIX,PB:RZ6,ONE,OFF1,Z0,EP5,ZT7,PE][\SL{OH},\oplus]
  \chemical[SIX,-RZ1,+RZ1][H,Br]
\stopchemical
\startchemical
  \chemical[SPACE,CLOSECOMPLEX]
\stopchemical
\stopformula
```

Without the use of `SPACE` the seperate structures would merge. Most of the time the optimization of such a reaction is an iterative proces.


OH
H
Br
OH
H
Br
OH
H
Br
OH
H
Br

11 Reactions

Not only the typesetting of chemical structures is supported but also the typesetting of normal reactions. The command `\chemical` has three other appearances:

```
\chemical{formula}
\chemical{formula}{bottom text}
\chemical{formula}{top text}{bottom text}
```

This command adapts itself to text mode. That means that it 'knows' whether it is used in:

- text–mode
- mathematical text–mode
- mathematical display–mode

When the command is used in running text it will automatically be surrounded by `$ $`. Typing `\chemical{NH_4^+}` will result in NH_4^+.

The result would be the same if we would place the command between `$ $`. In both cases the second and third argument can be left out. If we place the command between `$$ $$` (or `\startformula` and `\stopformula`) both arguments do have a function. First a simple example. The command `\placeformula` is a ConTeXt command and handles the positioning and numbering of the formula.

```
\placeformula
  \startformula
    \chemical{2H_2} \chemical{PLUS} \chemical{O_2}
    \chemical{GIVES} \chemical{2H_2O}
  \stopformula
```

$$2H_2 \quad + \quad O_2 \qquad \longrightarrow \qquad 2H_2O \tag{11.1}$$

The definition of the chemical part could be somewhat shorter:

```
\chemical{2H_2,PLUS,O_2,GIVES,2H_2O}
```

or even:

```
\chemical{2H_2,+,O_2,->,2H_2O}
```

A TEX–addict will notice from these examples that the plus sign and the arrow are on the baseline. Compare for example + and +. In the reaction you will see that the + and the ⟶ are vertically aligned.

You can use `PLUS`, `GIVES` and `EQUILIBRIUM` (`<->`) in this command. With `MESOMERIC` or `<>` you will get ⟷.

The reaction can be placed in the text. In that case a more compact display is used: $2H_2+O_2 \longrightarrow 2H_2O$. Some finetuning with `\,` would result in $2\,H_2 + O_2 \longrightarrow 2\,H_2O$.

It is also possible to display bonds in textmode. For example if you want H—CH=HC—H you should type `\chemical{H,SINGLE,CH,DOUBLE,HC,SINGLE,H}` or something like this `\chemical{H,-,CH,--,HC,-,H}`. A triple bond can be defined as `TRIPLE` or `---`: HC≡CH.

We return two the display–mode. The second and third argument can be used to add text to the reaction:

```
\placeformula
  \startformula
    \chemical{2H_2}{hydrogen} \chemical{PLUS} \chemical{O_2}{oxygen}
    \chemical{GIVES}{heat} \chemical{2H_2O}{water}
   \stopformula
```

So we can also place text over and under symbols!

$$\underset{\text{hydrogen}}{2H_2} + \underset{\text{oxygen}}{O_2} \quad \underset{\text{heat}}{\longrightarrow} \quad \underset{\text{water}}{2H_2O} \tag{11.2}$$

The last argument is placed under the compound.

$$\overset{\text{liquid}}{\underset{\text{water}}{H_2O}} \quad \text{c.q.} \quad \underset{\text{water}}{H_2O} \tag{11.3}$$

The formula above is defined with:

```
\placeformula
  \startformula
    \chemical{H_2O}{liquid}{water}
    \hbox{c.q.}
    \chemical{H_2O}{water}
  \stopformula
```

The size of the formulas or reactions in the running text can be set up with:

parameter	set up	default
size	small medium big	big

Table 11.1 Set up in text formulas.

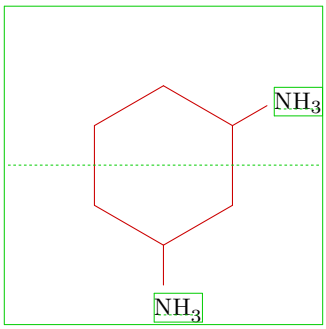
The definition `\chemical{H,SINGLE,CH,DOUBLE,HC,SINGLE,H}` result with `big`, `medium` and `small` in the following formulas:

H—CH=HC—H H—CH=HC—H H—CH=HC—H

12 Subscripts

Sub- and superscripts are placed somewhat lower as is recommended by Knuth in the TEXBook. The rather strange chemical compound that is shown on page 179 of the TEXBook is defined with `\chemical{Fe_2^{+2}Cr_2O_4}`. This will result in $Fe_2^{+2}Cr_2O_4$. Without correction it would have been: $Fe_2^{+2}Cr_2O_4$.

The position of the subscript is determined by the parameter `offset`: `HIGH` or `LOW`. This position can be influenced locally (per substituent) as is shown in the example below.

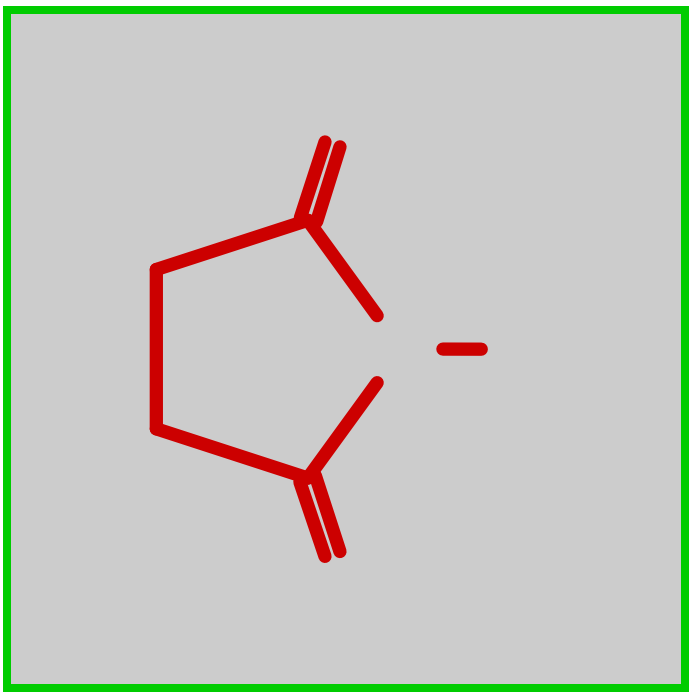


Example 12.1

```
\startchemical[frame=on,option=test,alternative=2]
  \chemical[SIX,B,R13,HIGH,RZ1,LOW,RZ3][NH_3,NH_3]
\stopchemical
```

However, it is recommended to set up this parameter globally to obtain consistent formulas.

The values `LOW` and `HIGH` can also be used in text formulas. For example if you type `\chemical{HIGH,H_2O}` then you will get H_2O and `\chemical{LOW,H_2O}` will become H_2O.



Part 2
Backgrounds

1 Installation

The package PPCHTEX is developed for use in combination with CONTEXT. PPCHTEX is activated in CONTEXT by:[1]

```
\usemodule[pictex,chemic]
```

This command loads the PICTEX macros so PPCHTEX knows what output is needed. The chemical macros are not automatically available.

The package can be used in combination with PLAIN TEX or LATEX. In that case the file `m-chemie.sty` is also used. PPCHTEX is then activated by `\documentstyle`:

```
\documentstyle[m-pictex,m-chemie]{}
```

In LATEX2$_\varepsilon$ it is somewhat different:

```
\usepackage{m-pictex}
\usepackage{m-chemie}
```

The file `m-pictex` takes care of an efficient loading of PICTEX. This is necessary because LATEX allocates a lot of `\dimen`s. Because of the userinterface a big version of TEX is needed to run PPCHTEX.

PPCHTEX can be used in three languages. The actual language can be activated by:

language	**files**
dutch	`m-che-nl` = `m-chemie.sty`
english	`m-che-en` = `m-chemic.sty`
german	`m-che-de` = `m-chemie.sty`

In the file `ppchtex.noc` you can see the coupling between CONTEXT and macropackage. This file also loads the system modules of CONTEXT.

The total distribution consists of the definition files `ppchtex.tex` and `ppchtex.noc` the starting files `m-che-nl.tex`, `m-che-en.tex` and `m-che-de.tex` and the alternative starting files `m-chemie.tex` and `m-chemic.tex`.

[1] The macros in file `ppchtex.tex`, are loaded by typing `m-chemic.tex` (the `m` stands for module).

2 Extensions

Users of PPCHTEX are free to use and alter the macros. However one should be careful because most macros are stil under development. Some macros may look more complex than necessary, but this is due to the userfriendly interface of CONTEXT and PPCHTEX.

Commands like `\setup...` make it possible to make readable ASCII–layouts. Compare for example:

```
\setupchemical[size=small]
```

and:

```
\setupchemical
  [size=small,
   scale=500,
   textsize=big]
```

The set up can be defined in a random order and newlines and spaces are not interpreted.

Originally PPCHTEX was meant to be a module in CONTEXT, therefore the package is multi lingual.

If you study the file `ppchtex.tex` you may notice that `\processaction` macros are being used while interpretating the keys in `\chemical[ ]`. These kind of macros are relatively slow but then the PPCHTEX interface is very flexible.

3 Fonts

The macros are in mathematical mode and therefore use `\textfont`, `\scriptfont` and `\scriptscriptfont`. When needed the `\fontdimen`s 14, 16 and 17 of `\font2` are adapted. The size of the actual font is derived from x height (`\fontdimen5`).

Changes in `\fontdimen` s have are global, so grouping makes no sense. The dimensions are therefore continually set and reset. This solution may seem poor but alternatives are not failsave and will result in problems with scaled fonts.

Typesetting atoms and molecules (text) during processing are rather time consuming. Speed depends on the complexity of the macro `\rm`.

4 Definitions

The interface of PPCHTEX is derived from the CONTEXT interface. This means that the interface is multi langual. The advantage is that one can use PPCHTEX in his or her own language. The disadvantage is the fact that macros have to be shared between languages.

At this moment the CONTEXT commands and parameters are dutch. PPCHTEX however has english commands.

```
\startchemical
  \chemical[SIX,B,C]
\stopchemical
```

Set ups are more difficult. We use system constants and variables that are dutch. In due time these will be translated. Adaption of these constants and variables is not too difficult.

```
\setupchemical[\c!breed=10cm,\c!height=\v!passend]
```

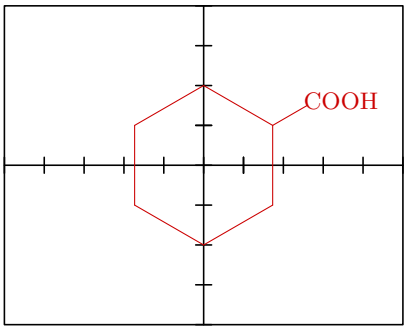
Parameters are preceded by `\c!` and set ups by `\v!`. This only works when `!` is a character. That is the reason that these set ups have to be surrounded by `\unprotect` and `\protect`. For example:

```
\unprotect

\setupchemical[\c!width=10cm,\c!height=\v!passend]

\startchemical
  \chemical[SIX,B,C]
\stopchemical

\protect
```

More information on the interface can be found in the documentation of the CONTEXT modules from the `mult` group.

5 Color

In CONTEXT you can colorize parts of a structure. In **example 5.1** the substituent as well as the bond are colored red.



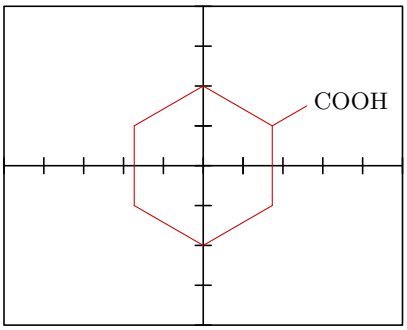
Example 5.1

```
\startchemical[axis=on,frame=on,width=5000]
  \chemical[SIX,B]
  \color[red]{\chemical[SIX,R1,RZ1][COOH]}
\stopchemical
```

First the color mechanism has to be activated by `\setupcolors[state=start]`.

6 Interaction

In combination with CONTEXT PPCHTEX supports interactive texts. An interactive text is a text that can be consulted on a computerscreen and contains many hyperlinked textareas. This means that clicking on such an area will result in a jump to the target area.



Example 6.1

```
\startchemical[axis=on,frame=on,width=5000]
  \chemical[SIX,B]
  \chemical[sub:cooh][SIX,R1,RZ1][COOH]
\stopchemical
```

We see a new argument: the reference `[sub:cooh]`. This means that we can refer from the text COOH to the structure with:

```
... text ... \goto{\chemical{COOH}}[sub:cooh] ... text ...
```

In this definition `\goto` is a CONTEXT–command. We can also refer from the structure to a particular part of the text.

Clicking in COOH in the structure results in a jump to the text that is marked with:

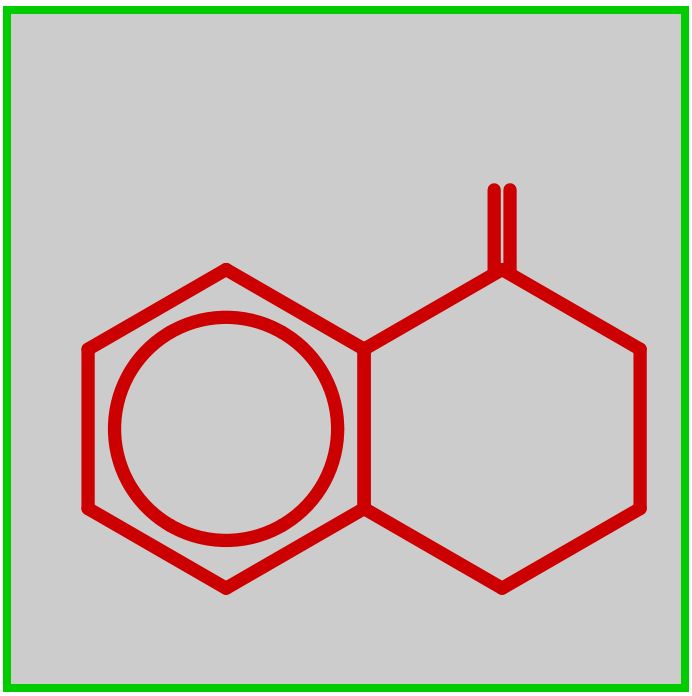
```
\paragraph[txt:cooh]{Substituents}

... text ... \chemical{COOH} ... text ...
```

A combination is also possible. In that case it is necessary to mark the reference with `\chemical` and to refer in the text with `\gotochemical`.

The coupling of the interaction mechanism is done with macros:

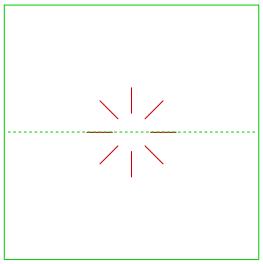
```
\localgotochemical   {reference} {text}
\localthisischemical {reference}
```

You can see the CONTEXT sources for more information.

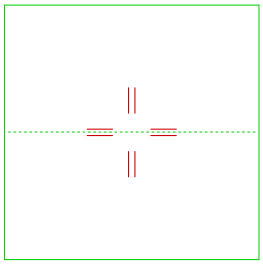


Part 3
Overview

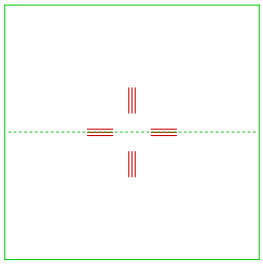
1 One



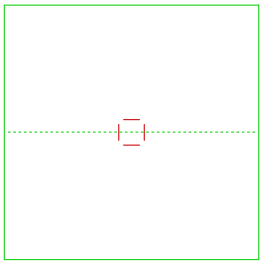
SB



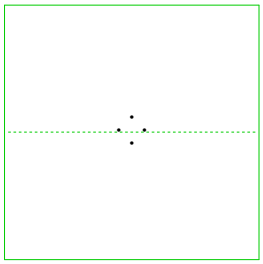
DB



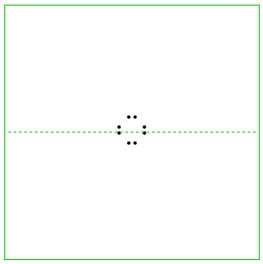
TB



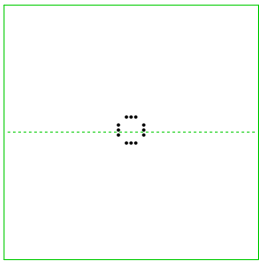
EP



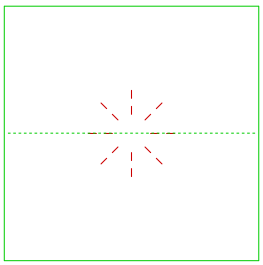
ES



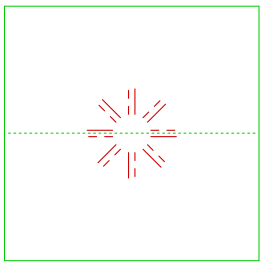
ED



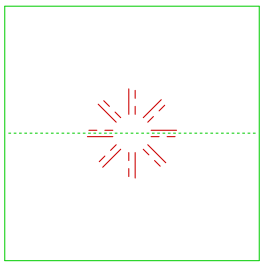
ET



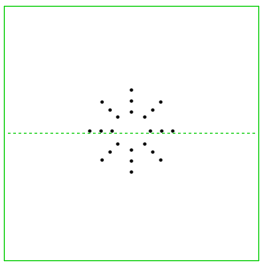
SD



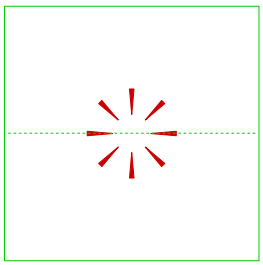
LDD



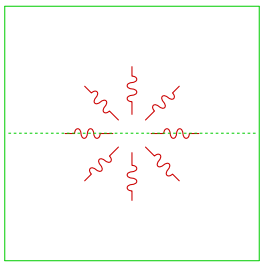
RDD



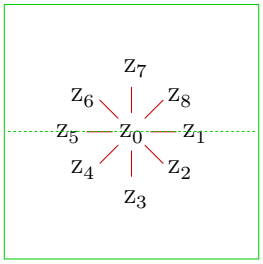
HB



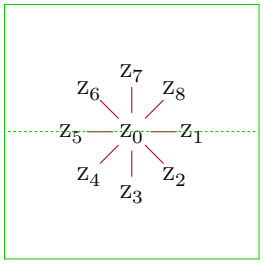
BB



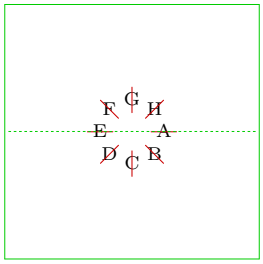
OE



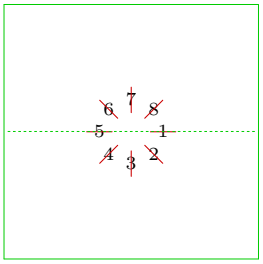
Z



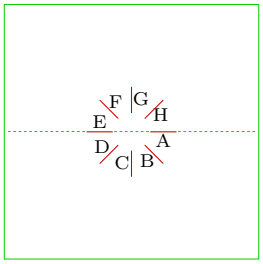
CZ



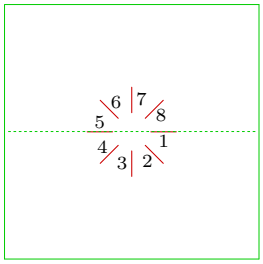
ZT



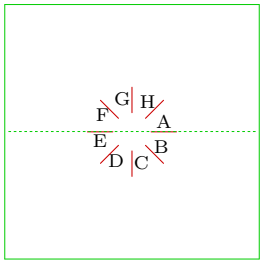
ZN



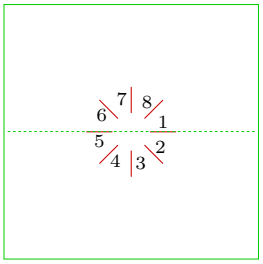
ZBT



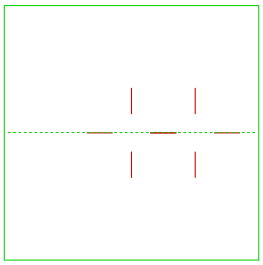
ZBN



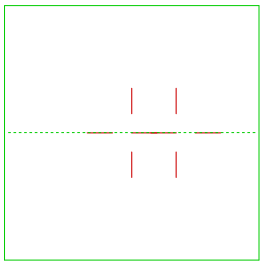
ZTT



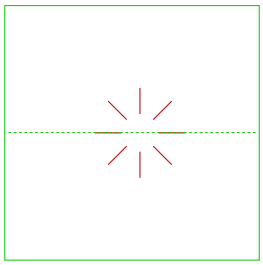
ZTN



MOV

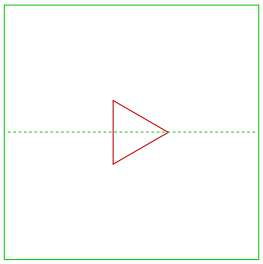


DIR

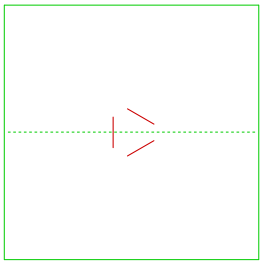


OFF

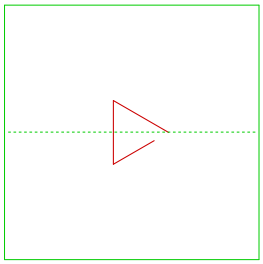
2 Three



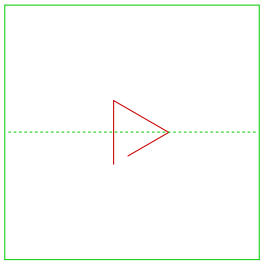
B



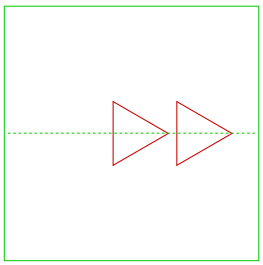
SB



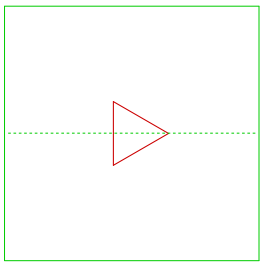
-SB



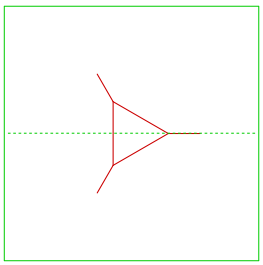
+SB



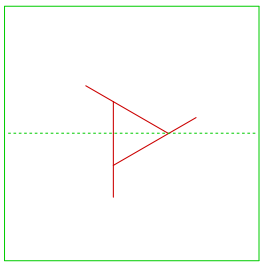
MOV



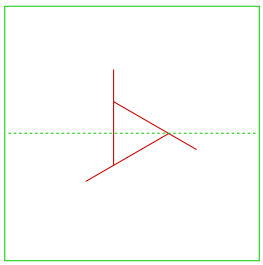
ROT



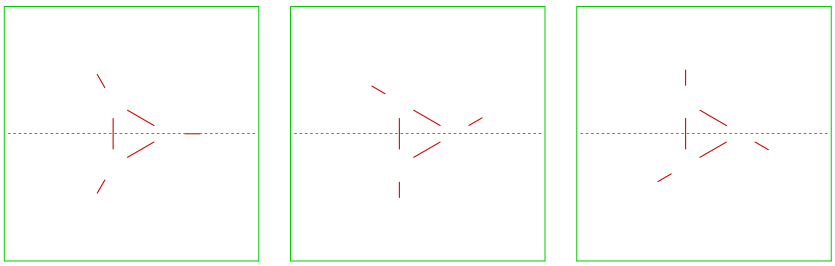
R



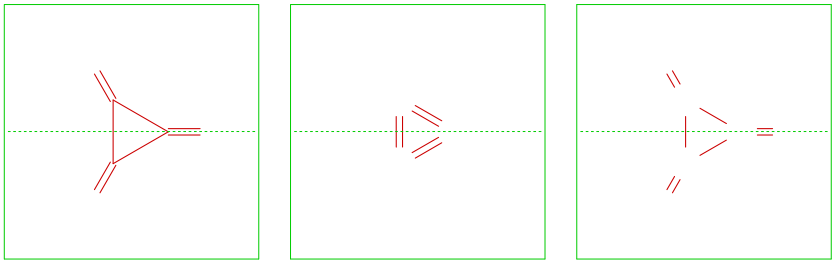
-R



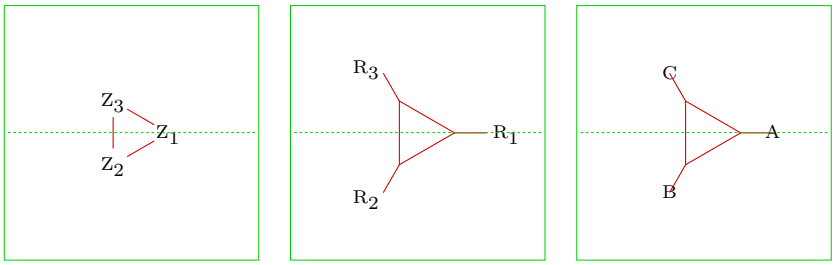
+R



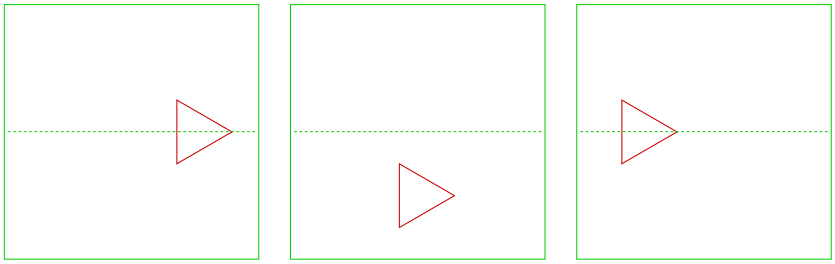
SR -SR +SR



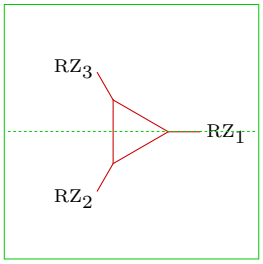
ER DB DR



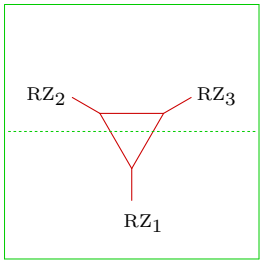
Z RZ CRZ



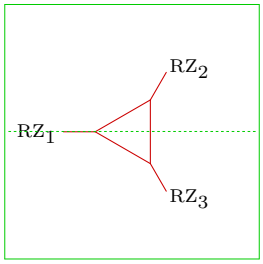
MOV1 MOV2 MOV3



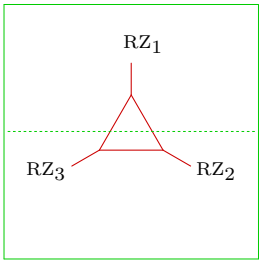
ROT1



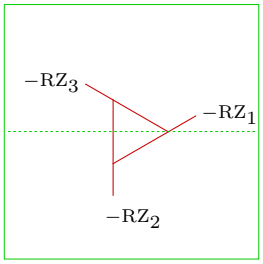
ROT2



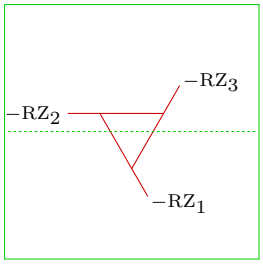
ROT3



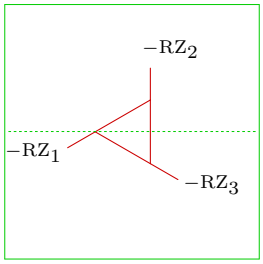
ROT4



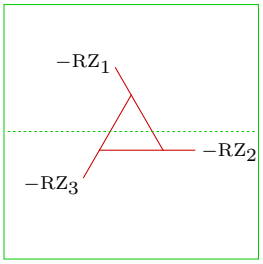
ROT1



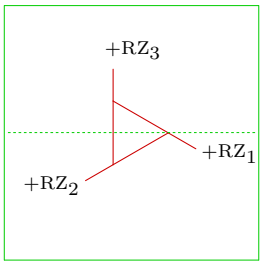
ROT2



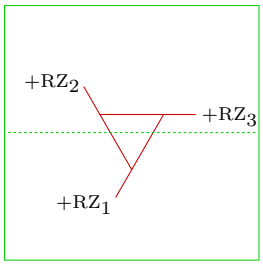
ROT3



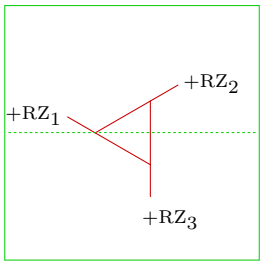
ROT4



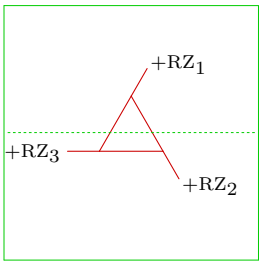
ROT1



ROT2

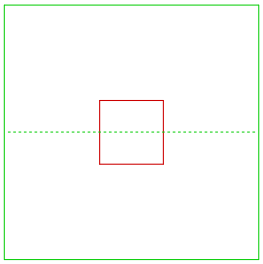


ROT3

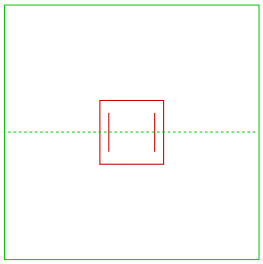


ROT4

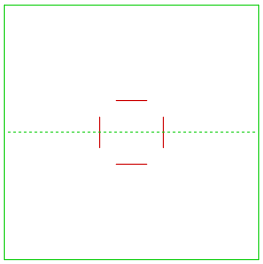
3 Four



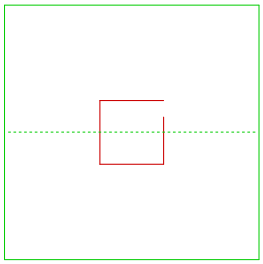
B



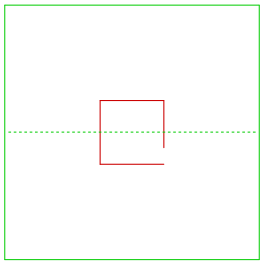
EB



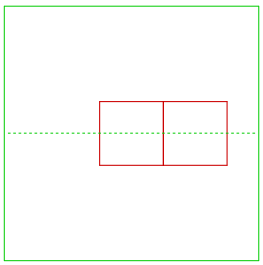
SB



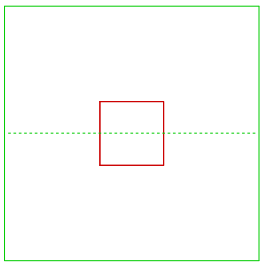
-SB



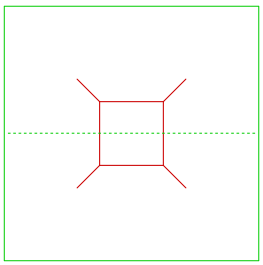
+SB



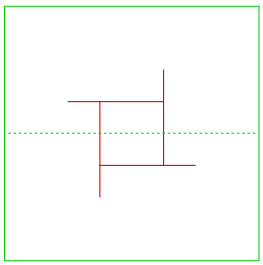
MOV



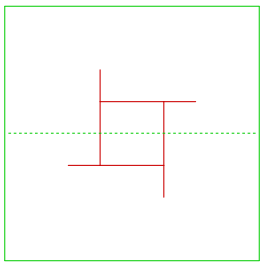
ROT



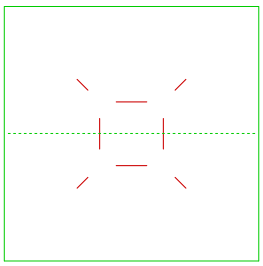
R



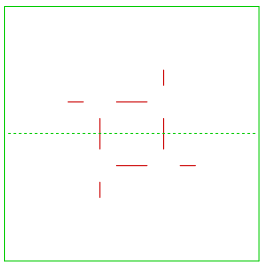
-R



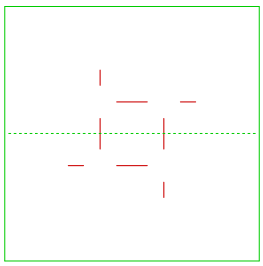
+R



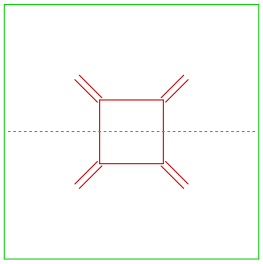
SR



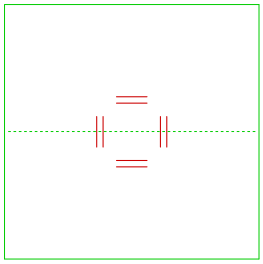
-SR



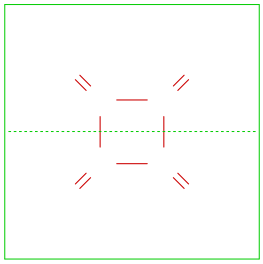
+SR



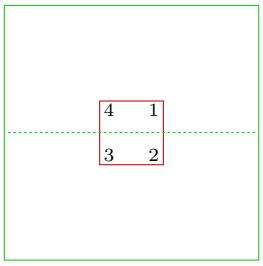
ER



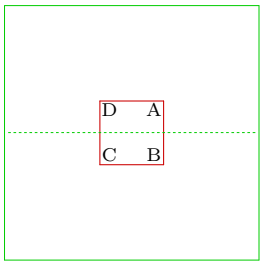
DB



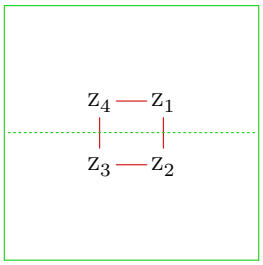
DR



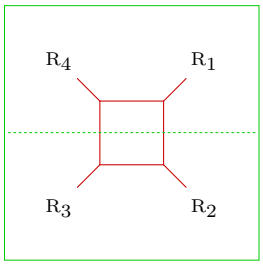
ZN



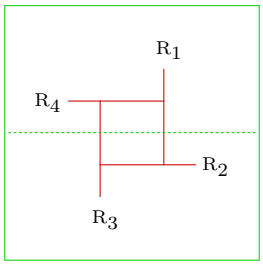
ZT



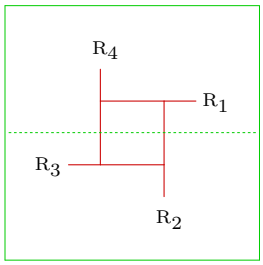
Z



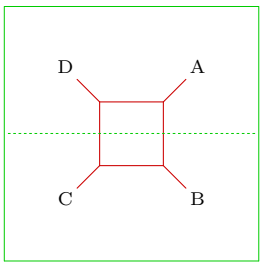
RZ



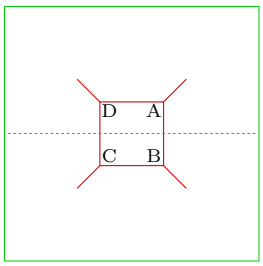
-RZ



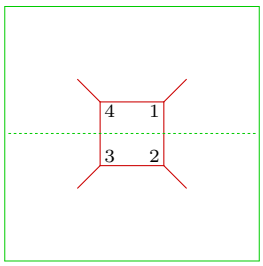
+RZ



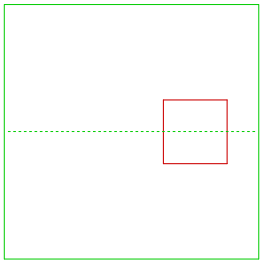
CRZ



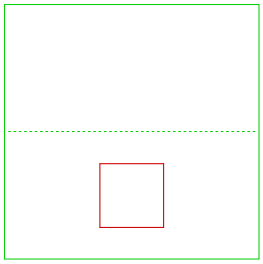
ZT



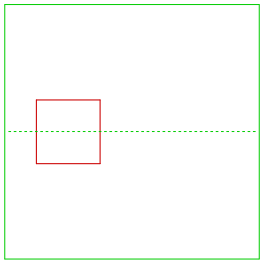
ZN



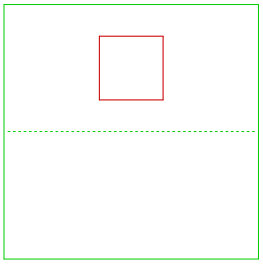
MOV1



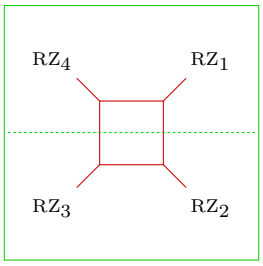
MOV2



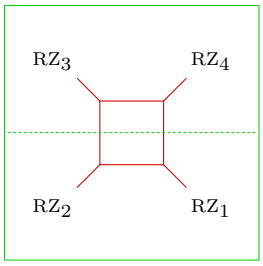
MOV3



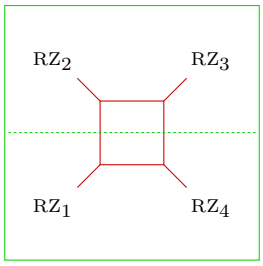
MOV4



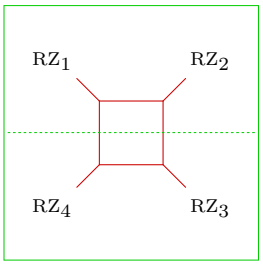
ROT1



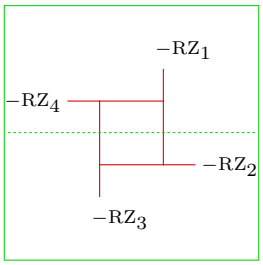
ROT2



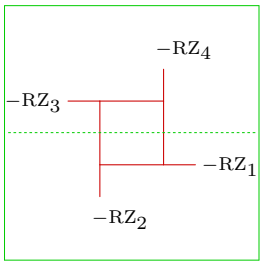
ROT3



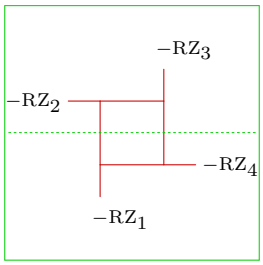
ROT4



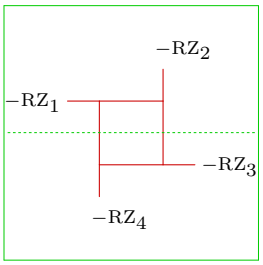
ROT1



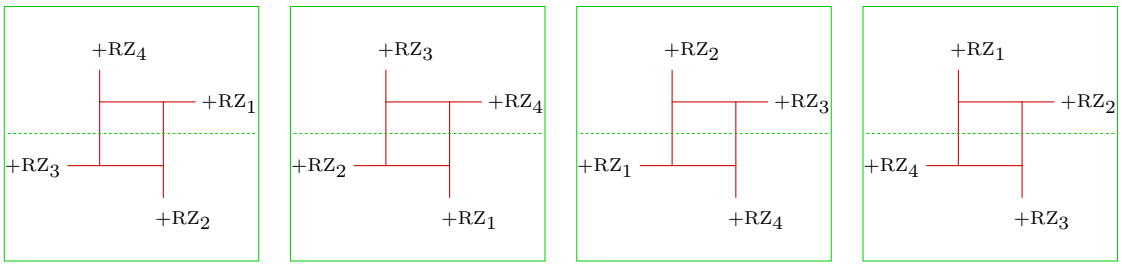
ROT2



ROT3

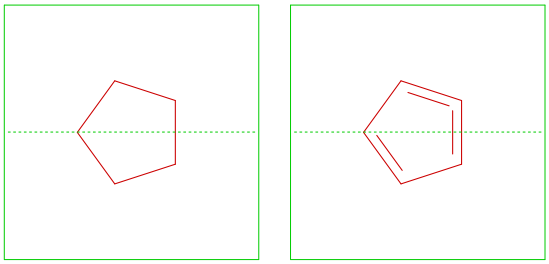


ROT4

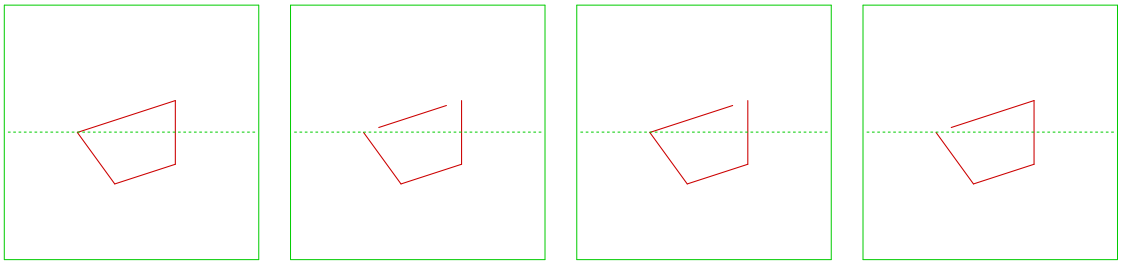


ROT1 ROT2 ROT3 ROT4

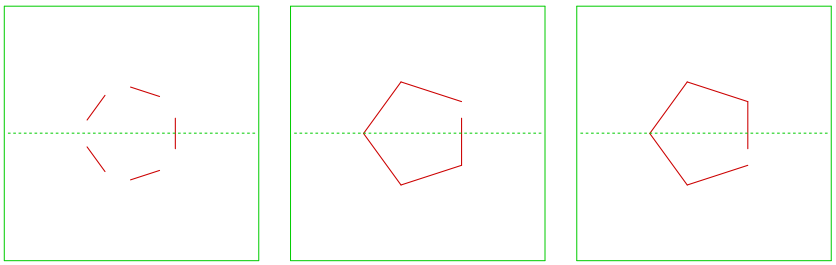
4 Five



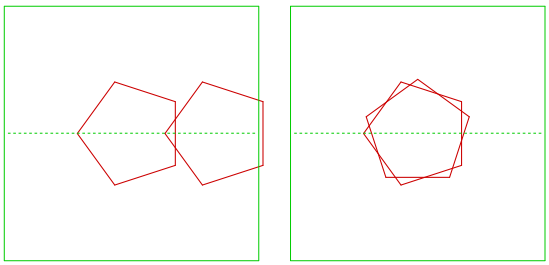
B EB



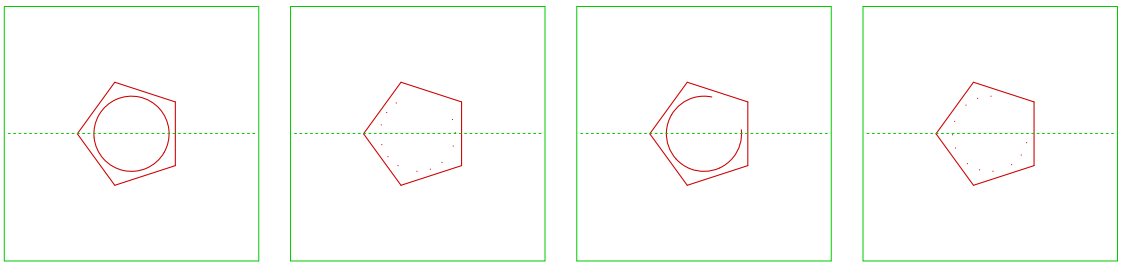
S SS -SS +SS



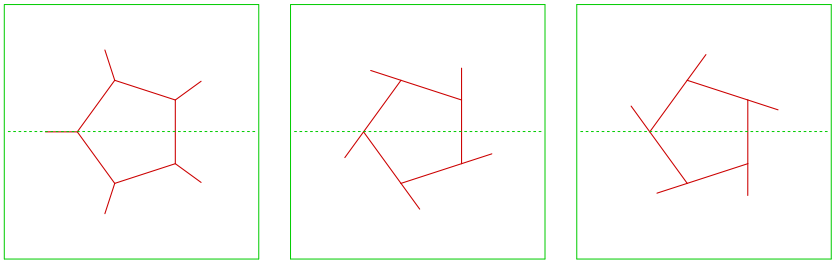
SB -SB +SB



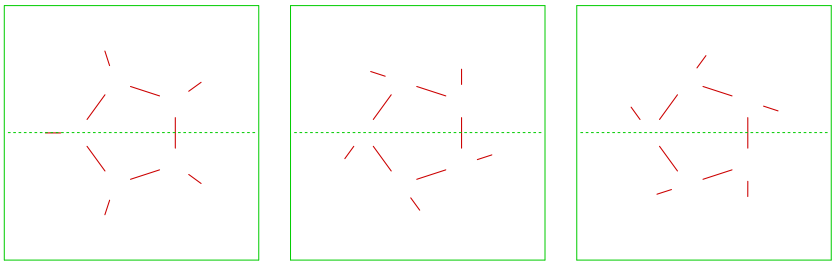
MOV ROT



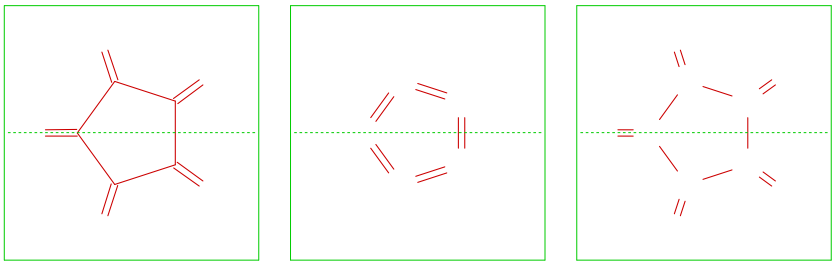
C CD CC CCD



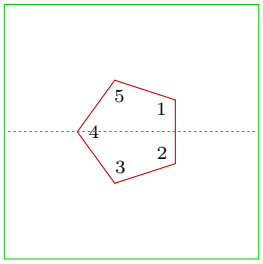
R -R +R



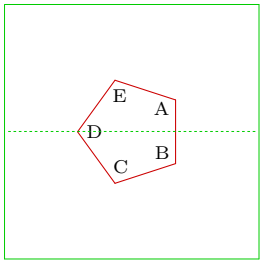
SR -SR +SR



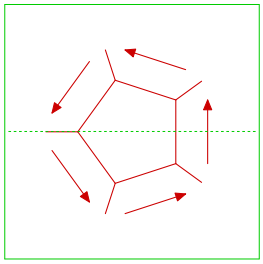
ER DB DR



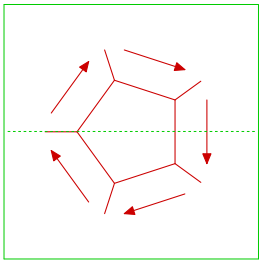
ZN



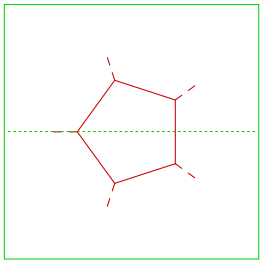
ZT



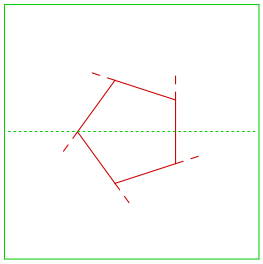
AU



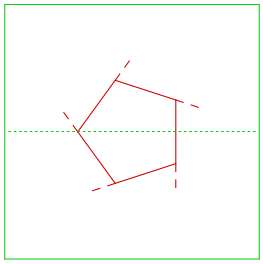
AD



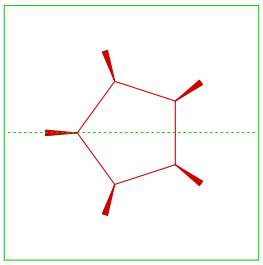
RD



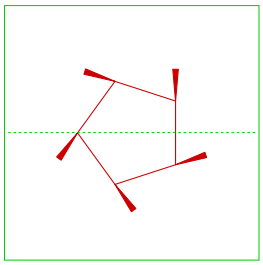
−RD



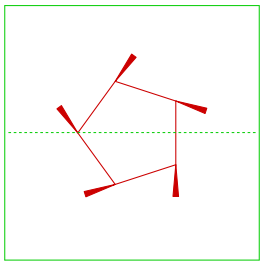
+RD



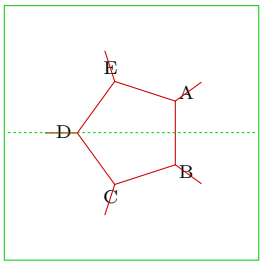
RB



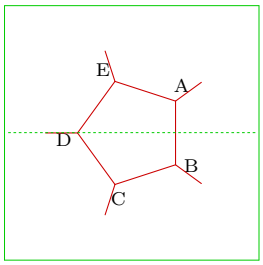
−RB



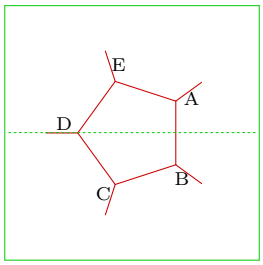
+RB



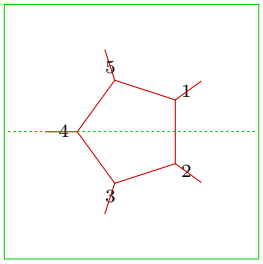
RT



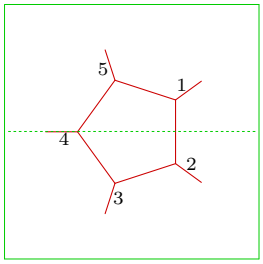
RTT



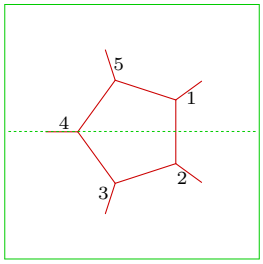
RBT



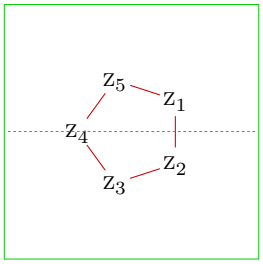
RN



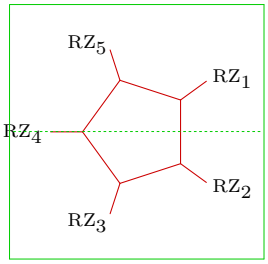
RTN



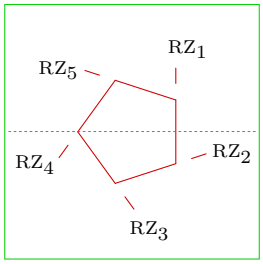
RBN



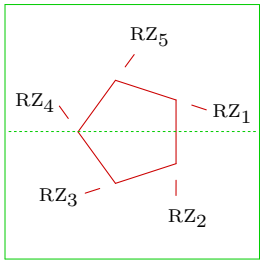
Z



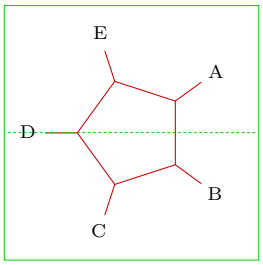
RZ



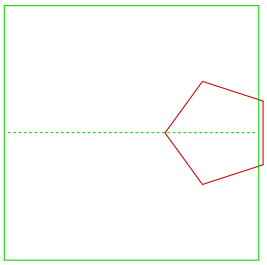
-RZ



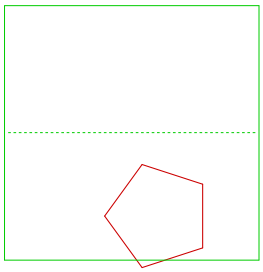
+RZ



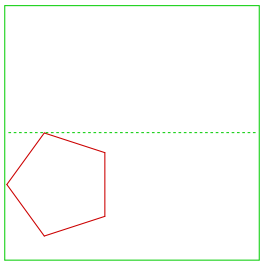
CRZ



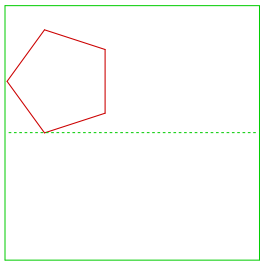
MOV1



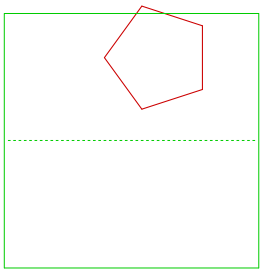
MOV2



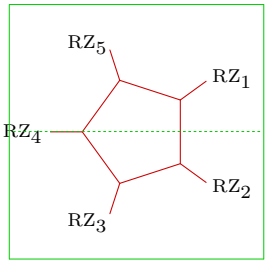
MOV3



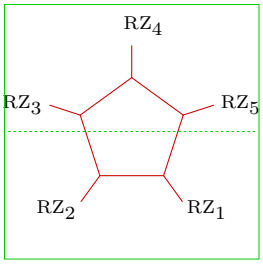
MOV4



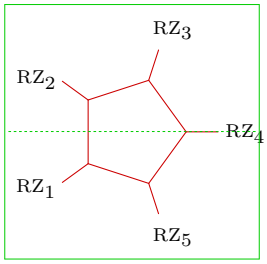
MOV5



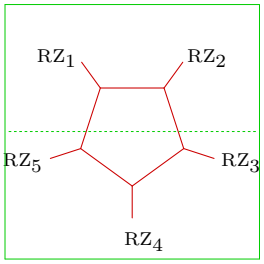
ROT1



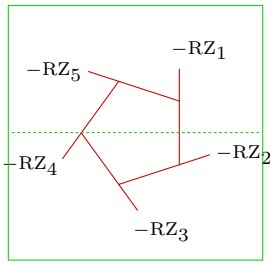
ROT2



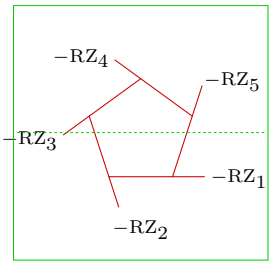
ROT3



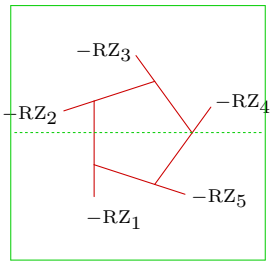
ROT4



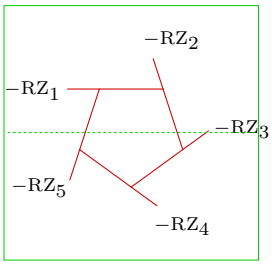
ROT1



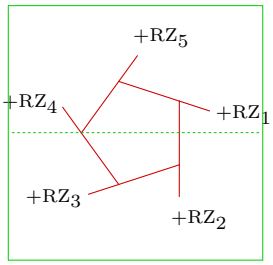
ROT2



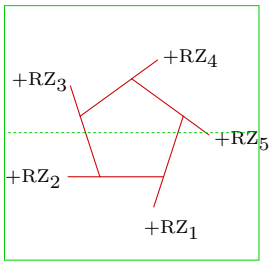
ROT3



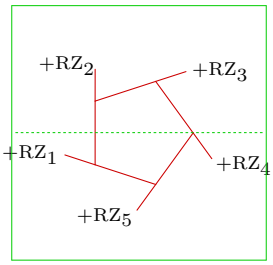
ROT4



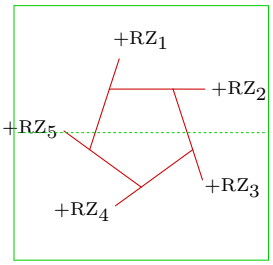
ROT1



ROT2

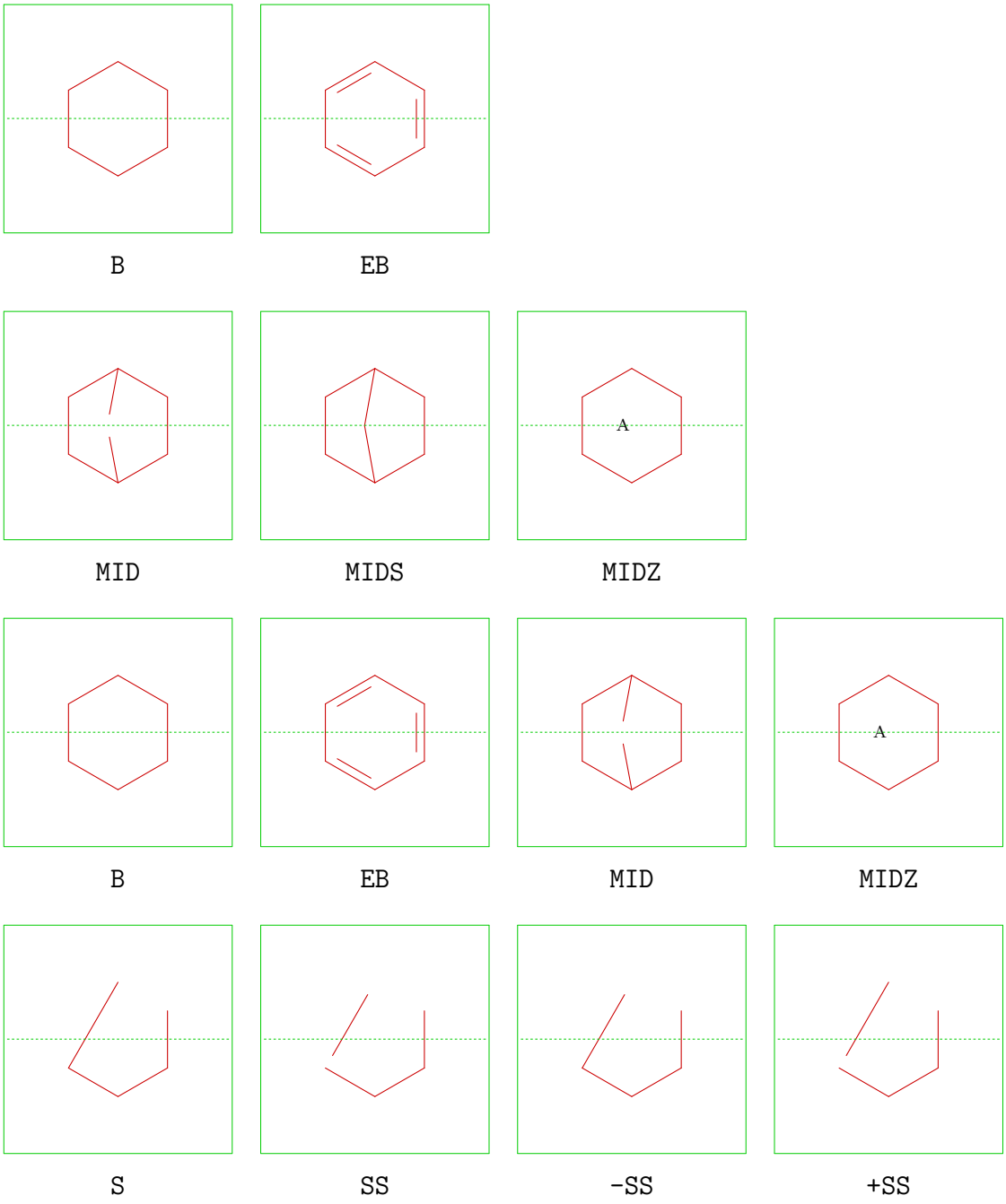


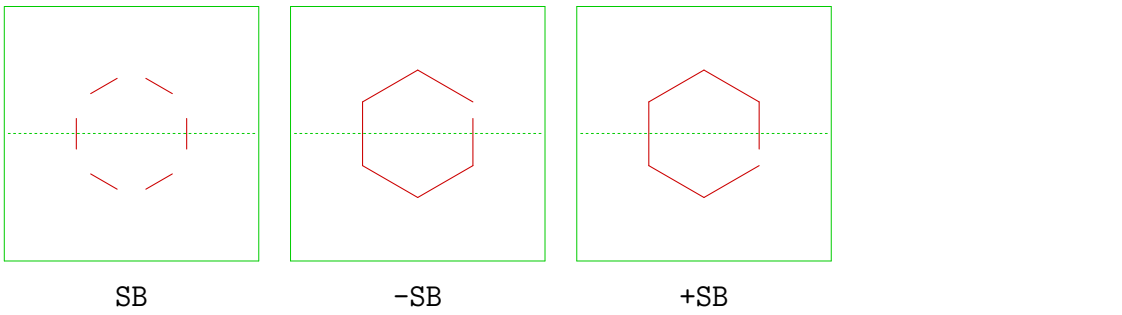
ROT3

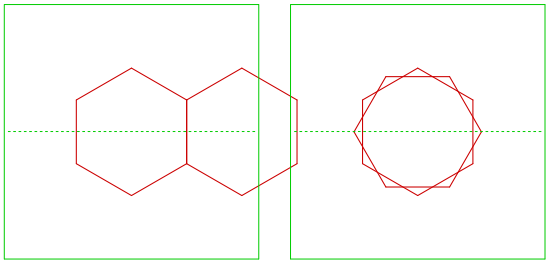


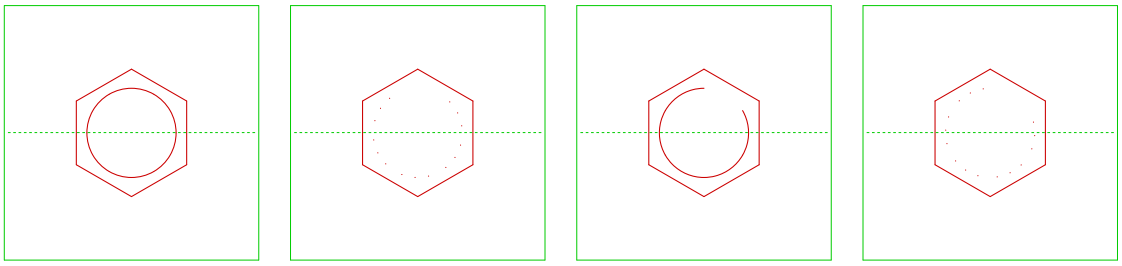
ROT4

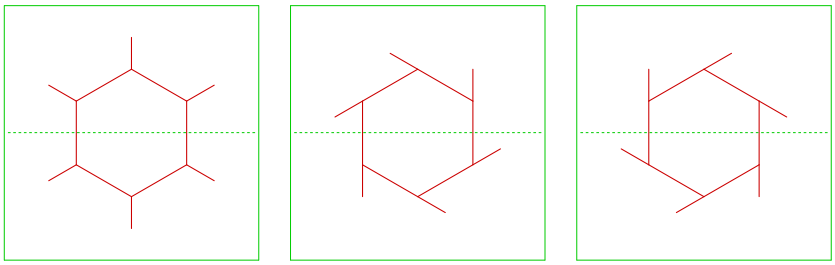
5 Six

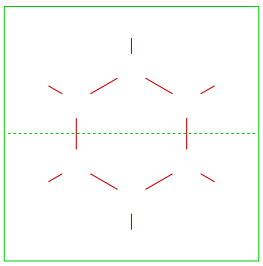



SB
-SB
+SB

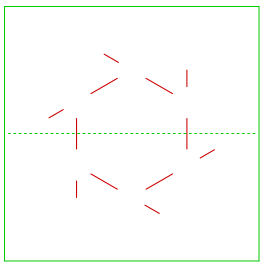

MOV
ROT


C
CD
CC
CCD

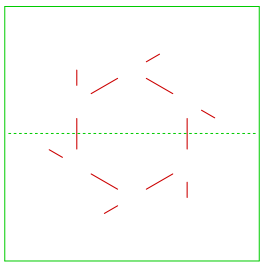

R
-R
+R



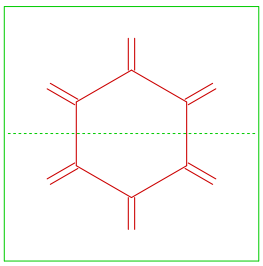
SR



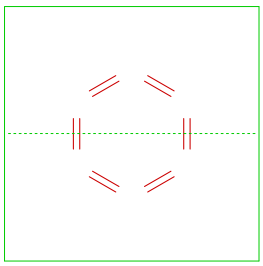
-SR



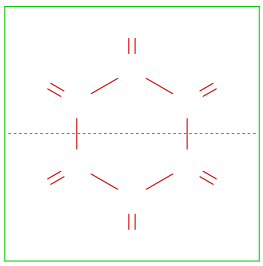
+SR



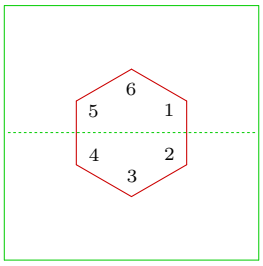
ER



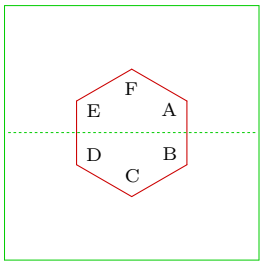
DB



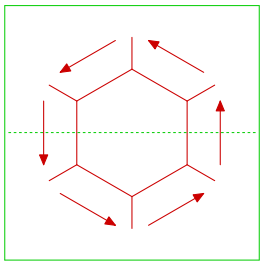
DR



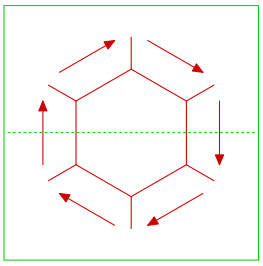
ZN



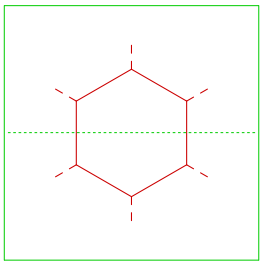
ZT



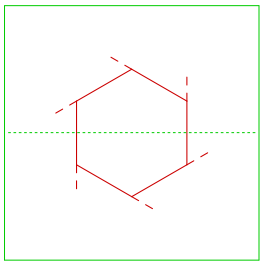
AU



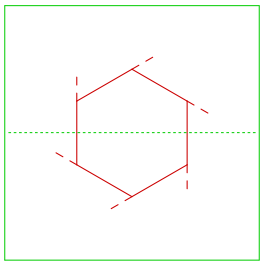
AD



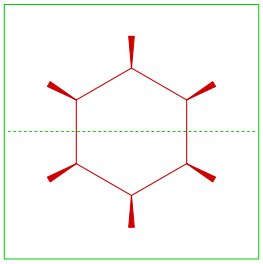
RD



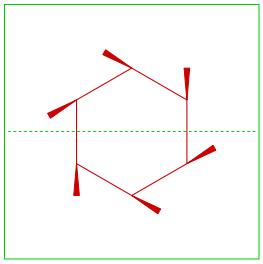
-RD



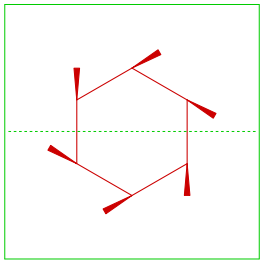
+RD



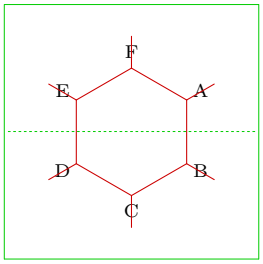
RB



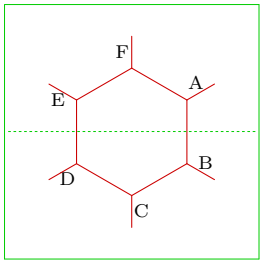
-RB



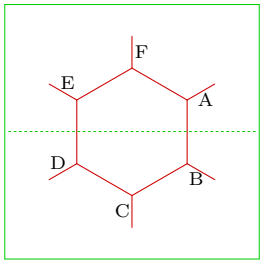
+RB



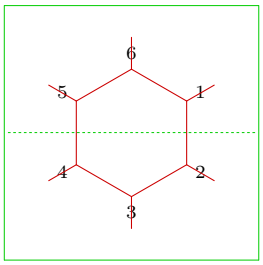
RT



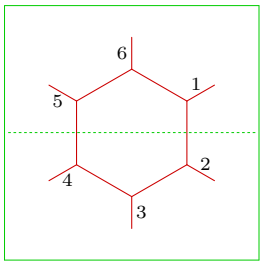
RTT



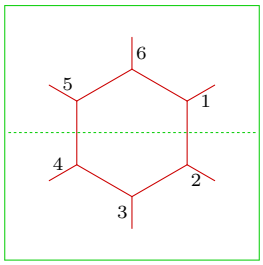
RBT



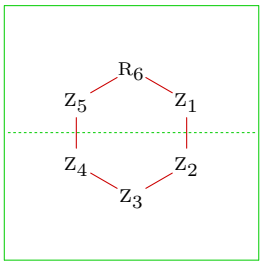
RN



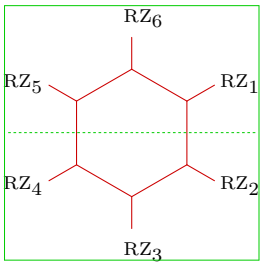
RTN



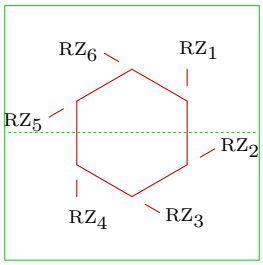
RBN



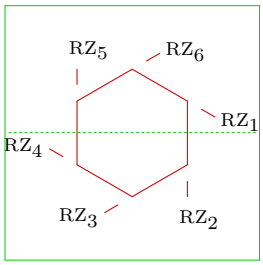
Z



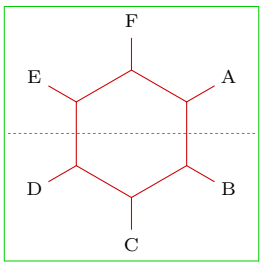
RZ



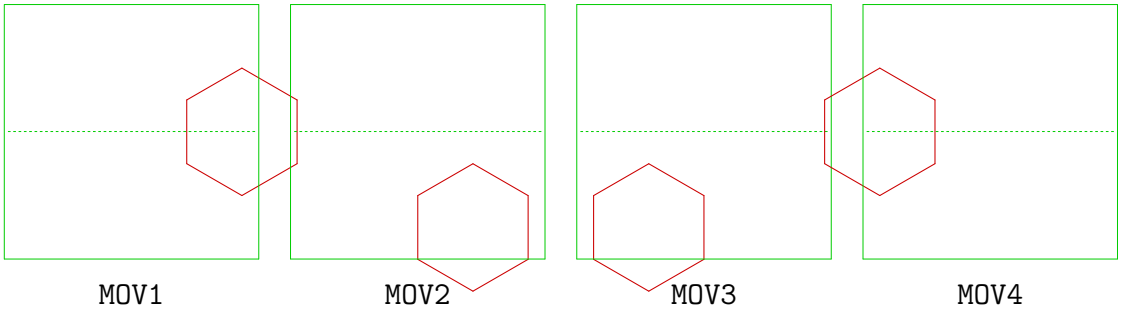
-RZ



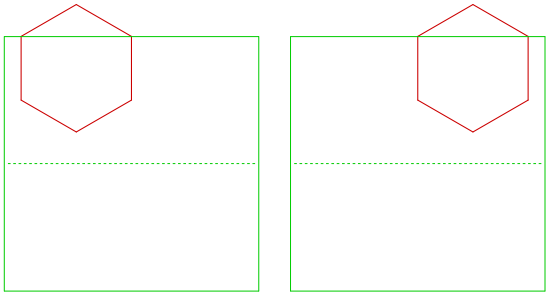
+RZ



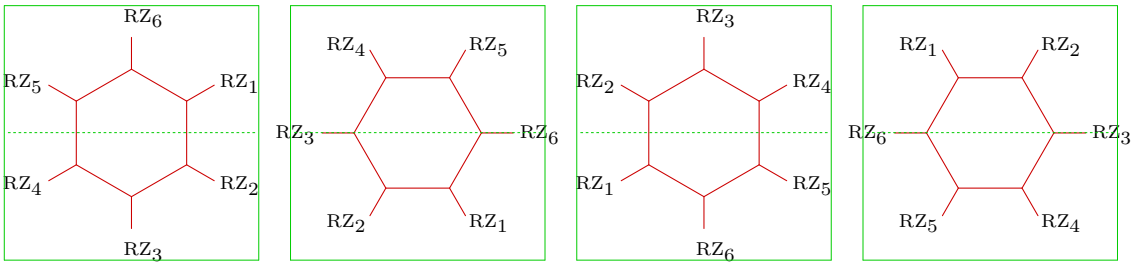
CRZ



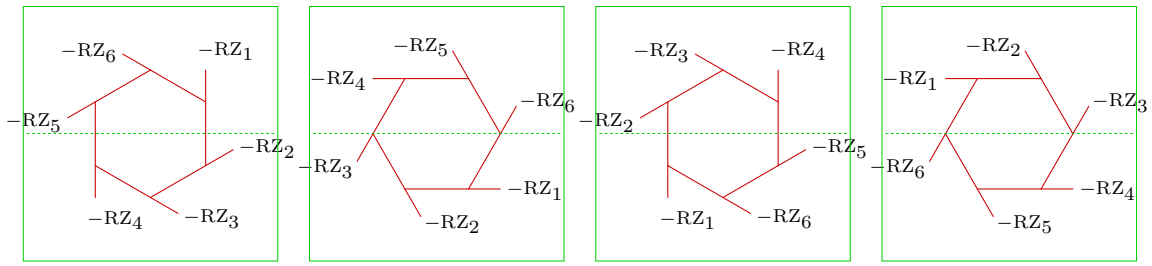
MOV1 MOV2 MOV3 MOV4



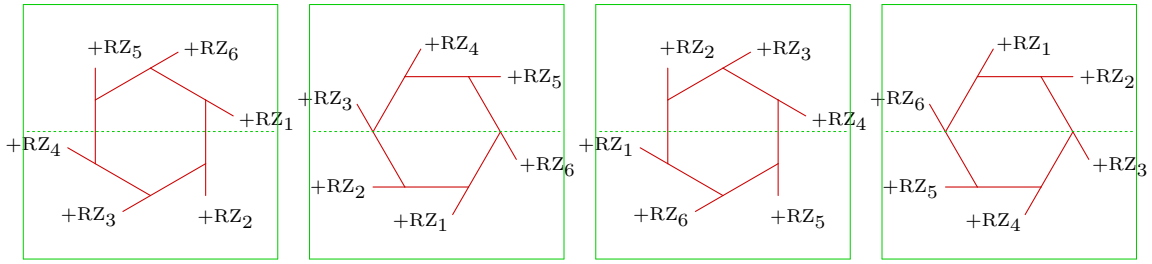
MOV5 MOV6



ROT1 ROT2 ROT3 ROT4

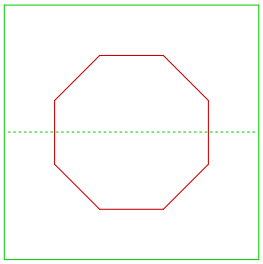


ROT1 ROT2 ROT3 ROT4

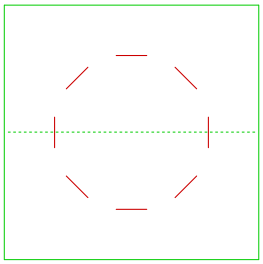


ROT1 ROT2 ROT3 ROT4

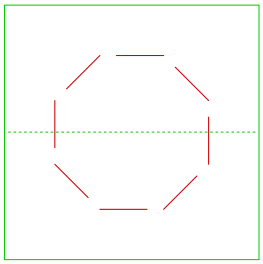
6 Eight



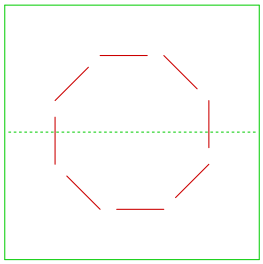
B



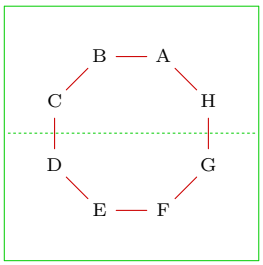
SB



-SB

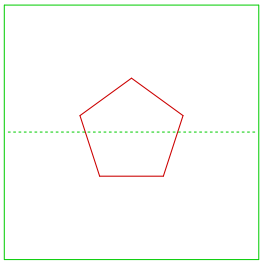


+SB

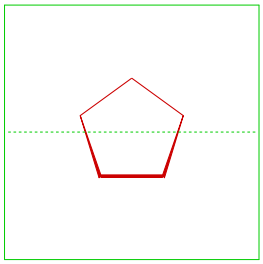


Z

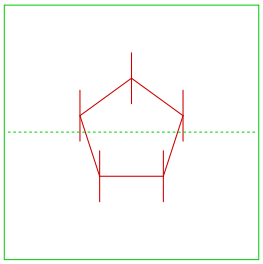
7 Five Front



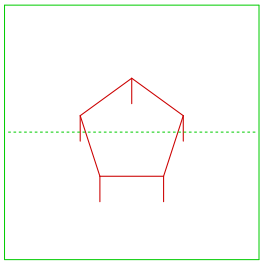
B



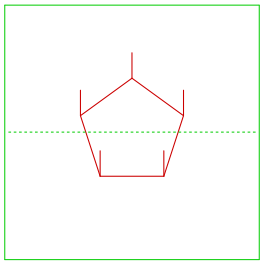
BB



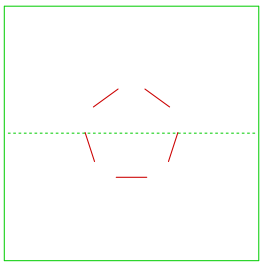
R



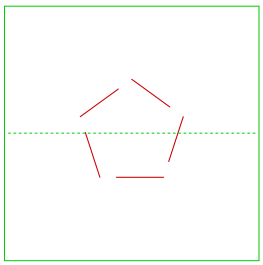
-R



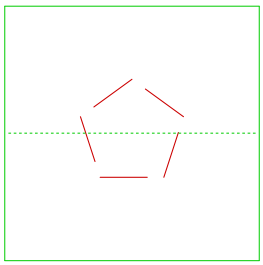
+R



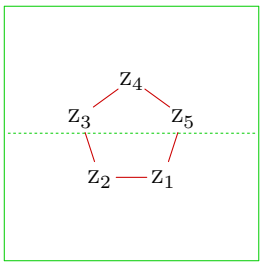
SB



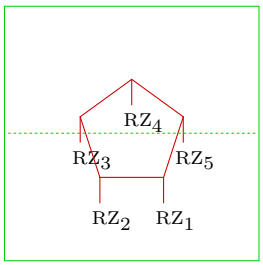
-SB



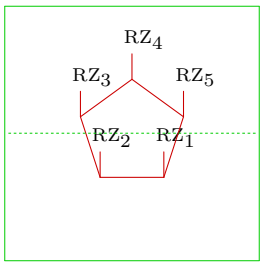
+SB



Z

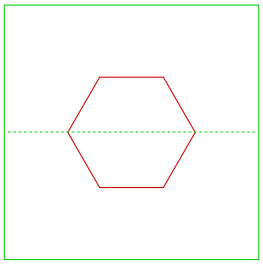


-RZ

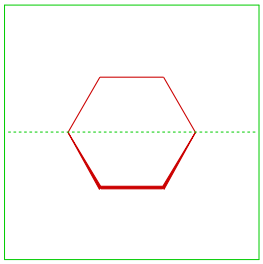


+RZ

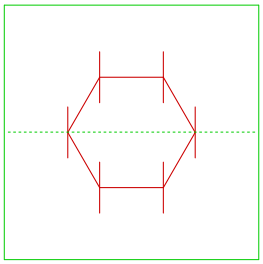
8 Six Front



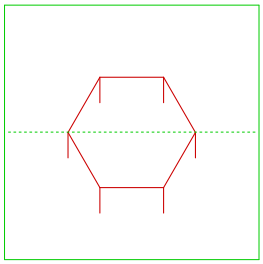
B



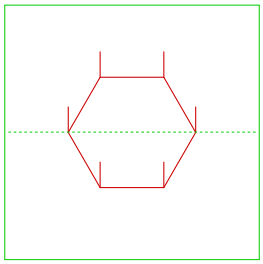
BB



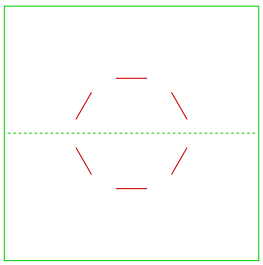
R



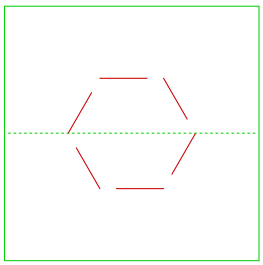
-R



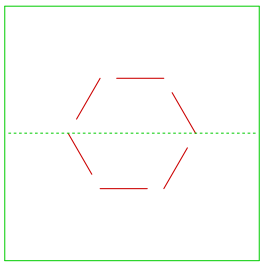
+R



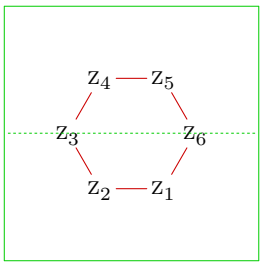
SB



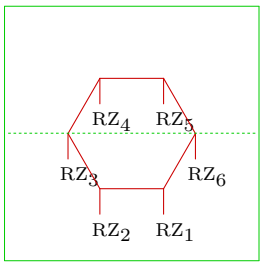
-SB



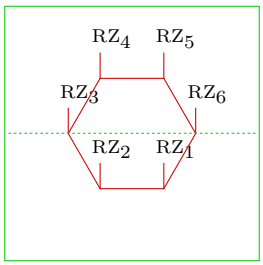
+SB



Z

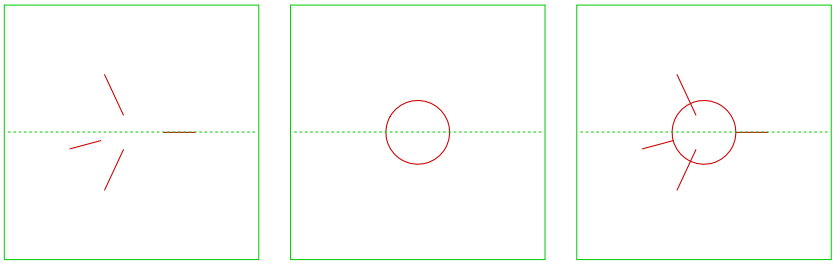


-RZ

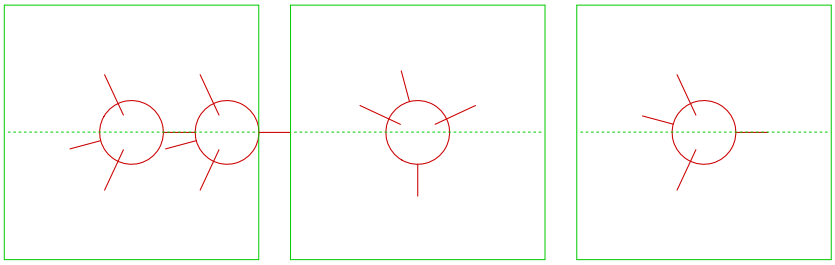


+RZ

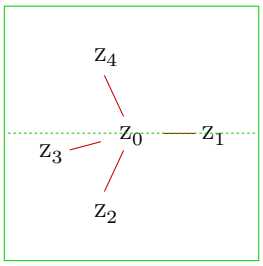
9 Carbon



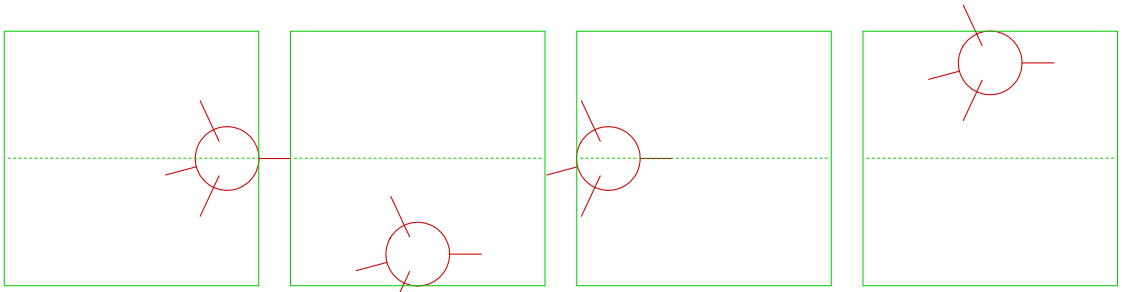
B C CB



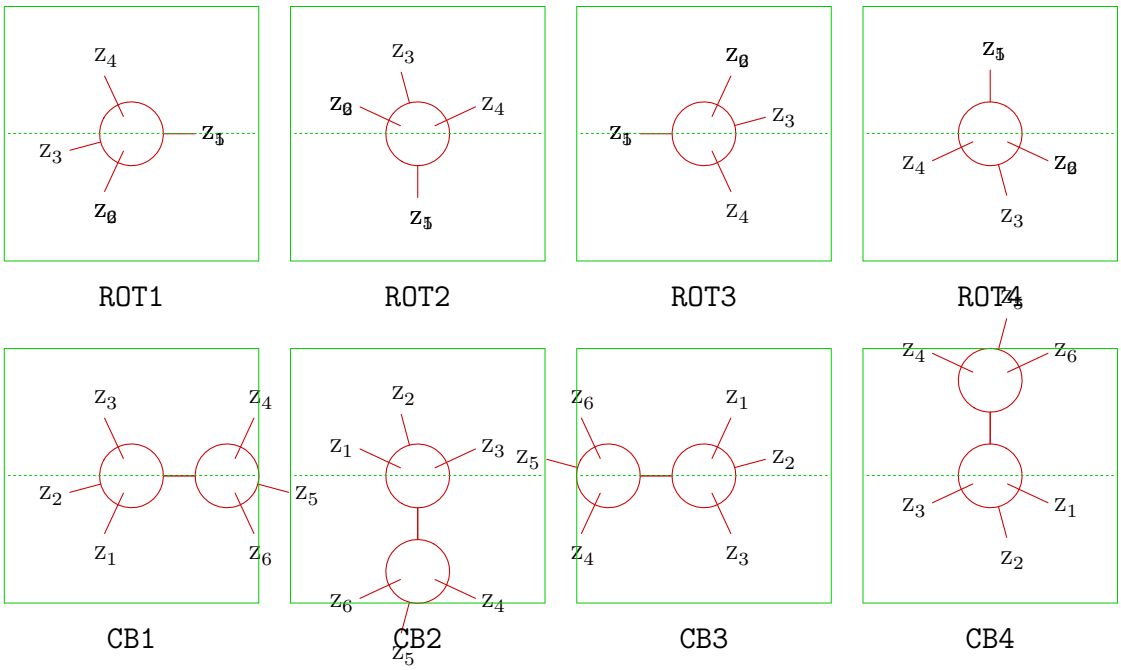
MOV ROT MIR



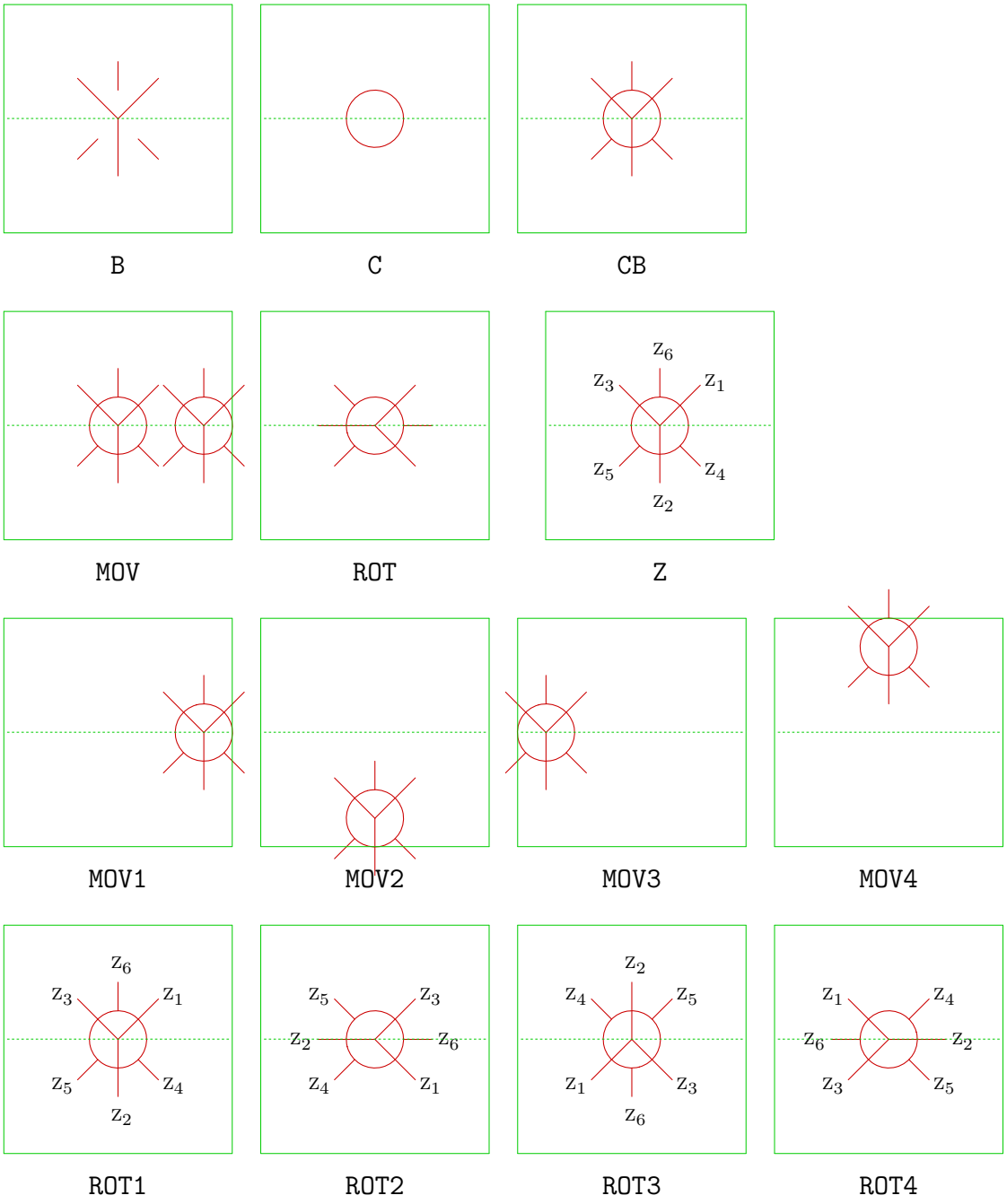
Z



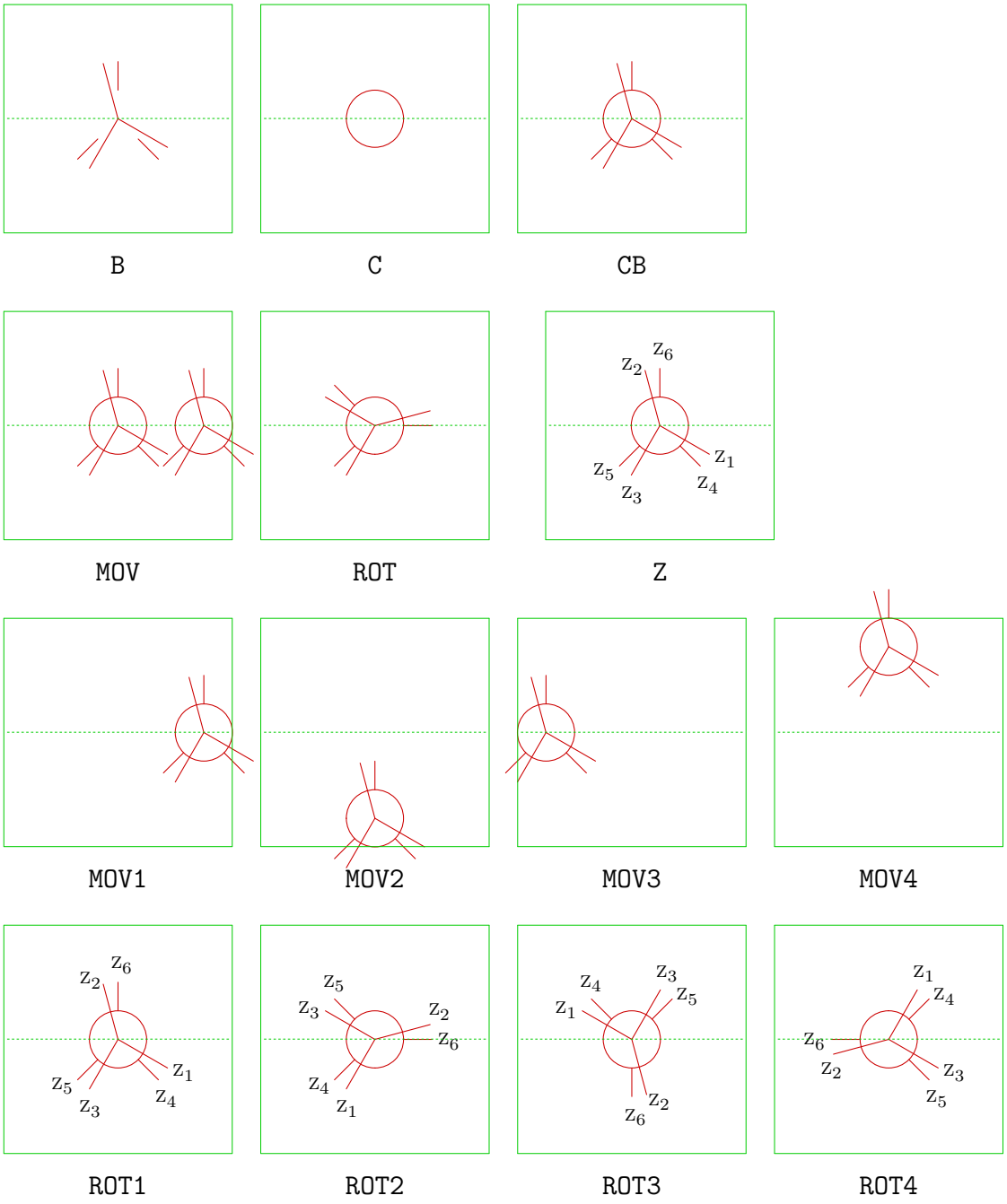
MOV1 MOV2 MOV3 MOV4


ROT1
ROT2
ROT3
ROT4
CB1
CB2
CB3
CB4

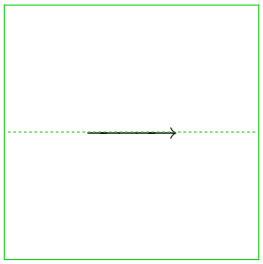
10 Newman Stagger



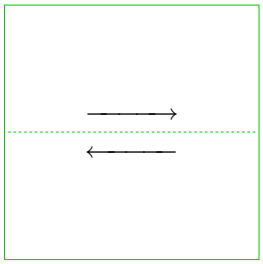
11 Newman Eclipse



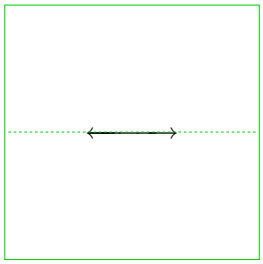
12 Symbol



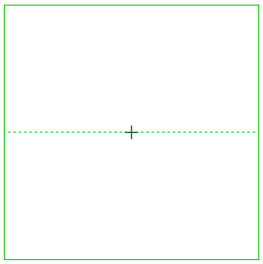
GIVES



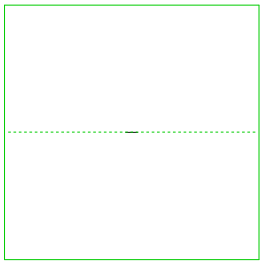
EQUILIBRIUM



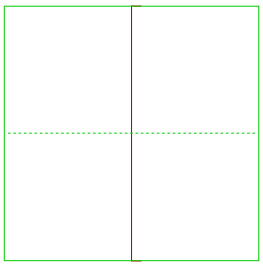
MESOMERIC



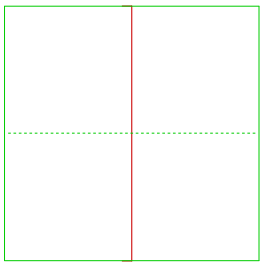
PLUS



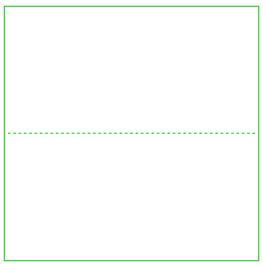
MINUS



OPENCOMPLEX



CLOSECOMPLEX



SPACE

Bijlage 18
The DJGPP port of web2c

Gilbert van den Dobbelsteen
gilbert@login.iaf.nl

abstract

Web2c is getting popular and popular. Why? Because it is widespread. I'll discuss some of the aspects of the port to the MS-DOS environment. I used the variant on the 4AllTEX4 CD-ROM.

Introduction

So what's so good about web2c[1]? Since I didn't know, I decided to find out. I know nothing about web2c — I still don't — except its wide-spread use on Linux systems and other unix based operating systems. The system is ported to win32 (win95 and NT) and various other platforms. There is also a win95/NT and a DOS port. This article focusses mainly on the DOS port.

Since I use emTEX, I was interested in getting a combination of things. This involved the usual TEX programs from the web2c distribution, and the dviscr viewer from emTEX. I also needed dvihplj because it's a good program which I know.

What is DJGPP?

DJGPP is a port of the GNU C/C++ compiler and development tools to the 32-bit protected mode environment on Intel 32-bit CPUs running MS-DOS and compatible operating systems. The port was made by D.J. Delorie and friends. Early versions of the compiler needed extenders to run the generated executables. With the coming of version 2.0 DJGPP programs do not need a separate extender program, only a DPMI server to run. DJGPP includes a free 32-bit DPMI server so you don't have to worry about that.

Extenders and DPMI

So now you say 'what is an extender?' I can imagine that question. Traditionally DOS (in specific MS-DOS) is a 16 bit operating system[2] with no 32-bit support at all. So how can you run 32-bit programs? Some years ago a number of companies made a standard called DPMI. It stands for Dos Protected Mode Interface. Under DOS a 32-bit intel CPU (like the 386, 486, pentium) runs in 8086 mode. Ok, it's a lot faster than a 8086 but it is still limited in some aspects. To gain the advantages of the modern processors, you (I mean the software guru's here) can switch the processor into something called 'protected mode'[3]. Protected mode has several advantages. To name a few:

- You can run more programs simultaneous. Windows 3.x does that. Each DOS box you open is a separate process.
- Protection. Programs cannot interfere with each other. This subject needs some attention. Implementation of this protection is usually different for different operating systems. For example a DOS box in Windows 95 can easily crash the system, where on the other hand a similar DOS box on a OS/2 platform does not. OS/2 is somewhat superiour in this aspect. Linux and other unix variants are far superior in crash protection. It's been told that Windows NT is far better in this but I don't tend to trust Microsoft on such advertising stuff.
- There is something called virtual 8086 mode. This is where the DOS boxes are. In this mode the CPU emulates a 8086 and re-routes specific functions to protected mode functions residing in the 32-bit kernel.
- Simple DOS boxes and multi-megabyte applications can co-exist. On Windows 95 this is very obvious, you can have DOS boxes, 32 bit applications and Windows 3.x applications all at the same time. Each program has its separate program space.

There are also disadvantages:

- Communication between tasks is difficult. The microsoft way is to use DDE/OLE but have you actually successfully used that? Modern programs use ActiveX and COM. I won't discuss these.
- Slower. A multitasking system is usually slower because more things need to be administrated. The operating system simply has to do more administration and must protect itself if more tasks make simultaneous system calls.
- It's harder for the software guys to create programs for these systems. Especially if these programs must interact with each other.

1. web2c: pronounce as web–to-cee.
2. Some say it's hardly an operating system, but merely a shell with some tools.
3. Windows 3.x calls this Enhanced mode

Installing web2c

So how to install the stuff? I assume you already have a working emTEX installation.

kpathsea

What? I don't know where this stands for but this is the tool all programs in the distribution use to find files. As you probably know TEX is a small program but you usually need an endless amount of files stored in another endless amount of subdirectories. Installation and maintainance is a difficult job and kpathsea gives you the option to store files in as many directories as you like. So you decide where to put the stuff and kpathsea finds out where it is.

What does it do? kpathsea maintains a database of files present in your TEX system. This database is not automatically maintained. You can decide not to use the database functionality but for the best, use it.

Each time you install new files you must update the database. There is a command for doing so `MakeTeXls-R`.

There is also the `kpsewhich` command which tries to find a file in the database. If you type:

```
kpsewhich cmr10.tfm
```

the program responds with:

```
c:/tex/share/texmf/fonts/tfm/public/cm/cmr10.tfm
```

If it doesn't, something is wrong and you should check the `texmf.cnf` file which is located in the c:/tex/share/texmf/web2c directory.

Change settings in this file as you see fit and experiment. I only changed `prefix` to `$SELFAUTODIR`.

Merging with the emTEX stuff

To merge in the emTEX stuff you have three options:

1. Merge some of the emTEX files into the new directory tree.
2. Maintain a separate emTEX tree.
3. Mingle with configuration files untill you get it right.

I wouldn't advise the first method unless you have configured emTEX lots of times and know what the ins and outs are.

If you choose to do the second, there won't be much of a problem. Beware of this though: you shouldn't have any TEX related enviroment variables like `%texinputs%`. This confuses the web2c stuff. So make sure your installation is standard and doesn't use environment settings[4]. Disadvantage of this solution: You'll need more disk-space for `PK` fonts and `TFM` files since both distributions need them. This means all `TFM` and `PK` files must be present twice: once for the emTEX tree, and one for the web2c tree. Though this is not a problem, suppose you got a new version of a font and you only updated the emTEX version. Things look fine on the viewer, but as soon as you use dvips from web2c to create postscript the old font shows up.

I chose the third method. I left all the files where they are, deleted the unnecessary ones and changed the texmf.cnf file so it also looks into the emTEX tree for information.

Changing texmf.cnf for emTEX

Apply the following changes:

1. Delete all `TFM`-files from the web2c tree.
2. Move all `TFM`-files from your emtex distribution into the `/emtex/tfm` directory. I'll explain this later. Make sure all `TFM` files are in a *single* directory.
3. Change texmf.cnf. Change the variable `TFMFONTS` and `PKFONTS`. You can use the dvidrvfonts environment variable for the latter.
4. Run MakeTeXls-R to update the file database.
5. rename the `xxxDPI` directories in the `PKFONTS` to `DPIxxx`.
6. You probably need to change some mfjob parameter files as well. Get into the /emtex/mfjob directory and edit modes.mfj. Look for the lines at the end. They say somehting like `def output_lj= ....` Change the `@Rdpi` into `dpi@R`. This is how web2c likes to have the `PK` files.
7. Rename the directory `pixel.lj` to the mode you are using (e.g. laserjet or something like that). Modify the `.CNF` file for dviscr to use this directory too.

Item 2 says you should move all you `TFM` files into a single directory. This is because web2c maintains the same directory structure when generating `PK` fonts. So if `cmr10.tfm` is in a subdirectory (cm), the `PK` file will also be in that subdirectory. This is different from emTEX.

The idea of having subdirectories is handy, but is not compatible between web2c and emTEX. In emTEX you can use the font cm/cmr10 by saying:

```
\font\f=cm/cmr10
```

but emTEX has no way of making a difference when using searching for cmr10.pk. In web2c this TEX feature doesn't work[5].

4. This is not an easy task.

5. This is very unlucky because in emTEX you could easily maintain different encodings of the same font, and still use the same file name. This is an aspect where TEX is not very portable accross systems, a missed oppurtunity.

You probably want to remove the emTEX metafont program. It is best to use only one metafont program. mfjob can call the METAFONT program that comes with web2c, but it's a bit tricky. Experiment here (clear out the `mfjobopt` in your environment).

Now start checking things out. Make sure TEX can find the font files. Simply run `testfont.tex` through TEX and try to create a font-sample from cmsy10 or something like that. If this checks out fine, you have to check the viewer. If the Mattes viewer works, try to delete a `PK` file that is used (you can find this in dviscr.dlg). When dviscr says *Run mfjob to generate the font?* type Y, and check if the font will be generated and positioned in the proper directory.

Don't forget: Your mode_defs in METAFONT must be the same for emTEX and for web2c.

Also check if dvips (from web2c) generates the correct `PK` fonts. Make sure the files end up in the correct directory. After a few hours of experimenting you'll have the TEX system you want, and you've learned a lot about configuring web2c.

Advantages of web2c

Here are some advantages of web2c:

- If you use web2c you will notice it is *extremely* flexible in configuration. Furthermore most programs related to TEX are available in web2c.
- One of the great advantages is the configuration. Although the `texmf.cnf` configuration file is not very easy to understand, once you master it you'll see the flexibility and power. On my old system (using emTEX) I had several environment variables and several configuration files depending on the person's name using TEX, are we running under windows, and so on. With web2c I can have a single installation on the network and provide stuff for all network users. Even the format files (latex.fmt and so on) are portable across platforms. So you can use a single installation and serve all users independant of the platform, whether it is DOS, win95/NT or even Linux. And if there's a port to the MAC, I wouldn't be suprised if it worked right away.
- When you are finally through configuring all the stuff (this takes an endless amount of time), the updating is pretty easy. If you get a new executable[6] you can overwrite the older version with the new one, re-generate the formats and you're done.
- Your users can take advatages of different platforms. The win95 version supports long filenames where the DOS version does not.
- Cross platform stuff. The format files are usually cross platform. An example: Suppose you have a file-server with a web2c installation. You only have to generate the format files once and then they can be used on win95, MS-DOS and linux. There are no separate format files needed for other platforms. Even endian conversion is done by web2c. This is real cool and very maintainable. Since you know you need only one format file you won't have different behaviour on different systems.
- And there is also pdfTEX. This is a TEX variant that has pdf files as output. So if you use TEX mainly to create `pdf` files (as I do) then this is a time-saver. You don't need to use dvips and distiller anymore. There is a drawback to pdfTEX: the graphics support is limited. You can't include fancy EPS files. You must convert them using GhostScript.
- The DOS version of web2c has another nice advantage. I always had problems running huge emTEX in a DOS box under Windows. The DOS port of web2c runs under DOS, win3.x, win95 and OS/2. It even runs under DESQview.
- TEX input files. In web2c you can specify which paths are used for `\input`. You can say for example: 'Hey TEX if you are using the latex format, look only in these directories for input files'. And for another format you can use a different set of directories. This comes in very handy if you have different formats each with their own set of input files. You don't want you LATEX 2_ε input files to mix up with input files of another package.
- Custom options. Example: Copy the program `tex.exe` in the `bin` directory to `latex.exe`. When you start the `latex.exe` it automatically loads the format `latex.fmt`. Ok, that's nice but you can do more than that. in the `texmf.cnf` configuration file you can say: `main_memory.latex = 500000`, which means: 'Dear TEX if the program name is latex.exe make sure there are 500000 main memory words available. This way you can customize various settings for different variants of the same TEX compiler.

Drawbacks of web2c

Are there any? Yes. First of all it is a huge package. The full installation requires some 20 megabytes (or more) and you have to administrate some things. Second, the configuration. Since web2c embodies the spirit of TEX, it is highly flexible. This results in lots of errors being made when configuring the system. Every distribution I used (win32, DOS and linux) didn't work right out of the box. There's always some variable wrong.

Currently there is no viewer for web2c. The viewer supplied is a quick hack and lacks lots of options. The best viewer for DOS is still dviscr, and it doesn't come easy

6. Although unlikely, suppose DEK found a bug in TEXand makes an update. It's more likely that a bug is fixed in the web2c distribution

to get this installed properly. The port for win95 includes xdvi. This is a good previewer and moreover it is well integrated into the web2c environment. At the time of this writing the port should be available as beta.

Conclusion

web2c is a decent TEX implementation and it is rather complete.

Installation and configuration is not easy, especially if you intend to use emTEX and web2c together. The way web2c uses directories is too complex for emTEX to handle. You must simplify things to them working.

I didn't talk about multiple directory trees, but it is possible to have more than one configuration file and let web2c use them simultaneously. Find some docs on that, there are plenty. As usual with TEX products, configuration and customizing things is not easy and takes time.

Bijlage 19

Gezeefd uit de TEX-NL discussie-lijst

Frans Goddijn
fg@fgbbs.iaf.nl

abstract
Dit is het vierde deel uit een serie die in 1993 door Philippe Vanoverbeke in de MAPS is gestart. Philippe maakte een selectie van berichten uit de TEX-NL lijst. Een aantal oplossingen, hints en gouden tips over onderwerpen waarvan je weleens denkt: „hoe zat dat ook alweer?" In de MAPS van voorjaar 1997 stond deel 3. Voor deze aflevering is een aantal berichten gezeefd uit TEX-NL van maart 1997 tot maart 1998.
Deze informatie is, samen met nog veel meer van dergelijke waardevolle berichten uit TEX-NL, ook na te lezen op FGBBS in het berichtengebied FGBBS.ARCHIVE.
De vragen zijn door velen gesteld. Voor een aantal van de meest ingewikkelde problemen was de oplossing het gemakkelijkst: overstappen op CONTEXT. Deze berichten zijn hier niet opgenomen, want de probleemstelling is vaak te moeilijk voor de gemiddelde lezer en de oplossing is zo simpel dat iedereen er op kan komen ...
Tenzij anders is vermeld, zijn de onderstaande antwoorden van Piet van Oostrum. Omdat het wat bewerkelijk is om sommige trucs over te tikken, staat dit artikel ook op FGBBS.

keywords
TEX-NL, Oostrum, Registered, Guru, tips, hints, hack, bug

Regelnummers

Kan iemand mij helpen aan de naam van de `.sty` file (LaTeX2.09) welke aan kantlijn van broodtekst een lijnnummer geeft van bijvoorbeeld om de 5 regels?
Gebruik:

`tex-archive/macros/latex209/contrib/misc/altnline.sty`

Gebruik `subfile.sty` voor het afdrukken van verbatimfiles met regelnummers.
Bij LaTeX2e: `lineno.sty`

Robin Fairbairns (`rf@cl.cam.ac.uk`) voegt toe:

There are two packages that do this. Neither is entirely satisfactory, since line–numbering isn't really a typesetting sort of thing, and TEX doesn't provide any direct hooks for it. Both packages are on CTAN.

`macros/latex/contrib/supported/lineno/`

(`readme`, `.sty` and `.tex`)

`macros/latex/contrib/supported/numline/`

(`.sty`, with a `.tex` tagged on after `\endinput`)

Lijn langs paragraaf

Ik wil graag een lijntje naast een belangrijke paragraaf in de marge zetten. Het lukt me niet met een combinatie van `\rule` en `\marginpar`, althans niet zonder handmatig de lengte van paragrafen te gaan zitten meten — en daarmee wordt het er trouwens ook niet echt mooier op. Is er een manier om de lengte van de paragraaf waar de `\marginpar` in staat aan de `\rule` mee te geven?
Als de paragraaf (alinea) zeker op één pagina blijft zou je een

```
\begin{tabular}{p{textwidth}|}
```

kunnen gebruiken. Als de alinea gebroken kan worden helpt het meten niet. Je kunt dan beter iets als een van de `changebar` pakketten nemen.

Nieuwe papiermaatbug?

De `config.ps` file is conform het recept zoals Piet van Oostrum reeds aangaf, en toch krijg ik nog de melding `Got a new papersize`. Zou dit nu echt een bug zijn?

Die melding had uit de programma source verwijderd moeten worden. DVIPS maakt een omschakeling naar A4 (omdat dat in de file `config.ps` staat), en meldt dat. Heel hinderlijk. Het wordt naar de standaard output geschreven. Als je je postscript file opgeeft met `-o` is het geen probleem, denk ik. Anders kun je met een binaire editor (of EMACS) de `dvips.exe` wijzigen en de „G" van „Got a new papersize" veranderen in een zero byte. Maak voor alle zekerheid een backup kopie. Waarschijnlijk staat de tekst er twee keer in.

Overfull?

I am just writing up my thesis and I have slight problems with the `fancyhdr` package. When I use it I always get an error saying

```
Overfull \vbox (2.49998pt too high) has occurred while \output is active[]
```

Hartmut Schirmer (`hsc@techfak.uni-kiel.de`*) schrijft:*

```
\documentclass[12pt,a4paper]{report}
\addtolength{\headheight}{2.5pt}
```

This should give you enough room in the header.

Walvisvaarders

Probleem: bij `${\rightharpoonup\atop AB}$` staat de harpoen naar mijn smaak te ver boven AB (en AB zakt iets onder de lijn.

Johan Wevers gaf de volgende oplossing aan:

Als je wilt dat AB niet zakt kun je het eens met `\mathop` proberen:

```
\mathop{AB}\limits^{\rightharpoon}
```

Als dit nog niet goed is kun je natuurlijk altijd nog met `\kern` e.d. gaan spelen, zie de definitie van LaTeX of b.v. `\AA` voor de details. Dan kun je het precies zo regelen als je wilt.

Verbatimomgeving

Waarom werkt het volgende niet?

```
\newenvironment{code}
  {\begin{scriptsize}
   \begin{verbatim}}
  {\end{verbatim}
   \end{scriptsize}}
```

Op deze manier kan het wel:

```
\usepackage{verbatim}
\newenvironment{code}
  {\scriptsize\verbatim}
  {\endverbatim}
```

Dus:

- `\usepackage{verbatim}` gebruiken
- `\verbatim` en `\endverbatim` gebruiken i.p.v. `\starttypen` en `\end{verbatim}`
- Geen andere `\begin` en `\end`'s erin nesten.
- Achter de `\end{code}` moet je op dezelfde regel niets meer zetten, want dat wordt weggegooid (met waarschuwing). Het spul is nl. met de verkeerde catcodes ingelezen en daar is verder niks meer aan te doen.

De implementatie in het `verbatim` package stopt niet bij de letterlijke tekst `\end{verbatim}`, maar bij een `\end{x}` waarbij `x` de laatste omgeving is die geopend was voordat het `\verbatim` commando uitgevoerd werd. Voorzover je tenminste over het uitvoeren van commando's mag spreken in LaTeX.

PS, dit is fout:

```
\newenvironment{code}
 {\begin{scriptsize}\verbatim}
 {\endverbatim\end{scriptsize}}
```

maar dit is goed:

```
\begin{verbatim}
\begin{code}
\begin{example}
xxxx
\end{example}
\end{code}
\end{verbatim}
```

waarbij natuurlijk de `\begin{example}` en `\end{example}` ook afgedrukt worden. De code van `verbatim` uit het `verbatim` package scant de input tot een `\end` en constateert dan dat het argument erachter (`{example}` in dit geval) niet hetzelfde is als de `{code}` waarin de `verbatim` begonnen is (de ‚current' environment).

In het foute voorbeeld is de current environment ‚scriptsize' maar er wordt nooit een `\end{scriptsize}` gescand. Wel een `\end{code}` maar die matcht niet.

Als je daarentegen `\scriptsize` en `\endscriptsize` (als die zou bestaan) gebruikt dan wordt de current environment niet gezet, terwijl verder hetzelfde gebeurt, op een extra groep na.

Bram Heerink (`bram@dutein3.et.tudelft.nl`), de steller van de oorspronkelijke vraag, schreef:

Het probleem met de verbatim omgeving binnen een nieuwe omgeving heb ik als volgt opgelost. Met dank aan iedereen die zo snel e-mail ging sturen.

```
\usepackage{verbatim}
\newenvironment{code}
{\scriptsize
\verbatim}
{\endverbatim\normalsize}
```

Dit werkt goed.

Ik blijf nog even op de lijst meelezen, al vind ik het wel verschrikkelijk technisch. Ik dacht dat ik LaTeX toch wel goed begreep maar de mail die hier over de groep vliegt doet mij toch enigzins verbazen. Maar goed, ik wil LaTeXbegrijpen dus ik blijf nog wel . . .

Huge en small in alinea's

Herman Haverkort loste een probleempje bij me op, waarvan ik de oplossing aan eventuele belangstellenden niet wil onthouden.

Probleem: een laatste alinea van een boek–in–de–maak had een grotere regelafstand dan de andere alinea's op de pagina. Heel hinderlijk omdat ik niet begreep waarom dat zo was. Erg lelijk ook.

Nu was die pagina, de laatste van het boekje, oorspronkelijk net iets te krap om de informatie (korte gegevens van de verschillende auteurs) in kwijt te kunnen en daarom begon de pagina met `{\small` en eindigde hij met een `}`.

Ik had nooit gedacht dat het gevaar daarin zat!

Oplossing: voeg een `\par` in, of een lege regel voor de laatste `}` komt.

De meeste lezers hier zullen het allang hebben geweten, maar ik begrijp nu ook waarom ik soms teleurstellend lelijke regelafstanden kreeg bij het gebruik van `\huge` in een tekst.

Enfin, hier is Hermans uitleg die ik van hem per email kreeg.

TEX zet een alinea pas als die helemaal is ingelezen, en kijkt ook *dan* pas naar de regelafstand. Als je twee alinea's als volgt zet:

```
{\small Ja. \TeX zet een alinea pas als die helemaal is ingelezen,
en kijkt ook dan pas naar de regelafstand. Als je twee alinea's
als volgt zet:

Ja. \TeX zet een alinea pas als die helemaal is ingelezen, en
kijkt ook dan pas naar de regelafstand. Als je een alinea als
volgt zet:}
```

dan gebeurt er het volgende. TEX schakelt over naar kleine letters en kleine regelafstand. TEX leest de eerste alinea in, maakt daar een lange rij kleine letters van, komt een witregel tegen, en verdeelt de lange rij letters over een aantal regels met kleine regelafstand. Vervolgens leest TEX de tweede alinea in, maakt daar een lange rij kleine letters van, schakelt (door de accolade sluiten) weer terug naar normale letters en normale regelafstand, komt een witregel tegen, en verdeelte de lange rij (kleine) letters over een aantal regels met de inmiddels weer geldende *normale* regelafstand.

Oplossing: zet in dit soort gevallen altijd een `\par` voor de accolade sluiten (of een witregel). TEX verdeelt (vanwege de `\par`) de alinea dan over regels nog vóórdat hij weer terugschakelt naar normale regelafstand (vanwege de accolade).

Alleenstaande dames en kinderen

Een ander probleem loste ik (alweer met behulp van de lokale TEX–guru op door het rustiger aan te doen met mijn penalties.

Probleem was: de pagina's hadden een wisselend aantal regels en de onderste regel stond niet steeds even hoog. Kennelijk had TEX, in zijn moeite vooral geen weduwen en wezen te maken, gekozen om op moeilijke pagina's er eerder mee op te houden, soms veel eerder, soms iets eerder.

De oplossing bleek te zijn:

```
\widowpenalty=150  % weduwen: last line on newpage: NO
\clubpenalty =150  % wezen
```

Dat zijn volgens mij de gewone standaard-waarden. Ik had, ijverig hackend en tips overnemend uit o.a. Wynton Snow's *TEX For The Beginner* (blz 319/320) de waarden gezet op:

```
\widowpenalty=10000  % last line on newpage: NO
```

```
\clubpenalty =10000
```

Ik was daarmee niet in slecht gezelschap, want deze waarden vind ik ook terug in `maps.cls` op mijn HD.

Het *kan* dus zo te zien onverstandig zijn om ijverig te spelen met zulke penalty–waarden. Terwijl het zo verleidelijk leuk is. „We willen geen wezen, dus verhogen we de penalty daarvoor naar 10.000, why not?"

Het ware leuk geweest als in zo'n *TEX for the Beginner* ook werd uitgelegd waarom je die waarden zou verhogen en waarom niet. Maar misschien kan iemand anders dat gaatje in de markt eens vullen.

Taco Hoekwater (`taco.hoekwater@wkap.nl`*) voegde hieraan toe:*

Een redelijke waarde voor `\widowpenalty` en `\clubpenalty` is afhankelijk van de lengte van je paragrafen en de hoeveelheid ‚vrije witruimte' waar TEX mee kan prutsen. In een wiskundig zwaar artikel (4–6 display formulas per pagina) kun je deze zonder bezwaar op 10000 zetten, maar voor 11 punts pagina's kale tekst schijnt rond de 2000 redelijk te werken.

Ik denk dat de bron van die 10000's in `maps.cls` dezelfde is als die van jou, en het is ook daar niet zo'n best idee.

Paragraaf recycling

Ik wil soms uitgebreide citaten invoegen: complete paragrafen of hoofdstukken uit andere documenten compleet met de originele paragraafnummering en verwijzingen. Totnutoe doe ik dat door de originele LATEX–code te nemen en alle verwijzingen en paragraafnummers met de hand te vervangen (dus bv. `\section*{Blabla}` wordt `\section*{3.2 Blabla}`) en deze code in het nieuwe document tussen te voegen.

Guy Geens (`guy.geens@elis.rug.ac.be`*) antwoordde:*

Met de hand de counters verzetten.

```
\newcounter{savechapter}
\newcounter{savesection}
\setcounter{savechapter}{\value{chapter}}
\setcounter{savesection}{\value{section}}
\setcounter{chapter}{3}
\setcounter{section}{1} % Merk op: het volgende \section commando
                        % telt er een bij op.
\section{Blah blah}
...
% En nu weer de normale tekst
\setcounter{chapter}{\value{savechapter}}
\setcounter{section}{\value{savesection}}
```

Ik zou wel aanraden om een ander font, of marge te gebruiken om duidelijk te maken wat er juist geciteerd wordt.

Lelijke vec

Als ik `$\vec{q}^4$` intiep, komt de pijl van vec recht door de vier heen te staan. Ik (en iemand anders die mij erop wees) vind dat lelijk.

Hoort die pijl door de vier te gaan, en zo nee, weet er iemand wat ik fout doe?

Harald Lubbinge `h.lubbinge@wb.utwente.nl` *schreef:*

Je zou een hspace kunnen invoegen, dus als volgt:

```
\vec{q}\hspace*{1mm}^4
```

*Taco Hoekwater (*taco.hoekwater@wkap.nl*) voegde toe:*

Je zou dit kunnen doen:

```
$\vec{\,q}^4$
```

Dat is iets beter, maar nog steeds ouderwets. Kluwer gebruikt al jaren deze definitie:

```
\def\vec#1{{\bf #1\/}}
```

*dr. R. Rietman (*rietman@natlab.research.philips.com*) reageerde:*

Eigenlijk hoor je helemaal geen vectoren tot de vierde macht te willen verheffen, (wat moet ik me daarbij voorstellen?) maar je kunt de 4 op een betere plaats krijgen met

```
$ \vec{q}{\,}^4 $
```

Een aantal wiskundige notaties, waaronder het pijltje om een vector aan te geven en ook de maffe N, Z, Q, R en C voor de getallichamen, zijn bedacht om met een krijtje op een schoolbord geschreven te worden. Verschilllende fonts gaat daar niet zo gemakkelijk. Dat wil niet zeggen dat je diezelfde notaties ook in geTEXte manuscripten moet gebruiken.

Hanging initials

Does anybody know about a style file providing ‚hanging initials' (The first letter of a section (or paragraph) being as big as to go over two or three lines. This package should take care of the witdh of the certain letter (not leave as much space for an I as for a W.

Jeroen Nijhof (nijhojhb@aston.ac.uk) wist de weg naar de volgende pakketten:

on CTAN:

```
ftp://ftp.dante.de/tex-archive
```

or

```
ftp://ftp.tex.ac.uk/tex-archive
```

or mirrors:

```
macros/latex/contrib/other/dropping/
```

Iemand anders stelde ongeveer dezelfde vraag:

Weet iemand of er een macro bestaat, die van het aangeboden argument (een woord) de eerste letter afhaalt, deze letter sterk vergroot afdrukt en de rest van het argument er vervolgens achter plaatst om daarna de rest van de zin af te drukken (zoals in sommige boeken gebeurt)?

*Hans Hagen (*pragma@pi.net*) antwoordde:*

De volgende macro heb ik ooit eens voor de gein gemaakt. Hij werkt heel redelijk en zal t.z.t. deel uitmaken van de CONTEXT fun stuff module (ik heb ook macros die b.v. de eerste regel een kleurtje kunnen geven).

```
\def\DroppedCaps#1#2#3#4#5#6#7% command  font     height
  {\setbox0=\hbox               hoffset  voffset  lines
     {\font\temp=#2 at #3%
      \temp#1{#7}\hskip#4}%
   \setbox0=\hbox
     {\lower#5\box0}%
   \ht0=\ht\strutbox
```

```
   \dp0=\dp\strutbox
   \hangindent\wd0
   \hangafter-#6\noindent\hskip-\wd0\vbox{\box0\nobreak}}

\DroppedCaps {\kleur[groen]} {cmbx12} {2.2\baselineskip} {2pt}
{\baselineskip} {2} Hello there are we once more hello there
are we once more hello there are we once more hello there are
we once more hello there are we once more hello there are we ...

\def\MyDroppedCaps%
  {\DroppedCaps
     {\kleur[groen]}
     {cmbx12}
     {2.2\baselineskip}
     {2pt}
     {\baselineskip}
     {2}}

\MyDroppedCaps Hello there etc etc etc etc etc etc etc
etc etc etc etc etc etc etc etc etc etc etc etc etc etc
etc etc etc etc etc etc etc etc etc etc etc etc etc etc
etc etc etc etc etc etc etc etc etc etc etc etc etc etc
```

(als er behoefte is aan dit soort gimmicks, dan kan ik er wel een stukje in de MAPS aan wijden.)

Titels afbreken

I have a title for a chapter is too long and I can't break it by `\\` or `\linebreak`. Any suggestion?

```
\chapter[This title has no linebreaks and may be
abbreviated]{This is the very long title\\and it has a
line break in it}
```

Hoe strekt men een baslijn?

How to use `\baselinestretch`? I use `\renewcommand{baselinestretch}{1.5}` in my dissertation to change line space for tables and sometime it works and sometime it doesn't.

This is basically the correct usage, but you must take care that a type size changing command is executed to get the desired effect. Also there must be a `\par` command explicitely or impicitely while the desired value is still valid. The setpace package has a number of environments that take care of these subtleties.

Stephen Eglen (`stephene@cogs.susx.ac.uk`*) voegde hieraan toe:*

I'd recommend looking at `setspace.sty` available from CTAN. It allows you to do things like:

```
\begin{spacing}{0.9}
\bibliographystyle{unsrt}
\bibliography{gen}
\end{spacing}
```

Van figures.sty naar graphics [dvips]

Sinds ik ben overgestapt van het figures pakket naar graphics (de variant met een x erin),

moet ik in al mijn docs bovenin `[dvips]` zetten als option voor de classfile. Is daar een makkelijker weg voor?

Het antwoord staat in de standaard–documentatie van `graphic[xs]`, in `grfguide.tex`: zet een file `graphics.cfg` in je `texinputs`–pad met daarin `\ExecuteOptions{dvips}`

Dat is alles.

Grijze box zonder PostScript

Deze 3 regels geven errors.

```
\documentclass[emtex]{article}
\usepackage{graphicsx}
\usepackage{color}
\begin{document}
\definicolor{dark}{gray}{0.5} %niet gedefinieerd???
\colorbox{dark}{test} %werkt niet!!
\rotatebox{30}{gnu} % roteert niet!!!
\end{document}
```

Al deze 3 lijnen geven errors. Weet soms iemand hoe dit moet opgelost worden want volgens Leslie Lamport zijn boek zou het zo moeten gedaan worden!

Bert Kassies (`kassies@nlr.nl`*) hielp:*

Bij mij werkt dit wel. Ik doe bijna hetzelfde:

```
..
\usepackage{color,graphicx}
..
\definecolor{mygray}{gray}{.90}
..
\colorbox{mygray}{\begin{minipage}[t]{10cm}
 mijn tekst
\end{minipage}}
```

De juiste drivers

Ik probeer, The LaTeX graphics Companion, pagina 42 lezend, te vergroten, verkleinen en reflecteren, maar het wil niet. De volgende source doet niets van dat al (de teksten zijn allemaal evengroot en niet gereflecteerd), weet iemand wat ik fout doe?

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
\scalebox{2}{groot}\\
normaal\\
\scalebox{.5}{klein}\\
\reflectbox{gespiegeld}
\end{document}
```

Erik Frambach (`e.h.m.frambach@eco.rug.nl`*) legde uit:*

Deze truuks zijn erg driver–afhankelijk, zie pp. 28–29. EMTEX–drivers bv. kunnen vrijwel niets; DVIPS kan alles. Als je een ‚gewone' previewer gebruikt zie je inderdaad niks gebeuren, maar als je het in POSTSCRIPT print of met GHOSTSCRIPT bekijkt/print zie je het wel degelijk.

Portrait en landscape

Als ik GHOSTSCRIPT gebruik en mijn `\documentclass[landscape,dvips]{}` zo definieer, dan kan mijn printer niet landscape printen (zelfst als ik de printer handmatig op landscape mode zet!), en het ziet er niet landscape uit in de GHOSTSCRIPT viewer eveneens.

De tip van Siep Kroonenberg (`n.s.kroonenberg@eco.rug.nl`*):*

Probeer eens de `-t landscape` parameter voor DVIPS.

Betaalde TEX

Waar is TEX te koop?
Originele URL's:

- `http://www.tug.org/interest.html#vendors`
- `http://www.YandY.com/`: Y&Y Inc, selling TeX systems for Windows, also the makers of the Lucida families of PostScript fonts
- `http://www.pctex.com/`: Personal TEX Inc, selling a package for MS–DOS and MS–WINDOWS users
- `http://idt.net/~kinch`: True TEXis a TrueType based TEX for MS–WINDOWS
- `http://www.micropress-inc.com/`: Visual TEX, an (extended) TEX for MS–WINDOWS

Taco Hoekwater legt uit:

vendor	**font types**	bitmap support	graphics
Y&Y	Type1 (+ pk)	TIFF, EPS, WMF	PS, TPIC, emlines
TrueTEX	TTF + ATM + VF	Veel (1)	PS (2)
pcTEX	TTF (+ ATM)	BMP, EPS, WMF	pictex
VTEX	alles + VF	BMP, PCX, TIFF, GIF, JPG, TARGA	(2)

1. Ingebouwd zijn (vage formaten gewist): WMF, EMF, BMP, JPEG, PCX, TIFF, Targa, Photo CD, MCP, PBM, PGM, PPM, X Window Dump, G3 FAX, GEM Bit, XPM, XBM, EPS previews (WMF, PICT en EPSI). Uitbreidbaar via DLL's, waarvan er ook een aantal bijgeleverd zijn, o.a. voor GIF.
2. Graphics via een subset van POSTSCRIPT die direct door de previewer wordt geïnterpreteerd.

Ze zijn denk ik alle vier goed, maar de uitstraling is wel anders. Y&Y richt zich vooral op professionals die veel met andermans bestanden werken of met PDF bezig zijn, TrueTEX op andere professionals (kleine zelfstandigen of zo), pcTEX heeft vooral een naam op te houden maar is eigenlijk een beetje achterhaald aan het worden, en VTEXricht zich meer op privé–gebruikers die leuk willen kunnen werken (syntax highlight in de editor, wysiwyg equations, HTML output en meer van dat soort maffigheden).

Dikkere lijn in tabular

Is it possible to get a thicker `\hline` within a tabular environment?

Rainer Schoepf (`schoepf@uni-mainz.de`*) tipt:*

```
\setlength{\arrayrulewidth}{3mm}
```

Een andere oplossing kwam van Axel Reichert (`reich@mpie-duesseldorf.mpg.de`*):*

```
\usepackage{booktabs}
\toprule
```

```
\midrule
\bottomrule
```

© symbool©

Can anyone tell me how to put a *registered mark* symbol i.e. a superscript circled R?

*Robin Fairbairns (*rf@cl.cam.ac.uk*) antwoordt:*

\textregistered (with a sufficiently ‚current' LaTeX– needs 1995/12/01, I suspect): ©.
And in superscript: © ^{\textregistered}

*Maarten Gelderman (*mgelderman@econ.vu.nl*) voegt toe:*

Misschien werkt \textcircled{R}?

*Harald Lubbinge (*h.lubbinge@wb.utwente.nl*) voegt toe:*

Met de package amssymb:

```
\circledR
```

*Hans Hagen (*pragma@pi.net*) voegt toe:*

```
% gejat van Knuth (zie \copyright, p356)
\def\omcirkeld#1%
  {{\ooalign{\hfil\raise0.07ex\hbox{{\tfx#1}}\hfil\crcr\mathhexbox20D}}}
\def\copyright%
  {\omcirkeld{C}}
\def\registered%
  {\omcirkeld{R}}
```

Hier is \tfx een kleiner corps met behoud van stijl, mogelijk in LaTeX iets als \def\tfx{\small}.

*Wybo Dekker (*wybo@servaly.hobby.nl*) sluit af met:*

Bedankt Hans, dat werkte prima. Alleen is small te groot en heb ik er superscript en sans serif van gemaakt: ®

```
\def\omcirkeld#1%
 {\raise1ex\hbox{\ooalign{\hfil\raise0.07ex\hbox{%
 {\textsf{\scriptsize#1}}}\hfil\crcr\mathhexbox}}}

\def\registered%
  {\omcirkeld{R}}
```

\textcircled{R} werkte ook wel (Maarten Gelderman) maar daar staat de R excentrisch.
De \circledR van Harald Lubbinga heb ik niet geprobeerd omdat ik amssymb niet bij de hand had.

Dotloze

I don't understand how/why LaTeX knows the dimensions of a dotlessj in Palatino if the glyph isn't defined. It does, because I get a black rectangle of the correct size when I print or display the PostScript file. Also I can't find where the character shapes are defined in the system I'm using; all I can find are font metrics; but GHOSTVIEW must get the information from somewhere — or is it in the code itself?

*Robin Fairbairns (*rf@cl.cam.ac.uk*) schrijft:*

However, nothing *knows* the dimensions of a glyph that doesn't exist; there is only a dotted j in the .afm file for palatino. Frans' code (for a dotless ▮) produces something

that makes an informed guess, but fonts with strange shapes might not conform to his otherwise apparently sensible guessing strategy.

If you want an accented j, you really need a font that contains the composite glyphs (since otherwise you won't be able to hyphenate words containing the accented glyphs: this is a known and tedious restriction of the way TEX does accents). Note that the LATEX T1 encoding doesn't have any accented j characters in it, since none of the languages that it was designed to support employ it. There are no accented–j–characters in ISO–Latin-1, either (which isn't exactly surprising, given that T1 covers a wider range of languages than ISO–Latin-1 ...)

If you're after a maths symbol of some sort, you are of course out of luck, since no one has designed a Palatino maths font (maths fonts are pretty rare on the ground, after all).

*Jörg Knappen (*knappen@iphcip1.Physik.Uni-Mainz.DE*) voegt toe:*

Unfortunately, Alan Jeffrey (as the maintainer of fontinst) didn't want to include the dotless ı into the fontinst distribution, despite the fact that it can be made available at least using DVIPS as driver (don't know about the commercial drivers, but it seems to me that Y&Y was hindering progress here).

Fortunately, Thierry Bouche made a distribution of the standard 35 font containing the dotless ı glyph.

It is available under:

ftp://fourier.ujf-grenoble.fr/pub/contrib-tex/psfonts/gs35nfss.tar.g

In Latin-3 you will find the j with circumflex for esperanto. Also a j with hachek is used in several transcriptional systems (and maybe in writing native languages of America).

TEX (and LATEX) are structured correctly, having a dotless ı in their encodings and default fonts *from the beginning*. It was not a strange glyph (like perthousandzero) added later to the T1 encoding.

However, commercial fonts are wrong not providing a dotless ı, and the default fontist is wrong in not correcting this unfortunate fact of life.

*Sophie Frisch (*frisch@blah.math.tu-graz.ac.at*) reageert:*

From CTAN, get:

pub/tex/fonts/psfonts/tools/dotlessj.*

and do as the readme says to create dotlessj by chopping off j at the height of dotless ı. Works for me (with plain TEX that is — everything's more complicated with LATEX of course).

David Carlisle (david@dcarlisle.demon.co.uk) discussieert mee:

Fontinst comes (or came) with a contributed prologue to make a dotless ı, and fontinst always works out the metrics for a \j, even if it just makes a warning (on the reasonable assumption that the metrics may be made by taking those of j and changing the height to the height of a dotless ı).

The reason it does not actually make the glyph by default is that that requires to use a POSTSCRIPT clip path to take out the dot.

This would make the vf files much less portable, the postscript inclusion \special is not standardised, and the afm metrics may not be being used on a POSTSCRIPT RIP. A type 1 engine such as ATM would not be able to cope with such arbitrary POSTSCRIPT code, nor would engines that are using the fonts in truetype form.

But you could do something like this:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{dotlessj}[1997/09/10 v0.01 dotless j package (DPC)]
```

```
\RequirePackage{color}
% delay everything as  \j and friends get defined by encoding files
% read in by, eg fontenc package.
\AtBeginDocument{%
% rule offset slightly to catch italic cases.
\DeclareRobustCommand\j{%
  {\sbox\z@{j}%
  j%
  \kern-.8\wd\z@
  {\color{white}\vrule \@height\ht\z@ \@depth -1.1ex \@width \wd\z@}%
   \kern-.2\wd\z@}}%
\DeclareRobustCommand\jacc[1]{%
  {\leavevmode\sbox\z@{j}%
   \hb@xt@\wd\z@{\hss\clap\j\clap{#1{}}\hss}}}%
\def\clap#1{\hb@xt@\z@{\hss#1\hss}}%
% Could do this for all known encodings, but just do it
% for the default encoding at begin document, to avoid wasting
% too much space.
\DeclareTextCompositeCommand\^{\encodingdefault}{\j}{\jacc\^}%
\DeclareTextCompositeCommand\~{\encodingdefault}{\j}{\jacc\~}%
\DeclareTextCompositeCommand\.{\encodingdefault}{\j}{j}%
\DeclareTextCompositeCommand\={\encodingdefault}{\j}{\jacc\=}}
\endinput
```

A test file and example for this package:

```
\documentclass{article}
\usepackage{dotlessj}
\usepackage{times}
\usepackage[T1]{fontenc}
\begin{document}
\section*{Do '\j ' and '\^\j' live in moving arguments?}
\large
\j \i  \^\j \^\i  \~\j \.\j \=\j
\it\noindent
\j \i  \^\j \^\i  \~\j \.\j \=\j
\end{document}
```

Nummerloze noot

*Op de vraag hoe je een voetnoot zonder nummer krijgt, schreef Thierry Bouche (*thierry.bouche@ujf-grenoble.fr*):*

Here is my \ufootnote (unnumbered footnote)

```
\def\ufootnote#1{\let\savedthfn\thefootnote\let\thefootnote\relax
\footnote{#1}\let\footnotemark\savedthfn\addtocounter{footnote}{-1}}
```

Extra regelruimte in tabel

(Hoe) is het mogelijk een extra stukje vertikaal wit in een tabular te zetten, d.w.z. bijvoorbeeld 1ex meer tussen twee \hline's?

Het volgende werkt niet:

```
\hline
\vskip1ex
```

\hline

*Jaap vd Zanden (*j.p.p.m.vanderzanden@wbmt.tudelft.nl*) schrijft:*

Ik gebruik \\[3.0ex] of *[3.2ex] om meer ruimte te maken.

*Herman Haverkort (*hh@fgbbs.iaf.nl*) voegt toe:*

\def\arraystretch{2} bijvoorbeeld:

```
\documentclass[11pt]{artikel3}
\begin{document}
\def\arraystretch{2}
\begin{tabular}{|l|l|}\hline
tra & la\\\hline
tra & la\\\hline
tra & la\\\hline
tra & la\\\hline
\end{tabular}
\end{document}
```

Lege regel voor verbatim?

Momenteel schrijf ik tekst met zo af een toe een stukje computercode in de verbatim–omgeving. Nu valt het me op dat er tussen de tekst en het begin van het verbatim–gezette blok zo ongeveer een lege witregel wordt geplaatst. Hoe kan ik de afstand tussen de reguliere tekst en het verbatim–blok zelf bepalen?

Een verbatim is een list omgeving (eigenlijk trivlist), en die ruimte is dus {\topsep (+\partopsep als er een \par of lege regel staat).

Kijk moeder zonder margins

This is a very basic question, how to I change the default margins in a latex document. Basically I want no margins at all, the full use of a page. I tried something like

```
\setlength{\textwidth}{7.0in}
\setlength{\oddsidemargin}{0.in}
\setlength{\evensidemargin}{0.in}
\setlength{\topmargin}{0.in}
\setlength{\textheight}{11.5in}
```

However I still have a left and top margin.

*Keith Reckdahl (*reckdahl@am-sparc7.stanford.edu*) antwoordt:*

I'm not quite sure why you would want *no* margins, but ... While you can set all the parameters by hand, the easiest way of setting margins is to use the geometry package.

```
\usepackage{geometry}
\geometry{left=0in,right=0in,bottom=0in,top=0in}
```

You might also have to include

```
\verb|headheight=0in,headsep=0in,marginparwidth=0in,marginparsep=0in|
```

Pagina's nummeren

How can I get this:

- in the lower right corner of the first page the number: 5
- in the lower left corner of the second page the number: 6

That positioning of the page numbers is not one of the default ‚page styles'. (There should have been more page styles in the base distribution!)

*Donald Arseneau (*asnd@erich.triumf.ca*) schreef:*

You can define your own page style:

```
\documentclass[twoside]{article}
\makeatletter
\def\ps@plaincorner
  {\let\@mkboth\@gobbletwo
   \let\@oddhead\@empty\def\@oddfoot{\reset@font\hfil\thepage}%
   \let\@evenhead\@empty\def\@evenfoot{\reset@font\thepage\hfil}}
\makeatother
```

Then use it, starting at 5:

```
\begin{document}
\pagestyle{plaincorner}
\setcounter{page}{5}
```

Or you can use `fancyhdr.sty` which provides a more intelligible interface, but at the expense of reading yet more pages of documentation.

Schaartjes in overvloed

I'm looking for a scissor–symbol. I know there's one in the POSTSCRIPT font Dingbats (char 34), but I'm not using PSNFSS. Is there another solution?

*Robin Fairbairns (*rf@cl.cam.ac.uk*) laat zichzelf versteld staan:*

I astound myself at times. There are two METAFONT dingbat fonts on CTAN (that are obvious to the naked eye), and I seem to have tried both at one stage or the other . . .

Anyway, . . . `fonts/bbding` seems to be what you need: it has a whole row (8 glyphs) of scissor symbols — a veritable childhood paradise.

Uitstekende woorden

Ik dacht dat het onmogelijk was, maar Jacoline ziet bij het nakijken van een drukproef dat een woord uitsteekt buiten de rechterkantlijn. In een zin met twee vrij lange woorden staat, voor de nieuwe regel wordt begonnen, het woord ‚,waarom" met de letters ‚,om" buiten de rechterkantlijn. Ik begrijp niet goed waarom TeX daar niet ‚,waarom" van maakte (is nu handmatig gedaan).

Naar welke setting, `\tolerance` of zoiets dergelijks, moet ik zoeken om de oorzaak te achterhalen?

*Hans Hagen (*pragma@pi.net*) schreef:*

Probeer eens:

```
\emergencystretch=1em
```

Piet van Oostrum voegt toe:

Dat ‚waarom' niet afgebroken wordt, komt doordat `\righthyphenmin` op 3 staat. Dus na het afbreekstreepje moeten tenminste 3 letters komen. Dit is een typografische aanbeveling. Je kunt het op 2 zetten, maar dan krijg je typografisch gezien minder wenselijke afbrekingen.

De oplossing van Hans met emergencystretch is ook aan te bevelen, eventueel met een iets grotere waarde.

Afbreken na /

Volgens Renkema (schrijfwijzer) mag je voor en na een schuine streep geen spaties zetten. Maar als je dan een uitdrukking/opmerking maakt met een schuine streep er in wil LaTeX weer niet afbreken. Hoemoetdatnu? Volgens mij zou de schuine streep bij voorkeur aan het eind van de regel na het eerste woord moeten staan, en moet ook eventueel een van beide woorden afgebroken kunnen worden.

Dus:

uitdrukking/opmerking als het even kan;
als het zo uitkomt: uitdrukking/
opmerking, en als het per se moet: uit-
drukking/opmerking.

Als iedereen begrijpt wat ik bedoel. Is dit niet iets voor `dutch.sty`?

Werenfried Spit (`w.spit@witbo.nl`) antwoordt:

De oplossing zal iets wezen als

```
\def\schrap{\allowhyphens\discretionary{/}{}{/}\allowhyphens}
```

maar dan moet je steeds `\schrap` gebruiken in plaats van `/`. Ik overzie niet helemaal wat er zoal mis kan gaan als je `/` botweg herdefinieert.

Piet van Oostrum voegt toe:

Overigens heeft PLAIN TEX hier het commando `\slash` voor, dat ook door LATEX is geërfd:

```
uitdrukking\slash{}opmerking als het even kan; als het zo uitkomt:
uitdrukking\slash{}opmerking, en als het per se moet:
uitdrukking\slash{}opmerking.  Als iedereen begrijpt wat ik bedoel.
```

Maar dat schrijft inderdaad lastiger dan `/` of Johannes' `"/`

Inslagschema's

Onze drukker gebruikt het programma InPosition (1.6.3), een Quark extensie om zijn inslagschema's te maken. Dit programma kan de paginainformatie niet vonden in een PostScript file gemaakt met `dvips` (5.58 van 4TEX–cd).

Wat moet ik doen om de paginainformatie beter ‚zichtbaar' te maken?

Je moet de optie `-N0` gebruiken (of nog beter `N0` in de `config.ps` zetten)

Peter de Jong (`p.dejong@twi.tudelft.nl`) kwam met:

Oplossing die bleek te werken:

```
%!PS-Adobe-3.0
          ^    even aanpassen
%%Requirements:
%%DocumentNeededFonts: (atend)
%%DocumentSuppliedFonts: (atend)
%%EndComments
```

Dit moet in de plaats van de gebruikelijke font informatie komen; eigenlijk vind ik dit vreemd want ik dacht dat (`atend`) voor kleinbreinigen was bedoeld.

Verder moet de paginasize anders worden gegeven:

```
%%BeginFeature: *Papersize a5
a5
%%EndFeature
```

Dit is moderner en kan via `config.ps`

Backslash

Hoe zet je een ‚roman' backslash in de tekst?

Jeroen Nijhof (`j.h.b.nijhof@aston.ac.uk`*) antwoordt:*

Het standaard font, `cmr10`, bevat geen backslash. En zo hoort het ook: wel eens een backslash op een typemachine gezien? (Run `tex testfont`, met `font=cmr10`, commando `\table` en commando `\bye`, dan bevat `testfont.dvi` een tabel van alle karakters in `cmr10`)

In PLAIN TEX is de backslash alleen in mathmode bereikbaar: `ab$\backslash$cd`
Als je LATEX gebruikt, kun is de Cork encoding gebruiken:

```
\usepackage[T1]{fontenc}
```

en de backslash is dan te vinden onder `\textbackslash`

Johan Wevers (`johanw@vulcan.xs4all.nl`*) schreef:*

De volgende file laat je zien welk karakter in een font zit (verander de `cmr10` in het font dat je wilt bekijken):

```
\documentstyle[a4wide]{article}
\begin{document}
\def\hsp{\hspace*{5mm}}
\pagestyle{empty}
\newfont{\fonttest}{cmr10 scaled \magstep0}
\noindent
\fonttest

\char"00\hsp \char"01\hsp \char"02\hsp \char"03\hsp
\char"04\hsp      ...     \char"9A\hsp \char"9B\hsp
\char"9C\hsp \char"9D\hsp \char"9E\hsp \char"9F
\end{document}
```

Regelafstand

Kan ik in LATEX de regelafstand vergroten?

Paul Huygen (`phuygen@xs4all.nl`*) antwoordde:*

- Zoek het package `setspace` op in CTAN.
- Goedkoop alternatief: Zet in de preamble: `\renewcommand{\baselinestretch}{1.5}`
  Volgens The LATEX Companion moet je een *font size changing command* (zoals `\small`, `\large` o.i.d.) doen om de baselinestretch verandering te effectueren.

Excell en SPSS in LaTeX

Hoe kan ik van een MSexcell bestand of van een SPSS bestand een ‚afdruk' in EPS–formaat maken?

Luc de Coninckc (`luc.deconinck@kh.khbo.be`*) Schrijft:*

Met MS–WINDOWS95 —zorg voor een driver die POSTSCRIPT aan kan (bijv. Adobe, Apple, HP) en installeer op je PC. Dan kun je de gewenste bestanden (bijv. Excell) naar die drukker sturen. ... afdrukken naar een bestand (je geeft zelf de naam, extensie POSTSCRIPT) en de nodige instellingen doen voor POSTSCRIPT: uitvoer naar POSTSCRIPT (portability format). Dat POSTSCRIPT–bestand kan je dan behandelen met GHOSTVIEW om er per blad een

EPS–bestand van te maken. En die gaan dan zomaar in je TEX–bestand.

Afdrukken naar een bestand in EPS–formaat gaat ook (indien het Excell–bestand maar 1 pagina is) maar dat lijkt niet altijd goed te werken. Vandaar de bovenstaande weg!

We probeerden dat hier met Mathcad en dat gaat uitstekend. We hebben echter lang gezocht om dat aan de praat te krijgen. En de oorzaak van de fout was dat we ‚harde returns' op de pagina's aanbrachten. En dat loopt blijkbaar grondig verkeerd voor het EPS–formaat (mag slechts 1 (volledige) pagina zijn!).

*Maarten Gelderman (*mgelderman@econ.vu.nl*) voegt toe:*

Je kunt de POSTSCRIPT driver (bij voorkeur die van Adobe, te downloaden van www.adobe.com) gewoon naast je bestaande printerdriver installeren. Gewoon een questie van downloaden en de installatie–instructies volgen. De twee printer–drivers krijgen geen ruzie met elkaar, je moet alleen even opletten dat je de goede selecteert. Daarna, zoals reeds eerder aangegeven kiezen voor EPS–uitvoer, printen naar bestand, en optimize for compatibility.

Zelfbouw superscript

Hoe kan ik iets afdrukken in superscript zonder gebruik te maken van ‚math' mode. Het gaat hier bijvoorbeeld om 1^e, 2^e enz. De bedoeling is dan dus dat de e–tjes bovenaan staan.

\raisebox*{*distance*}*[extend-above][extend-below]*{*text*}*

The \raisebox command is used to raise or lower text. The first mandatory argument specifies how high the text is to be raised (or lowered if it is a negative amount). The text itself is processed in LR–mode.

Sometimes it's useful to make LATEX think something has a different size than it really does — or a different size than LATEX would normally think it has. The \raisebox command lets you tell LATEX how tall it is.

The first optional argument, ‚extend-above', makes LATEX think that the text extends above the line by the amount specified. The second optional argument, ‚extend-below', makes LATEX think that the text extends below the line by the amount specified.

(Noot FG: ^{e} is hier net zo handig.)

Examens

Ik ben op zoek naar een macro of template oid waarmee je makkelijk een examen layout kan maken. Ik bedoel zoiets als:

Vraag 1. De molecuulformule van water is H_2O.

a. Hoeveel H atomen bevat 1 watermolecule?
b. . . .

Op CTAN kon ik niets vinden. Heeft misschien iemand iets bruikbaars voor dit doel?

Wat bedoel je met „kon ik niets vinden"?

tex-archive/macros/latex/contrib/supported/exams/
tex-archive/macros/latex/contrib/supported/exam
latex/contrib/supported/examdesign

Pointsize and baselineskip

Kan er zonder heksentoeren uit te halen een font van 11.5pt met een baselineskip van 13pt opgelegd worden aan LATEX?

*Taco Hoekwater (*taco.hoekwater@WKAP.NL*) legt uit:*

Bijna met je ogen dicht zelfs. Maak een copie van `size11.clo` (of `bk11.clo`, naar keuze) naar de directory waar je tex in runt, en verander daarin

```
\renewcommand\normalsize
  {\@setfontsize\normalsize\@xipt{13.6}
```

Naar

```
\renewcommand\normalsize
  {\@setfontsize\normalsize{11.5}{13}
```

Let wel even op, want omdat er in de `.fd` files voor CMR geen 11.5 pt gedefinieerd is, krijg je voor de math fonts de CMR̃2pt versie tenzij je MathTime gebruikt. Als je dat niet ziet zitten moet je ook de
`XXXXCMR.FD` aanpassen. Er staan daarin altijd lijsten met zoiets:

```
<10.95>cmr10
```

en die kun je fixen door dat te veranderen in

```
<10.95><11.5>cmr10
```

(verschil is maar een half puntje, maar LaTeX zeurt hier nogal over)

Zebra d

Voor het noteren van een inexacte differentiaal in een tekst zoek ik een d met een streepje erdoor.

Johan Wevers (`johanw@vulcan.xs4all.nl`) schreef:

Ooit eens gemaakt voor eenzelfde doel (differentiaal van een niet–toestandsgrootheid):

```
\def\dd{d\hspace{-1ex}\rule[1.25ex]{2mm}{0.4pt}}
```

Euro symbool

Hoi, Ik ben een nieuwe gebruiker van TeX en ik heb een vraag: Bestaan er fonts met daarin opgenomen het symbool voor de Euro? Het duurt namelijk niet echt lang meer voor we (echt) met de Euro te maken krijgen.

De EC fonts hebben een text companion font met een euro symbool erin.

```
\usepackage{textcomp}
...
\texteuro \textsf{\texteuro}
```

Maar je moet waarschijnlijk de laatste versie van LaTeX2.09 hebben (22 Jan 1998).

Tweekoloms in kader

Ik heb de volgende omgeving aangemaakt om een tweekolomstekst in een kadertje te zetten:

```
\newsavebox{\tmpbox}
\newenvironment{vraagbox}
  {\sbox{\tmpbox}
   \bgroup\begin{minipage}{\boxwidth}
   \begin{multicols}{2}
   \setlength{\parindent}{\baselineskip}\noindent\ignorespaces}
  {\end{multicols}\end{minipage}
   \egroup\noindent\fbox{\usebox{\tmpbox}}\medskip}
```

Nu vind ik dat er aan de onderkant van het kader te weinig wit staat tussen het kader en de tekst. Met andere woorden, ik wil of de witruimte tussen de onderkant van `\tmpbox` en het frame vergroten, ofwel de diepte van tmpbox vergroten voor ik hem gebruik. Dit staat ongetwijfeld in het TEXbook, maar dat ligt helaas thuis. Kan iemand mij uit de nood helpen met een oplossing?

(OK, nood is wat overdreven, het geheel kan natuurlijk ook gewoon geprint zonder de extra witruimte, maar ja, je gebruikt LATEX, of niet hè.)

het antwoord van Piet:

- Een `\vspace{..}` tussen `\end{multicols}` en `\end{minipage}`
- Je kunt `\dp\tmpbox=...` zetten voor je hem gebruikt maar dan moet je de minipage een optionele parameter `[b]` geven.

Titels

Is het mogelijk om i.p.v. op alle pagina's de hoofdstuktitel te hebben de titel van de `\section` te bekomen door gebruik te maken van deze opmaakodes?

...

Bijlage 20

BachoTEX'98 – TUG at hand

Kees van der Laan

abstract
A report of GUST's 6th conference in Bachotek, Poland

keywords
Bachotek, BachoTEX'98, GUST, Poland, Polish TEX Users Group, conference

The usual BachoTEX meeting of the Polish TEX usersgroup was on. I was once again invited, this time because of an out-of-the-ordinary occasion. The meeting was cooked from the ingredients tutorials, workshops, reports, and the bonfire social event with the 'guitars at night.' Some old faces did not show up, alas, while promising new TEXies joined. In total roughly 80 people attended accompanied by some 25 relatives who just enjoyed the holiday; the family affair as noticed last year continues. Only 5 from abroad attended: the diehards Volker Schaa, Gyöngy Bujdosó, and me, the new ones Petr Olsak and the polyglot Christopher Piñón – he is fluent in English, French, German, Hungarian, Polish, Spanish . . . – from the USA. I was this time the only dutchie, but accompanied by my dog. He socialized better than I ever did. All loved him.

Participants got a BachoTEX'98 T-shirt.

The upcoming TUG meeting later this year in Toruń, to be hosted by GUST, was dangling.

I was happy to be able to communicate now with Janusz Nowacki in . . . Russian.

Conference

Nearly all reports were in Polish, except for Petr Olsak's lecture in Czech and my 'Tiling in PostScript and Metafont' in English. Happily, abstracts were also provided in English, and volunteers provided simultaneous translations

As usual the BoP(P) company – Jacko, Piotr and Piotr – had their impressive contributions. A tutorial about the in-and-outs[1] of digital graphics, and demonstrations of their new tools, all in (portable) PostScript: improved version of PS_View, TIFF2PS, TTF2PS, PF2AFM, and Colormap. The latter allows modifying a bitmap – colourizing black and white graphics, brightening and the like – from within TEX. In short all graphic formats can be brought back to EPS and handled in a tidy, trustworthy, and efficient way. Other reports dealt with

- Haskell – yet another tool supporting work with TEX by Ryszard Kubiak
- Aspects akin to the Polish language, such as spellingcheckers for Polish, LATEX 2_ε and the Polish language[2]
- Petr Olsak lectured about TEX and Czech[2]
- Computerizing the old Półtawski's Antykwa font by Janusz Nowacki[3].
- Andrzej Tomaszewski once again captivated us by his typographer's experience.

The following tutorials offered the participants to keep up with the world outside.

- PostScript and raster graphics – what is it all about, by Jacko (Bogusław Jackowksi)
- SGML by Adam Dawidziuk
- AUCTEX, TEXing from within the emacs editor by Ryszard Kubiak
- TEX and acrobat by Tomek (Thomasz Przechlewski)
- LATEX2HTML by Piotr Bolek

Workshops

As usual there were also various workshops offered: Jacko (Bogusław Jackowksi) about TEX, Metafont, PostScript, . . . Staszek (Stanislaw Wawrykiewicz) about the file server and so on. However, the interest was less than usual.

General meeting

Nearly all showed up. Very lifely, joyful and democratic. Tomek was reelected as president. Jacko, Jola (Jolanta Szelatyńska), Bogus, Jerzy Ludwichowski, and so on – in total 10! – are on the board. To the list of honorary mem-

1. From the abstract: The basic kinds of discrete computer graphics will be presented (the black-and-white, the shadowed, the palette, RGB, CMYK, and other kinds of graphics). Topics related to retrieving graphics on raster devices will be discussed (kinds of screens, Moire's effect, smoothing, etc.) together with known ways of data compression. The lecture will be crowned with a survey of the most popular formats of graphics: PCX, GIF, TIFF, JPEG, with a stress on the capabilities of PostScript and how to use them for illustrations in TEX documents.

2. They refrain from Babel?!?

3. Interesting was the non-computer aspect: which of the variants is the real one!

bers were added: Hanna Kołodziejska, Marek Ryćko, and . . . me.

Conclusions

BachoTEXs continue to be a family affair. GUST is doing well. PostScript and TEX-Metafont/Post are GUSTs primary tools, with literate programming in the background. They have their knowledgeable stable kernel of TEXies, a nice bulletin, their (email) network and fileserver, and annual meeting to socialise. Next to that they take their share in organising internatial TEX meetings, with the Toruń TUG meeting in August as their second effort. My congratulations.

Thank you GUST for having invited me, thank you Ryszard Kubiak and Piotr Bolek for the simultaneous (private) translations of the lectures.

See you next year, hopefully earlier.

Bijlage 21

BLUe's OTR for notes: back-to-the-roots

C.G. van der Laan

abstract

The back-to-the-roots OTR for BLUe's notes is discussed. It consists of Plain TEX's OTR for 1-column and the compatible extension as given In The TEXbook for 2-columns. Only the pagebody differs: 2-columns instead of 1-column. This replacement is aimed at facilitating a personalized preprint OTR, such that BLUe can easily adapt it. The modified `blue.tex` will be distributed by CTAN, and NTG's 4AllTEX CD-ROM.

keywords

BLUe, education, macrowriting, manmac, output routine, plain TEX, preprint format.

1 Introduction

The abbreviation OTR denotes OutpuT Routine. In The TEXbook 251 the function of an OTR is described as follows

> 'Page numbers, headings, and similar things are attached after each page has been ejected, by a special sequence of TEX commands called the current output routine.'

The OTR has all to do with with the look-and-feel of a publication. Important issues are

- size of the page body, the page proper
- the head'line' and foot'line'
- fonts (kind, representation and associated quantities)
- global magnification.

In this note the replacement of BLUe's OTR for notes is accounted for. It is not a plea for more, not for moving frontiers of science.[1] More the opposite, to go back-to-the-roots, to honour what was already given in The TEXbook, and to distill what for PPT notes is needed and integrate this in `blue.tex`.

Knuth discusses OTRs in The TEXbook Chapter 23, and in the Appendices B—`\plainoutput`—and E—`\begindoublecolumns`, `\doublecolumnout`, and `\enddoublecolumns` as part of the example format manmac, to typeset the 2-column index in The TEXbook.

I'm only sorry that I was led astray a few years ago by the appeal of TUGboat's OTR, and I apologize for what I made out of it. In the change some errors have been accounted for. It was less than what Knuth already had described in The TEXbook Chapter 23.

I leave advanced wishes and perfections to professionals, especially those in a demand-driven environment, because `blue.tex` is all about minimal markup with results of a preprint nature.

2 PLain's OTR

`\plainoutput` is compact. It is explained in The TEXbook 253–256, and listed in Appendix B 364. Its purpose is to set a page proper preceded by a headline and followed by a footline—both just one line—in 1-column. Top insertions and footnotes have been accounted for. The size of the pagebody block is invariant towards scaling.

The usual `\hoffset` and `\voffset` control positioning of the page on the paper. Default 1 inch on top and on the left. The toplevel of the macros read as follows.

```
\def\plainoutput{\shipout\vbox{%
 \makeheadline\pagebody\makefootline}%
 \advancepageno
 \ifnum\outputpenalty>-20000
 \else \dosupereject%end of note
 \fi}
```

The next level reads.

```
\def\makeheadline{\vbox to 0pt
 {\vskip-22.5pt
  \line{\vbox to8.5pt{}%
        \the\headline}%
  \vss}\nointerlineskip}
\def\pagebody{\vbox to\vsize
 {\boxmaxdepth=\maxdepth \pagecontents}}
\def\makefootline{\baselineskip24pt
  \line{\the\footline}}
```

Explanation. Makeheadline moves upward within a zero-sized vbox. The headline is set in an hbox of `\hsize`, and that is it. For the footline Knuth assumes that the footline is

1. David Salomon has discussed various advanced applications of the OTR in a series of 4 notes published in TUGboat.

really just one line and modified the baselineskip to create the separation between the page proper and the footline.[2].

The user can adapt the look-and-feel by changing

- `\hsize` and `\vsize`, the page proper parameters[3]
- `\headline` and `\footline`, token variables
- the magnification.

3 Incorporation in BLUe

All we have to do is to envelop the OTR and parameter settings in `\onecol`.

```
\def\onecol{\output{\plainoutput}%
  \vsize=25truecm\hsize=16truecm
 \def\makeheadline{\vbox to0pt{%
  \vskip-22.5pt
  \hbox to\hsize{\the\headline}\vss}}%
 \def\makefootline{\baselineskip24pt
  \hbox to\hsize{\the\footline}}}
```

Remark. The header on the first page can be suppressed by the following.

```
\headline={\global\headline
 {\sevenrm\the\issue\hfill\it\the\title}}
```

The line which marks the beginning of the note is wired-in `\beginscript`, but can be suppressed by omitting there

```
\hrule\kern2ex\noindent
```

4 Knuth's PPT 2-columns OTR

In The TEXbook 257 Knuth provides a 2-column variant.[4] A simple one in the spirit of the PPT idea, straight, no fancy tricks. No beginning with the title matter over the width of the page, no balancing of the columns on the final page, no switch over from 2-to-1 column, or vice versa, in general. Please do read Knuth's 257.

4.1 Incorporation in BLUe

All we have to do is to envelop the OTR and parameter settings in `\twocol`, and handle `\fullhsize` appropriately..

```
\newdimen\fullhsize \newbox\leftcolumn
\def\twocol{\let\lr=L
\output{\if L\lr
   \global\setbox\leftcolumn=\pagebody
   \global\let\lr=R%
  \else\doubleformat\global\let\lr=L%
  \fi
  \ifnum\outputpenalty>-20000
  \else\dosupereject\fi
}%end output
%
\def\doubleformat{\shipout\vbox{%
  \makeheadline
  \hbox to\fullhsize{\box\leftcolumn
                \hfil\pagebody}%
  \makefootline}\advancepageno
}%end doubleformat
%
\vsize=25truecm \hsize=7.75truecm
\intercolwd=.5truecm
\fullhsize=2\hsize
\advance\fullhsize by\intercolwd
\def\endscript{\makesignature
  \vfill\supereject
  \if R\lr\null\vfill\eject\fi
  \end}
\def\makeheadline{\vbox to0pt{%
  \vskip-22.5pt
  \hbox to\fullhsize{\the\headline}\vss}}%
\def\makefootline{\baselineskip24pt
  \hbox to\fullhsize{\the\footline}}%
}%end twocol
```

Remarks. The headline and footline read for 1- and 2-column format the same only the `\hsize` must be adapted.

The 2-column OTR for the index for The TEXbook, that is in Manmac, sets small, halfwidth, pages of doubble length and after that each is split into 2-columns. For typesetting the index in 2-columns Knuth used a special OTR for that purpose next to the general one. This special OTR calculates a 'page' of half the width but twice the usual length, and after that splits this into 2 equal halves, to be combined as two columns on the real page. A nice idea, but it does not account for footnotes.[5] Knuth mentions

> 'It's possible to do fancier column balancing on the last page but the details are tricky if footnotes and other insertions need to be accommodated as well.'

It becomes simpler when footnotes are printed at the end of the second column, I presume.[6]. However, the OTR for the index of manmac allows the beginning—title of the chapter and introductory remarks—to be set over 2-columns, and it ends again in 1-column to set the quotations.

2. Remember that TEX encloses the OTR invoke by (scope) braces. See The TEXbook 253
3. With defaults 6.5 true in and 8.9 true in.
4. The extension to 3-column is given in The TEXbook exercise 23.4.
5. See The TEXbook 417.
6. The latter is done in the MAPS approach of 1997, in LATEX

5 Conclusions

BLUe's note OTRs have been simplified, by taking over what Knuth already provided. A user can adjust `\onecol` and `\twocol` easily, by copying these macros and changing parameter values.

For the size of the page proper the (independent) parameters are `\vsize` and `\hsize`, the height and width of the text in one column.

For `\twocol` the parameter `\intercolwd` is also an independent parameter and can be adapted. For this situation the width of the page—`\fullhsize`—is a dependent parameter and equals $2 * \text{“}\texttt{hsize} + \text{“}\texttt{intercolwd}$.

`blue.tex` of 1997 will be sent to CTAN, and hopefully be distributed on NTG's 4AllTEX CD-ROM.

Have fun, and all the best.

Bijlage 22

The Oldenburg eTEX/LATEX3/conTEXt meeting

David Carlisle

Present[1]

etex Peter Breitenlohner, Philip Taylor, Bernd Raichle, Jiri Zlatuska, Karel Skoupy.
latex3 Frank Mittelbach, Rainer Schöpf, Chris Rowley, Matthias Clasen, David Carlisle.
context Hans Hagen, Taco Hoekwater.

Categories

To start the discussion, Peter (initially) suggested some basic 'categories' of problem that could be discussed. The following discussion did not always follow this categorisation but it proved a useful starting point (in fact this list was modified during the discussion in the first session, and ended up something like this:

Syntax/Expansion/Programming Extensions to the macro programming language, not directly affecting typesetting.
Line-breaking/hmode
Page-breaking/vmode
Grid Typesetting
Characters and Fonts
Alignments
Math
Character Attributes (eg Colour)
Translation (eg I/O)
List Manipulations Manipulations on token or node lists.

Expansion Control

This was more or less discussed by email before the meeting. The following new commands were *agreed*:

\expanded⟨***general text***⟩ Expandable command returning the full expansion of the tokens in ⟨*general text*⟩.
\expandlater⟨***number***⟩ One level expand the nth token ahead. (`\expandafter` = `\expandlater\tw@`)

The following variants were discussed with no definite conclusion (?)

\expandfromtoken⁀⟨***number***⟩ This would look ahead to token ⁀(as in delimited argument matching) and then move forward (or back if negative) ⟨*number*⟩ tokens and expand the resulting token one level.
\undef⟨***csname***⟩ Undefine ⟨*csname*⟩. Like `\let\xxx\@undefined` but without relying on `\@undefined` being undefined. Some discussion over whether this was really any improvement over the `\let` form (given that you need to know in practice that `\undef` has not been redefined) or whether it would be worth it if it did not also remove ⟨*csname*⟩ from the hash table (which would be hard).
Real Register A data type that has the same behaviour/arithmetic as ⟨*dimen*⟩ but without the need to supply units. (This would save the many uses of `\strip@pt` in latex where units are added just to do arithmetic, and then need to be removed.
Boolean Datatype Some question as to what if anything was wanted here. Boolean variables, or boolean expressions, or... Also any extension would need to fit with current `\if\fi` nesting behaviour.
\ifinsidebox Similar to `\ifinner` but different. (See Hans' email).
Local/Global Assignments Some mechanism to specify that within the current scope global assignments are 'global' to nested groups but local to the current group.[2]

Line Breaking (hmode)

Full typography in hmode A switch to allow hboxes to be constructed with all the features that currently are only active in outer hmode. (Language nodes etc).
Discretionaries after - Switch to turn off automatic insertion of discretionaries after (ligatures ending in) -.
Hyphen desirability levels Two basic proposals.

1. Extended hyphenation patterns with weighted hyphenation points.
2. Extended hyphenation algorithm that tries first with patterns designed to split on compound word boundaries, then if needed a second pass with a finer set of patterns.

1. not everyone present all the time
2. Rainer sketched a possible interface on the board, which didn't make it to my notes — Rainer?

Second version has possible advantages in keeping the current hyphenation file format. Not clear what exactly is needed.

Page Building (vmode)

Need for a vertical analogue of `\discretionary` identified. (Probably just make `\discretionary` work in vmode.) This would seem to solve several problems with current lack of control over top-of-page behaviour.

Agreed new commands:

`\forcebreakpenaltylimit` If set to ≥ -10000 this is is ignored, but if set to < -10000 is taken as the threshold at which penalties force a break.

`\nthmark`⟨***number***⟩⟨***number***⟩⟨***number***⟩
`\nummarks`⟨***number***⟩⟨***number***⟩

`\nummarks`⟨*mark class*⟩⟨*box number*⟩ returns the number of marks of that class in the specified box. `\nthmark`⟨*mark class*⟩⟨*mark number*⟩⟨*box number*⟩ returns the nth mark of the specified class. (Generalises `\firstmark`).

This pair of commands was thought better than the original suggestion of an `\allmarks` command returning a list of all marks.

Other items to be considered:

`\outputpenalty10000` Look at removing the penalty node in the `\outputpenalty`10000 case.

holdinginserts `\holdinginserts=2` (or different name for compatibility reasons) Hold but also copy inserts (ie a combination of current behaviour for 0 and ≥ 1). Alternatively just use `\holdinginserts=1` but give access to individual inserts in some way similar to `\nthmark`.

buildpagehook Insert a token register at the point where 'build page' is called. The (effective) default contents of the register would be `\buildpage`, a new primitive that actually builds the page. cf `\output`/`\shipout`.

Node Handling

Perennial discussion about lack of `\last`* and `\un`* for most node types, and problems with revealing machine dependence and lack of register type corresponding to some nodes, eg rules.

Agree to look at `\removelastnode`.

Parsing General Texts

There was some discussion of generalising the `\uppercase` mapping of character tokens, although any decision would have to wait to see the final outcome of the proposals for symbolic character handling (see below)

General idea is `\map`⟨*map name*⟩⟨*general text*⟩.

Thus `\uppercase`would be `\map\uccode`(or perhaps a special csname to refer to the ⟨*uppercase map*⟩.

Some discussion of whether a token-token map would be enough (especially for the kind of parsing done in the context system) or whether a more general multiple token to multiple token system would be needed, but that leads down the road to implementing regexp replace (or OTP) and it was not clear if that fitted the etex framework.

An alternative to having an explicit map prefix command would be to have a way of automatically applying the map to input characters (but see later discussion on encoding support).

Mathematics

Matthias Clasen[3] reported on an investigation of possible extensions that he had recently done with Michael Downes. (See document Matthias posted.) Briefly the main points were:

mathstyle The providing of an integer valued variable that reveals and controls the current math mode (would work in a way similar to `\interactionmode`). This would require the provision of a prefix form for `\over`and friends. `\over`would still work but would be documented as *not* setting the mathstyle variable to the expected value if this infix form is used.

`\ifcramped` Test for cramped style, together with switch to force entry to cramped style.

kerning Between more kinds of math atoms.

Spacing table Access to the table of spaces inserted between the math atoms.

math classes Extend the number of these to allow more varied automatic spacing possibilities. Also primitive to ask the math class of an atom. (This would save bm taking the `\meaning` and then parsing the information out of the hex string returned.)

subformulas Avoid boxing subformulas.

glyph positions Remove dependence on weird glyph positioning in the font, like radical hanging below baseline, so math fonts could be used with other systems.

3. Matthias has since experimented with some change files to implement some of these ideas, see `ftp://peano.mathematik.uni-freiburg.de/pub/etex`

radical extensions Right end for radical? Or (harder) Both ends extending together.

Horizontal extensibles Some discusion of whether this would need an extension of tfm format. Outcome seemed to be that it would not, just use exsiting tfm data structure, the TeX primitive would know whether it was constructing a horizontal or vertical extension.

overloading of font dimens *Agreed* to look at reducing this.

Fixed parameters eg clearance around rules. *Agreed* to look at this as well.

under accents Use information that could be coded in the 'reverse' kerning with the skewchar for underaccents. However full accent solutions, including multiple accents requires table driven lookup. or otp or ...

16 families Increase?

tfm If format was extended, possible benefits?

Inner loop — lig/kern problems

After discussing several strategies for solving the problem that anything non expandable breaks ligature and kerning, including the possibility of allowing assignments in the inner loop (finally thought to dangerous) the following proposal (from Bernd, initially) was agreed to be worthy of further consideration:

Redesign hpack routine and the linebreak algorithm so that kerning and ligature is separated. First create the hlist with no lig/kern then make a second pass adding the ligatures and kerns. (This is the conceptual algorithm, an implementation may want to interleave these processes when 'safe' eg word by word.)

This would mean that fi always produced the fi ligature.

new primitive would be needed to break ligatures but not break hyphenation process also similarly `\nokern`.

Some questions on semantics of unhbox in this setup, whether the list should be "re-ligatured".

Modified algorithm could further split things up separating ligatures first then kerning. (Currently tex just does whichever comes first in the fonts lig/kern table if both are specified for a pair)

Also discussed mechanisms of overriding ligatures for a font eg the Portugese no fl ligature example. Also, simpler the possibility to turn off *all* ligatures in a font, for verbatim type uses.

Reconsider paragraph

If the possibility is given to re-break an hlist, tex needs to return more information about the resulting paragraph to enable a choice to be programmed. (It was thought unlikely that one could algorithmically set the paragraph parameters to 'correct' an earlier try, but one could at least try several pre-programmed possibilities, and then use the returned information to choose the 'best' outcome.

Possible info that might be useful

- Information currently only available in display math (`\predisplaysize`)
- total number of hyphens
- maximum number of adjacent hyphens
- maximum badness of a line
- minimum badness of a line
- total demerits
- fit class of worst line

The plan implemented but not released in V2 broke the paragraph onto the current vertical list, as normal but returned an hlist for further tries. This means that you need to program the removal of the original paragraph if you need to retry, which means not working on the main vertical list. Some alternatives were considered.

Basic scheme in TEX is:

construct hlist until reach `$$` or `\par` or end-of-vmode-group. Then

- Modification at end (unskip, add parfillskip etc), leave hmode.
- (A)
- look at legal breakpoints & choose optimal path (+/- `\looseness`)
- line break decision (1st pass, hyph, 2nd pass, 3rd pass)
- (B)
- postlinebreak (package, migration, append to vlist)
- in vmode or disply math (first point tex looks for token)

Plan 1 copy list at (A) into one global variable (destroy previous contents). This is a private unpackaged hbox. New primitive to 'unhbox' this where required.

Plan 2 At (B) call 'output routine'. Has badness information from linebreaking routine but not vertical size information. (This plan rejected, as vertical size information seems vital for intended applications).

Plan 2b Append the new vertical material, but don't exercise the page builder.

Word Grabbing

The following proposal seemed the most plausible (after several other schemes were rejected)

New expandable primitive `\grabword`

Expans tokens up to the first non expandable token that is not a ⟨*letter-other-space*⟩. Returns the resulting tokens surrounded by explict `{}` tokens and leaves the 'terminating' non expandable token in place.

Typical use

```
\def\foo#1{<pcdata>#1</pcdata>}
\expandafter\foo\grabword user supplied stuff.
```

Note that the 'word' that is picked up will be prematurely terminated by all the stuff that currently breaks ligatures, but if the symbolic character propsals could be implemented these would rarely need to appear mid word.

Named Reference Points

Named reference points on (h or v)boxes.

Like `\mark\special`etc puts special named reference point into current (vh)list.

When hbox is packaged reference point node has a position relative to reference point on box (integer arithmetic).

These positions could be made available similar to current `\wd` and friends.

`\xxxxx`⟨*number*⟩⟨*name*⟩

takes a box number and a symbolic name and returns the position (might need two commands to get x,y coordinates, or one that returns a pair)

When unbox a vertical list the information is lost but the reference nodes are still in the list so their positions in the new list will be made available when the new list is packaged.

When packaging an outer box, inherit all labels from inner ones.

Possibly (or more likely not) return some some info about the nesting.

Would the linebreaking algorithm need to add labels to each line???

Special `\specials`

A `\special` for writing positional information to the dvi file?

A delayed `\special`. (cf non-immediate `\write`).

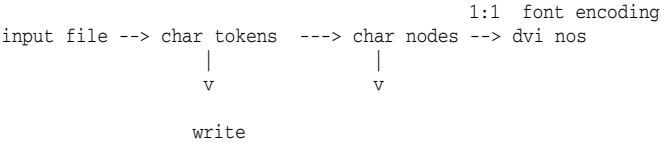
Combining penalties

`\penalty-500 \penalty10000` currently still allowable break, so user has no way to overrule some automatically inserted penalties. Various algorithms considered, eg some average, max, min, etc. In practice probably doesn't matter as long as it preserves 'finiteness'.

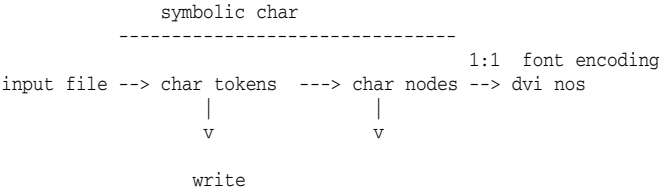
Any algorithm that allows to overwrite existing penalties that favour a break would be a very big improvement and should be *looked at closely*.

Symbolic Characters

Some schematics of current and proposed flow of 'characters' through tex. (Peter can write on the board faster than I can draw ascii art on this laptop. The following diagrams probably are rubbish, but I keep them exactly for now. Need replacing by something a bit more meaningful)[4]



want:



for each language have subset $\leq$ 256 SC which map to hyphenable ascii codes

SC+language encoding+font encoding ——————

Hyphenation Patterns (always based on 8bit table)

Possibly the mapping from symbolic characters to ascii codes may be many to one?

SC + font enc — represented in a font (somehow)

possibility of using unicode as 'symbolic names'

[This section needs filling in in a lot more detail]

Paramater matching

Two sets of extensions to paramater matching were considered.

Alternative delimited argument tokens

`\def\foo#1\a#|\b#|\c#2{...}`

The argument `#1` would be all the tokens up to the first occurence at the current brace group level of `\a` *or* `\b` *or* `\c`.

Within the definition the following forms would be available

`#1` the tokens of the first argument.

`#|1` the tokens delimiting the first argument, thus one of `\a`,`\b`,`\c` in the example.

`#2` the tokens of the second (non-delimited) argument.

4. Mathias has made some more extensive notes on this subject

`#|2` As the second argument is not delimied it is not clear what this should be. Probably empty, but could be an error, either would be OK as a behaviour.

Note that this extension is upwords compatible as the `#|` forms are all error conditions in the current tex.

A step towards regexp matching

Currently `\def\foo#1#2#3\xx{...}` always assigns the smallest possible arguments to the the first and second arguments with the third argument taking up the maximum number of tokens up to the delimiting token.

Proposal to allow

`\def\foo#*1#2#?3\xx{...}`

`#*1` argument may take non or more brace groups or tokens.

`#+1` argument may take one or more brace groups or tokens.

`#?1` argument may take at most one brace groups or tokens.

February 26–28 1998

Bijlage 23

Summary of math font-related activities at EuroTEX '98

Barbara Beeton
American Mathematical Society
Providence, RI, USA
bnb@ams.org

Thierry Bouche
Université Joseph Fourier
Grenoble, France
thierry.bouche@ujf-grenoble.fr

Taco Hoekwater
Kluwer Academic Publishers
Dordrecht, The Netherlands
taco.hoekwater@wkap.nl

Patrick Ion
American Mathematical Society
Ann Arbor, Mich., USA
ion@ams.org

Jörg Knappen
Johannes-Gutenberg-Universität
Mainz, Germany
joerg.knappen@uni-mainz.de

Chris Rowley
Open University
London, UK
C.A.Rowley@open.ac.uk

Ulrik Vieth
Heinrich-Heine-Universität
Düsseldorf, Germany
vieth@thphy.uni-duesseldorf.de

1 Introduction

The subject of math symbol fonts has been one of the major topics of interest at the 10th European TEX Conference (EuroTEX '98), which was held on March 29–31, 1998 at St. Malo, France as part of the 2nd Week on Electronic Publishing and Digital Typography (WEPT '98).

During the conference a paper summarizing the activities of the Math Font Group[1] (MFG) [1] was presented and two BOF sessions on math fonts were held, bringing together members of the MFG and representatives of other interested parties, such as the W3C MathML working group, the STIX project, as well as publishers and typesetters.

In addition, there were also many private discussions on math fonts at lunches, dinners, and at informal get-togethers in the local cafés or pubs.

The discussions at the BOF sessions primarily revolved around two major topics:

- the organization of math symbols in general, including their representation on the WWW,
- the development and implementation of new 8-bit math fonts for (LA)TEX in particular.

2 Organization of math symbols

On the first topic, Barbara Beeton and Patrick Ion of the AMS provided some information about the so-called STIX project, which is driven by a group of scientific and technical publishers (STIPUB).

So far, the primary goal of the STIX project has been to compile a comprehensive list of *all* math symbols used by the participating publishers (also including what many people might call "unreasonable" ones), to document their intended meanings, and to provide examples of their use in support of an application to the Unicode Consortium and the ISO working group on coding standards.

A preliminary list of symbols has already been submitted to Unicode in March 1998, but it appears that there are quite a few symbols that have been missed, so the Unicode submission will have to be followed up when more material is available.

Apart from compiling a comprehensive list of math symbols, there is also a commitment to commission the production of a set of high-quality fonts implementing all the symbols, which should be freely distributable. It is hoped that the availability of such a font set will be a crucial step to help promote the use of MathML on the World Wide Web without being restricted to the symbol complement provided by the system fonts.

Information about the list of symbols collected by the STIX project currently resides on internal pages on the AMS Web server [2] and is kept in a format similar to the Unicode symbol tables [3], but there are plans to release a printable version of these tables in PDF format to the general public soon.

1. also known as: LATEX3 Project / TUG Technical Working Group on extended math font encodings (WG 92-01)

It was pointed out that a printable version of the symbol tables from the MathML specification is supposed to be available in PDF format on the W3C Web server [4]. The latest version of the MathML specification also includes some background information about the STIX project and references to various other glyph collections [5, Chapter 6].

3 Implementation of new 8-bit math fonts for (LA)TEX

On the second topic, the status of the activities of the Math Font Group was reported in a conference paper [6] presented by Ulrik Vieth in the morning session on the first day of the conference.

So far, a set of encodings for new 8-bit math fonts for (LA)TEX has been developed based on a proposal dating back to TUG '93 [7], which aims to fulfill certain design goals and to satisfy a number of technical constraints.

These encodings, which consist of three primary encodings and several additional ones, have been implemented as a set of virtual font using glyphs taken from existing or newly-developed METAFONT or POSTSCRIPT fonts. Several such sets of virtual fonts have been developed, covering most of the presently available sets of math fonts usable with TEX, but the implementation unfortunately remains incomplete in several cases. It also doesn't yet take into account many of the symbols identified by the STIX project, which may have to be added to the proposed encodings if they are really needed.

A LATEX interface to access the new encodings and to switch between different font sets implementing these encodings is already in place and may be used either as a module to build a modified LATEX kernel or as an add-on package for use with standard LATEX. However, a Plain TEX interface is still missing and remains to be developed.

Given all these preparations, the question remains whether the encodings developed by the MFG are acceptable to the user community and whether they satisfy the needs of scientific and technical publishers. While there wasn't a clear answer to this question, there seemed to be a consensus that new 8-bit math fonts addressing the organizational problems of the old 7-bit math fonts were indeed needed and that the work of the MFG provides a suitable starting point, which may have to be refined later during the implementation process.

In particular, there was a suggestion to relax the strict requirement for compatibility with Plain TEX or LATEX within the first four math families, and to adopt a slightly more rational organization which would allow to have fewer missing glyphs in some implementations of the primary symbol font by leaving out some problematic glyphs and relocating them to one of the extra symbol fonts.

Another request came from A. Berdnikov, the coordinator of the 8-bit Cyrillic encodings for TEX, who pointed out that Russian math typesetting traditions required different shapes of big operators, such as upright integrals and bigger versions of summation and product signs, and asked to support such variants in the new math font encodings as well.

Since it is clear that it will be necessary to add several additional symbol font encodings if all the STIX glyphs are to be incorporated eventually, minor adjustments to the present proposal may be needed anyway and should not present a problem. In any case, the encoding tables presented at the conference should not be taken as the final word.

A strong driving force to push forward the implementation of new 8-bit math fonts for (LA)TEX came from Taco Hoekwater of Kluwer Academic Publishers. As part of his professional activities, he is currently working on a project to implement as many mathematical symbols as possible in Type 1 format by the end of this year, possibly including everything in the list of STIX glyphs.

Since Kluwer Academic Publishers consider their products to be journals and books, not fonts, Taco is allowed to put all the fonts he produces for Kluwer into the public domain. He has already converted several existing METAFONT symbol fonts (including

rsfs, stmary and wasy) to Type 1 format using the MetaFog converter [8], and released the results to CTAN shortly before the conference.

Concerning the production of new 8-bit math fonts, he suggested concentrating on the Computer Modern version which appears to be the easiest one to start with. In particular, he proposed to start by de-virtualizing the present implementation, which happens to draw characters from a number of base fonts, so as to have a real METAFONT font that could be converted more easily with MetaFog.

New symbols from the STIX collection could then be added by new METAFONT designs, which shouldn't be too difficult to develop in most cases, as many symbols can be realized by combinations or variations of existing symbols.

On the other hand, there seems to be little that can be done about the versions based on commercial font sets such as MathTime or Lucida New Math, which will probably remain restricted to whatever symbol complement is provided in the present versions of the base fonts, unless the suppliers of these font sets will invest some work themselves.

Another suggestion also discussed was to have a set of 8-bit fonts serving as glyph containers organized by types of symbols, which could either be used as the basis for a virtual font implementation of 8-bit math fonts, meeting the technical constraints of TEX, or combined into a single huge 16-bit math font for Omega. While this might be an interesting option for the future, it was pointed out by several participants that neither Omega nor virtual fonts could be assumed to be available everywhere and that a straightforward METAFONT implementation of new 8-bit fonts for TEX was still needed.

Finally, it was discussed what to do about the Plain TEX support of the new math fonts. Since Kluwer Academic Publishers are using Hans Hagen's ConTEXt package, which happens to be based on Plain TEX, Taco will take care of this task as well, since he will need it for his own work.

A suggestion to use the existing LATEX support on top of a Plain TEX emulation of the NFSS interface was rejected, since the LATEX-like syntax doesn't easily fit into the framework of the ConTEXt system, so a low-level Plain TEX interface is preferred.

4 Summary and Conclusions

In summary, one might say that the EuroTEX '98 conference was a great success for math font-related activities in that it helped to bring together members of several working groups and other interested parties, who so far have been working on closely related topics independently of each other.

In particular, the STIX project provided a lot of input to the 8-bit math font encodings for TEX, while on the other hand there was also some feedback to the STIX project in the form of additional symbols that have been missed so far.

While the STIX project will continue to work on getting their list of symbols assigned to Unicode, primarily in support of MathML and SGML-based authoring systems, the Math Font Group will continue to work on completing the implementation of new 8-bit math fonts for (LA)TEX, hopefully by the end of this year.

Current plans include starting with the development of the Plain TEX interface and a de-virtualized METAFONT implementation of the Computer Modern version as soon as possible and to begin adding more symbols once this is in place.

It is hoped that we will be able to provide at least one very comprehensive implementation of new 8-bit math fonts (including the STIX glyphs) in Computer Modern style in both METAFONT and Type 1 formats. In addition, we weill provide several partial implementations for other font families such as MathTime and Lucida New Math, which will be implemented as virtual fonts based on the symbol complement provided by these font sets.

References

1. Home Page of the Math Font Group (MFG). http://www.tug.org/twg/mfg/. Includes complete archives of test releases, discussion papers and mailing list traffic.
2. STIPUB Working Group. STIX symbol tables. http://www.ams.org/STIX/
3. Unicode Consortium. Unicode 2.0 symbol tables. http://www.unicode.org/Unicode.charts/
4. W3C Math Working Group. MathML symbol tables. http://www.w3.org/Math/pdf/mathchars.pdf
5. W3C Math Working Group. W3C Official Recommendation: Mathematical Markup Language 1.0. http://www.w3.org/TR/REC-MathML/
6. Matthias Clasen and Ulrik Vieth. Towards a new math font encoding for (LA)TEX. *Cahiers GUTenberg*, 28–29:94–121, 1998. Proceedings of the 10th European TEX Conference.
7. Alan Jeffrey. Math font encodings: A workshop summary. *TUGboat*, 14(3):293–295, 1993.
8. Richard J. Kinch. MetaFog: Converting METAFONT shapes to contours. *TUGboat*, 16(3):233–243, 1995.

Bijlage 24

Summary of indexing-related activities at EuroTEX '98

Roger Kehr
Computer Science Department
Darmstadt University of Technology
kehr@iti.informatik.tu-darmstadt.de

editor's note
This is an adaptation of the ascii file posted to the xindy newsgroup by Roger in the week following EuroTEX.

As some of you may have noticed, I gave a talk together with Joachim about xindy at the EuroTEX'98 conference in St. Malo, March, 30th.

On March, 31st we had a BOF session about xindy where we discussed several issues how xindy can be further improved and extended.

The most important results from this session and some other issues are as follows:

- Improvement of LATEX support in the future. This mostly concerns a LATEX style file that writes the index entries into the .aux files. Additionally, we think about LATEX macros that write information about the document itself (such as encodings and language) into the .aux file as well. This enables one to add index style definitions into the LATEX source, instead of a completely separate index style file, which seems to be one of the greatest hurdles for the acceptance of xindy.
- Support for input filters. Many have asked for such a feature. This allows to tag index entries from the raw index with an additional atttribute, indicating the index class the index entry belongs to. xindy will be extended to accept only those index entries that belong to the selected classes.
  My current suggestion is to add a new option :CLASS <string> to the INDEXENTRY command to indicate the class of an index entry. Additionally, we need another index style command such as (INPUT-FILTER (<list-of-class-names>)) that defines the set of class names to accept. This specification should be made available to the command line options of xindy as well.
- Hans Hagen asked for a way to pass data attachments through the indexing process into the tagged index. This would enable xindy to process glossaries (with the glossary text as a data attachment) and sort the entries with xindy.
  Several small problems have to be solved to implement such a scheme. My suggestion is to add data in the form of a dictionary of keyword/value pairs in the raw index interface. The question is whether data should be attached to location references or index entries or both. Any ideas are welcome.
  This dictionary asks how the mark-up of this data in the backend should be implemented. I have some ideas how to realize this, but this will need some time.
- Add new hooks in the sorting process to allow sorting rules that can be applied only once at the beginning of the rewriting procedure.
- Further simplification of the installation procedure.
- Re-structuring of the documentation.
- many other things to do . . .

Hope, this gives you an impression what is planned for the future.

The $\mathcal{NTS}$ Team
Peter Breitenlohner
Max-Planck-Institut für
Physik, München

Bijlage 25
The ε-TEX manual, Version 2, February 1998

abstract
The preparation of this report was supported in part by DANTE, Deutschsprachige Anwendervereinigung TEX e.V. 'TEX' is a trademark of the American Mathematical Society.

1 Introduction

The $\mathcal{NTS}$ project intends to develop an 'New Typesetting System' ($\mathcal{NTS}$) that will eventually replace today's TEX3. The $\mathcal{NTS}$ program will include many features missing in TEX, but there will also exist a mode of operation that is 100% compatible with TEX3. It will, necessarily, require quite some time to develop $\mathcal{NTS}$ to maturity and make it widely available.

Meanwhile ε-TEX intends to fill the gap between TEX3 and the future $\mathcal{NTS}$. It consists of a series of features extending the capabilities of TEX3.[1]

Since compatibility between ε-TEX and TEX3 has been a main concern, ε-TEX has two modes of operation:

1. In TEX compatibility mode it fully deserves the name TEX and there are neither extended features nor additional primitive commands. That means in particular that ε-TEX passes the `TRIP` test [1] without any restriction. There are, however, a few minor modifications that would be legitimate in any implementation of TEX.
2. In extended mode there are additional primitive commands and the extended features of ε-TEX are available.

We have tried to make ε-TEX as compatible with TEX as possible even in extended mode. In a few cases there are, however, some subtle differences described in detail later on. Therefore the ε-TEX features available in extended mode are grouped into two categories:

1. Most of them have no semantic effect as long as none of the additional primitives are executed; these 'extensions' are permanently enabled.
2. The remaining optional ε-TEX features ('enhancements') can be individually enabled and disabled; initially they are all disabled. For each enhancement there is a state variable `\...state`; an enhancement is enabled or disabled by assigning a positive or non-positive value respectively to that state variable.

For ε-TEX Versions 1 and 2 there is just one enhancement: mixed direction typesetting (TEX--XET) with the state variable `\TeXXeTstate`.

Version 1.1 of ε-TEX was released in November 1996, Version 2.0 in February 1998. It is expected that there will be about one ε-TEX version per year, where each later version adds new features. It would be desirable if these ε-TEX versions were incorporated into many of the existing implementations of TEX3 without much delay.

1. The TEX3 program; for the moment there are no plans to extend the software related to TEX.

With each ε-TEX version there will be an `e-TRIP` test [2] in order to help to verify that a particular implementation deserves the name ε-TEX in the same way as the `TRIP` test [1] helps to verify that an implementation deserves the name TEX.

2 Generating ε-TEX

2.1 Generating the ε-TEX Program An implementation of TEX consists of a WEB change file `tex.ch` containing all system-dependent changes for a particular system. The WEB system program `TANGLE` applies this change file to the system-independent file `tex.web` defining the TEX program in order to generate a TEX Pascal file for that system [3]. Similarly an implementation of ε-TEX consists of a system-dependent change file `etex.sys` to be applied to the system-independent file `e-tex.web` defining the ε-TEX program. Since ε-TEX differs from TEX by a relatively small fraction of its code `e-tex.web` does, however, not exist as a physical file; it is instead defined in terms of a system-independent change file `e-tex.ch` to be applied to `tex.web`. Similarly it should be possible to define the system-dependent change file `etex.sys` for a particular system in terms of its deviations from the corresponding file `tex.ch` [4].

2.2 Generating Format Files for ε-TEX When (the INITEX or VIRTEX version of) the TEX program is started, it analyzes the first non-blank input line from the command line or (with the `**` prompt) from the terminal: The first non-blank character of that input line may be an `&` followed immediately by the name of the format to be loaded; otherwise VIRTEX uses a default format whereas INITEX starts without loading a format file.

For eINITEX (the INITEX version of ε-TEX) there is an additional possibility: If the first non-blank input character is an `*` (immediately followed what would be the first non-blank input character for INITEX), the program starts in extended mode without loading a format file. If the first non-blank character is neither `&` nor `*` then eINITEX starts without loading a format but in compatibility mode. Whenever a format file is loaded by eINITEX or eVIRTEX the mode (compatibility or extended) is inherited from the format.

It is recommended that the input file `etex.src` be used instead of `plain.tex` when generating an ε-TEX format in extended mode. That file will first read `plain.tex` (without reading `hyphen.tex`) and will then supply macro definitions supporting ε-TEX features.

3 ε-TEX Extensions

3.1 Compatibility and Extended Mode Once ε-TEX has entered compatibility mode it behaves as any other implementation of TEX. All of ε-TEX's additional commands are absent; it is therefore impossible to access any of the extensions or enhancements. The ability of eINITEX to initially choose between compatibility and extended mode is, however, by itself a feature not present in any TEX implementation.

The remainder of this document is devoted to a detailed and mostly technical description of all aspects where ε-TEX (in extended mode) behaves differently from TEX. It will be assumed that the reader is familiar with *The TEXbook* [5] describing TEX's behaviour in quite some detail.

All of ε-TEX's extensions and enhancements available in extended mode are activated by either executing some new primitive command or by assigning a nonzero value to some new integer parameter or state variable. Since all these new variables are initially zero,[2] ε-TEX behaves as TEX as long as none of ε-TEX's new control sequences are used, with the following exceptions which should, however, have no effect on the typesetting of error-free TEX documents (produced with error-free formats):

2. To be precise all state variables are zero when eINITEX or eVIRTEX is started; integer parameters that are not state variables are zero when eINITEX is started without loading a format file or inherited from the format file otherwise.

1. When \tracingcommands has a value of 3 or more, or
when \tracinglostchars has a value of 2 or more, ε-TEX will display additional information not available in TEX.
2. When using a count, dimen, skip, muskip, box, or token register number in the range 256–32767, ε-TEX will access one of its additional registers whereas TEX will produce an error and use register number zero.

3.2 Optimization When a value is assigned to an ⟨internal quantity⟩ within a save group, the former value is restored when the group ends, provided the assignment was not global. This is achieved by saving the former value on TEX's 'save stack'. ε-TEX refrains from creating such save stack entries when the old and new value are the same ('reassignments').

\aftergroup tokens are also kept on TEX's save stack. When the current group ends, TEX converts each \aftergroup token into a token list and inserts this list as new 'input level' into the input stack. ε-TEX collects all \aftergroup tokens from one group into one token list and thus conserves input levels.

When a completed page is written to the DVI file (shipped out), TEX multiplies the relevant stretch or shrink components of glue nodes in a box by the glue expansion factor of that box and converts the product to DVI units. In order to avoid overflow each resulting value x is artificially limited to the range $|x| \leq 10^9$. Consider the example:

```
\shipout\vbox to100pt{
  \hrule width10pt
  \vskip 0pt plus1000fil
  \vskip 0pt plus1000fil
  \vskip 0pt plus-2000fil
  \hrule
  \vskip 0pt plus0.00005fil
  }
```

Here the three glues between the two rules add up to zero; when TEX converts each stretch component individually they will, however, add up to 10^9 DVI units due to the truncation mentioned above. ε-TEX, however, accumulates the relevant stretch or shrink components of consecutive glue nodes (possibly separated by insert, mark, adjust, kern, and penalty nodes) before converting them to DVI units. During this process glue nodes may be converted into equivalent kern nodes and some glue specifications may be recycled; this may affect the memory usage statistics displayed after the page has been shipped out.

3.3 Tracing and Diagnostics When \tracingcommands has a value of 3 or more, the commands following a prefix (\global, etc.) are shown as well, e.g.:

```
\global\count0=0     =>     {\global}
                            {\count}
```

When \tracinglostchars has a value of 2 or more, missing characters are displayed on the terminal even if the value of \tracingonline is 0 or less.

When \tracingscantokens has a value of 1 or more, the opening and closing of pseudo-files (generated by \scantokens) is recorded as for any other file, with '' as filename.

When the program is compiled with the code for collecting statistics and \tracingassigns has a value of 1 or more, all assignments subject to TEX's grouping mechanism are traced, e.g.:

```
\def\foo{\relax}     =>     {changing \foo=undefined}
```

```
                          {into \foo=macro:->\relax }
\global\count17=7   =>    {globally changing \count17=0}
                          {into \count17=7}
\count17=7          =>    {reassigning \count17=7}
```

When `\tracingifs` has a value of 1 or more, all conditionals (including `\unless`, `\or`, `\else`, and `\fi`) are traced, together with the starting line and nesting level; the `\showifs` command displays the state of all currently active conditionals. Thus the input

```
\unless\iffalse
   \iffalse
   \else
      \showifs
   \fi
\fi
```

might yield

```
{\unless\iffalse: (level 1) entered on line 1}
{\iffalse: (level 2) entered on line 2}
{\else: \iffalse (level 2) entered on line 2}
### level 2: \iffalse\else entered on line 2
### level 1: \unless\iffalse entered on line 1
{\fi: \iffalse (level 2) entered on line 2}
{\fi: \unless\iffalse (level 1) entered on line 1}
```

When `\tracinggroups` has a value of 1 or more, the start and end of each save group is traced, together with the starting line and grouping level; the `\showgroups` command displays the state of all currently active save groups. Thus the input

```
\begingroup
   {
      \showgroups
   }
\endgroup
```

might yield

```
{entering semi simple group (level 1) at line 1}
{entering simple group (level 2) at line 2}
### simple group (level 2) entered at line 1 ({)
### semi simple group (level 1) entered at line 1 (\begingroup)
### bottom level
{leaving simple group (level 2) entered at line 2}
{leaving semi simple group (level 1) entered at line 1}
```

Occasionally conditionals and/or save groups are not properly nested with respect to `\input` files. Although this might be perfectly legitimate, such anomalies are mostly unintentional and may cause quite obscure errors. When `\tracingnesting` has a value of 1 or more, these anomalies are shown; when `\tracingnesting` has a value of 2 or more, the current context (traceback) is shown as well. Thus the input

```
\newlinechar=`\^^J
\begingroup
   \iftrue
      \scantokens{%
```

```
   \endgroup
^^J\fi
^^J\bgroup
   ^^\tracingnesting=2
   ^^J\iffalse
   ^^J\else
     }%
  \egroup
\fi
```

might yield[3]

```
Warning: end of semi simple group (level 1) entered at line 2 of a diff
  erent file
Warning: end of \iftrue entered on line 3 of a different file
Warning: end of file when simple group (level 1) entered at line
  3 is incomplete
Warning: end of file when \iffalse\else entered on line 5 is inc
  omplete
l.7 \else

l.11      }
           %
```

The command `\showtokens{`⟨token list⟩`}` displays the token list, and allows the display of quantities that cannot be displayed by `\show` or `\showthe`, e.g.:

```
\showtokens\expandafter{\jobname}
\showtokens\expandafter{\topmarks 27}
```

3.4 Status Enquiries A number of TEX's internal quantities can be assigned values but these values cannot be retrieved in TEX. ε-TEX introduces several new primitives that allow the retrieval of information about its internal state.
`\eTeXversion` returns ε-TEX's (major) version number;
`\eTeXrevision` expands into a list of character tokens representing the revision (minor version) number. Thus

```
\message{\number\eTeXversion\eTeXrevision}
```

should write the complete version as shown when ε-TEX is started.
When used as number, `\interactionmode` returns one of the values 0 (batchmode), 1 (nonstopmode), 2 (scrollmode), or 3 (errorstopmode). Assigning one of these values to `\interactionmode` changes the current interaction mode accordingly; such assignments are always global.
`\currentgrouplevel` returns the current save group level;
`\currentgrouptype` returns a number representing the type of the innermost group:

0:	bottom level (no group)	9:	math group
1:	simple group	10:	disc group
2:	hbox group	11:	insert group
3:	adjusted hbox group	12:	vcenter group
4:	vbox group	13:	math choice group
5:	vtop group	14:	semi simple group
6:	align group	15:	math shift group

3. The `\scantokens` command will be discussed later.

7: no align group
8: output group
16: math left group

`\currentiflevel` returns the number of currently active conditionals;
`\currentifbranch` indicates which branch of the innermost conditional is taken: 1 'then branch', −1 'else branch', or 0 not yet decided;
`\currentiftype` returns 0 if there are no active conditionals, a positive number indicating the type of the innermost active conditional, or the negative of that number when the conditional was prefixed by `\unless`:

1:	`\if`	8:	`\ifmmode`	15:	`\iftrue`
2:	`\ifcat`	9:	`\ifinner`	16:	`\iffalse`
3:	`\ifnum`	10:	`\ifvoid`	17:	`\ifcase`
4:	`\ifdim`	11:	`\ifhbox`	18:	`\ifdefined`
5:	`\ifodd`	12:	`\ifvbox`	19:	`\ifcsname`
6:	`\ifvmode`	13:	`\ifx`	20:	`\iffontchar`
7:	`\ifhmode`	14:	`\ifeof`		

`\lastnodetype` returns a number indicating the type of the last node, if any, on the current (vertical, horizontal, or math) list:

-1:	none (empty list)	8:	disc node
0:	char node	9:	whatsit node
1:	hlist node	10:	math node
2:	vlist node	11:	glue node
3:	rule node	12:	kern node
4:	ins node	13:	penalty node
5:	mark node	14:	unset node
6:	adjust node	15:	math mode nodes
7:	ligature node		

The commands `\fontcharht`, `\fontcharwd`, `\fontchardp`, and `\fontcharic` followed by a font specification and a character code, return a dimension: the height, width, depth, or italic correction of the character in the font, or `0pt` if no such character exists; the conditional `\iffontchar` tests the existence of that character.
When used as number, `\parshape` returns the number of lines of the current parshape specification (or zero).
ε-TEX's `\parshapeindent`, `\parshapelength`, and `\parshapedimen`, followed by a number n return the dimensions of the parshape specification:
`0pt` for $n \leq 0$ or when no parshape is currently active, otherwise
`\parshapeindent`n and `\parshapedimen`$2n - 1$ both return the indentation of line n (explicitly specified or implied by repeating the last specification),
`\parshapelength`n and `\parshapedimen`$2n$ both return the length of line n.

3.5 Expressions ε-TEX introduces the notion of expressions of type number, dimen, glue, or muglue, that can be used whenever a quantity of that type is needed. Such expressions are evaluated by ε-TEX's scanning mechanism; they are initiated by one of the commands `\numexpr`, `\dimexpr`, `\glueexpr`, or `\muexpr` (determining the type t) and optionally terminated by one `\relax` (that will be absorbed by the scanning mechanism). An expression consists of one or more terms of the same type to be added or subtracted; a term of type t consists of a factor of that type, optionally multiplied and/or divided by numeric factors; finally a factor of type t is either a parenthesized subexpression or a quantity (number, etc.) of that type. Thus, the conditional

```
\ifdim\dimexpr (2pt-5pt)*\numexpr 3-3*13/5\relax + 34pt/2<\wd20
```

is true if and only if the width of box 20 exceeds 32 `pt`. Note the use of `\relax` to terminate the inner (numeric) expression, the outer (dimen) expression is terminated automatically by the token $<_{12}$ that does not fit into the expression syntax.

The arithmetic performed by ε-TEX's expressions does not do much that could not be done by TEX's arithmetic operations `\advance`, `\multiply`, and `\divide`, although there are some notable differences: Each factor is checked to be in the allowed range, numbers must be less than 2^{31} in absolute value, dimensions or glue components must be less than 2^{14} `pt`, `mu`, `fil`, etc. respectively. The arithmetic operations are performed individually, except for 'scaling' operations (a multiplication immediately followed by a division) which are performed as one combined operation with a 64-bit product as intermediate value. The result of each operation is again checked to be in the allowed range. Finally the results of divisions and scalings are rounded, whereas TEX's `\divide` truncates.

The important new feature is, however, that the evaluation of expressions does not involve assignments and can therefore be performed in circumstances where assignments are not allowed, e.g., inside an `\edef` or `\write`. This also allows the definition of purely expandable loop constructions:

```
\def\foo#1#2{\number#1
  \ifnum#1<#2,
    \expandafter\foo
    \expandafter{\number\numexpr#1+1\expandafter}%
    \expandafter{\number#2\expandafter}%
  \fi}
```

such that, e.g., '`\foo{7}{13}`' expands into '`7, 8, 9, 10, 11, 12, 13`'.

The commands `\gluestretch` and `\glueshrink` are to be followed by a glue specification and return the stretch or shrink component of that glue as dimensions (with `fil` etc. replaced by `pt`), the commands `\gluestretchorder` and `\glueshrinkorder` return the order of infinity: 0 for `pt`, 1 for `fil`, 2 for `fill`, and 3 for `filll`.

The commands `\gluetomu` and `\mutoglue` convert glue into muglue and vice versa by simply equating 1 `pt` with 1 `mu`, precisely what TEX does (in addition to an error message) when the wrong kind of glue is used.

3.6 Additional Registers and Marks ε-TEX increases the number of TEX's count, dimen, skip, muskip, box, and token registers from 256 to 32768. The additional registers, numbered 256–32767, can be used exactly as the first 256, except that they can not be used for insertion classes.

As in TEX, the first 256 registers of each kind are realized as static arrays that are part of the 'table of equivalents'; values to be restored when a save group ends are kept on the save stack. The additional registers are realized as sparse arrays built from TEX's main memory and are therefore less efficient. They use a four-level index structure and individual registers are present only when needed. Values to be restored when a particular save group ends are kept in a linked list (again built from main memory) with one save stack entry pointing to that list.[4]

ε-TEX generalizes TEX's mark concept to mark classes 0–32767, with mark class 0 used for TEX's marks.
The command `\marks` followed by a mark class *n* and a mark text appends a mark node to the current list; `\marks0` is synonymous with `\mark`. The page builder and the `\vsplit` command record information about the mark nodes found on the page or box produced, separately for each mark class. The information for mark class 0 is kept in a small static

4. With the effect that the order of restoring (or discarding) saved values may be somewhat surprising.

array as in TEX, the information for the additional mark classes is again kept in a sparse array with entries present only when needed.
The command `\firstmarks`*n* expands to the mark text for mark class *n* first encountered on the most recent page, etc., and again `\firstmarks0` is synonymous with `\firstmark`.

3.7 Input Handling The command `\readline`⟨number⟩ `to` ⟨control sequence⟩ defines the control sequence as parameterless macro whose replacement text is the contents of the next line read from the designated file, as for `\read`. The difference is that the current category codes are ignored and all characters on that line (including an endline character) are converted to character tokens with category 12 ('other'), except that the character code 32 gets category 10 ('space').

The command `\scantokens{...}` absorbs a list of unexpanded tokens, converts it into a character string that is treated as if it were an external file, and starts to read from this 'pseudo-file'. A rather similar effect can be achieved by the commands

```
\toks0={...}
\immediate\openout0=file
\immediate\write0{\the\toks0}
\immediate\closeout0
\input file
```

In particular every occurrence of the current newline character is interpreted as start of a new line, and input characters will be converted into tokens as usual. The `\scantokens` command is, however, expandable and does not use token registers, write streams, or external files. Furthermore the conversion from TEX's internal ASCII codes to external characters and back to ASCII codes is skipped. Finally the current context (traceback) shown, e.g., as part of an error message continues beyond an input line from a pseudo-file until an input line from a real file (or the terminal) is found.

When ε-TEX's input mechanism attempts to read beyond the end of an `\input` file or `\scantokens` pseudo-file, and before checking for 'runaway' conditions and closing the file, it will first read a list of tokens that has been predefined by the command `\everyeof={`⟨token list⟩`}`.

3.8 Breaking Paragraphs into Lines Traditional typesetting with lead type used to adjust (stretch or shrink) the interword spaces in the last line of a paragraph by the same amount as those in the preceding line. With TEX the last line is, however, usually typeset at its natural width due to infinitely stretchable parfillskip glue. ε-TEX allows interpolation between these two extremes by specifying a suitable value for `\lastlinefit`. For a value of 0 or less, ε-TEX behaves as TEX, values from 1 to 1000 indicate a glue adjustment fraction f times 1000, values above 1000 are interpreted as $f = 1$.

The new algorithm is used only if

1. `\lastlinefit` is positive;
2. `\parfillskip` has infinite stretchability; and
3. the stretchability of `\leftskip` plus `\rightskip` is finite.[5]

Thus the last line of a paragraph would normally be typeset at its natural width and the stretchability of parfillskip glue would be used to achieve the desired line width. The algorithm proceeds as usual, considering all possible sequences of feasible break points and accumulating demerits for the stretching or shrinking of lines as well as for visually incompatible lines. When a candidate for the last line has been reached, the following conditions are tested:

5. As usual for parameters influencing TEX's line-breaking algorithm, the values current at the end of the (partial) paragraph are used.

4. the previous line was not ‘infinitely bad’ and was stretched with positive finite stretchability or was shrunk with positive shrinkability;
5. the last line has infinite stretchability entirely due to parfillskip glue;
6. if the previous line was stretched or shrunk the last line has positive finite stretchability or shrinkability respectively.

If all three conditions are satisfied, a glue adjustment factor of f times that of the preceding line will be applied to the relevant stretch or shrink components of all glue nodes in the last line, and the corresponding demerits are computed. (The last line will, however, not be stretched beyond the desired line width.)

When all possible candidates for the last line of the paragraph have been examined, the one having fewest accumulated demerits is chosen. If ε-TEX's modified algorithm was applied to that last line, the actual stretching or shrinking is achieved by suitably modifying the parfillskip glue node.

All computations described so far are performed with machine-independent integer arithmetic. Note, however, that the actual stretching requires machine-dependent floating point arithmetic. Therefore, when a paragraph is interrupted by a displayed equation and the line preceding the display is subject to the adjustment just described, the display will in general be preceded by abovedisplayskip and not by abovedisplayshortskip glue.

After breaking a paragraph into lines, TEX computes the interline penalties by adding the values of:

`\interlinepenalty` between any two lines,
`\clubpenalty` after the first line of a (partial) paragraph,
`\widowpenalty` before the last line of the paragraph,
`\displaywidowpenalty` before the line immediately preceding a displayed equation, and
`\brokenpenalty` after lines ending with a discretionary break.

ε-TEX generalizes the concept of interline, club, widow, and display widow penalty by allowing their replacement by arrays of penalty values with the commands

`\interlinepenalties`,
`\clubpenalties`,
`\widowpenalties`, and
`\displaywidowpenalties`.

Each of these commands is to be followed by an optional equal sign and a number n. If $n \leq 0$ the respective array is reset and TEX's corresponding single value is used as usual; a positive value n declares an array of length n and must be followed by n penalty values. When one of these arrays has been set, its values are used instead of TEX's corresponding single values as follows (repeating the last value when necessary):

the i^{th} interline penalty value is used after line i of the paragraph;
the i^{th} club penalty value is used after line i of a partial paragraph;
the i^{th} widow penalty value is used after line $m - i$ of a paragraph without displayed equations or the last partial paragraph of length m;
the i^{th} display widow penalty value is used after line $m - i$ of a partial paragraph of length m that is followed by a displayed equation.

When used after `\the` or in situations where TEX expects to see a number, the same four commands serve to retrieve the arrays of penalties. Specifying, e.g.,

\clubpenalties⟨number⟩ with a number n, returns 0 for $n < 0$ or when the club penalty array has been reset, the length of the declared club penalty array for $n = 0$, or the n^{th} club penalty value for $n > 0$ (again repeating the last value when necessary).

3.9 Math Formulas TEX's \left⟨delimiter⟩ ... \right⟨delimiter⟩ produces two delimiters with a common size adjusted to the height and depth of the enclosed material. In ε-TEX this can be generalized by occurrences of \middle⟨delimiter⟩ dividing the enclosed material into segments resulting in a sequence of delimiters with a common size adjusted to the maximal height and depth of all enclosed segments. The spacing between a segment and the delimiter to its left or right is as for TEX's left or right delimiter respectively.

3.10 Hyphenation TEX uses the \lccode values for two quite unrelated purposes:

1. when \lowercase converts character tokens to their lower-case equivalents (in the same way as \uppercase uses the \uccode values); and
2. when hyphenation patterns or exceptions are read, and when words are hyphenated during the line-breaking algorithm.

ε-TEX introduces the concept of (language-dependent) hyphenation codes that are used instead of the \lccode values for hyphenation purposes. In order to explain the details of ε-TEX's behaviour, we need some technical aspects of hyphenation patterns. When INITEX starts without reading a format file, the (initially empty) hyphenation patterns are in a form suitable for inserting new patterns specified by \patterns commands; when INITEX attemps hyphenation or prepares to write a format file, they are compressed into a more compact form suitable for finding hyphens. Only these compressed patterns can be read from a format file (by INITEX or VIRTEX).

In ε-TEX the hyphenation patterns are supplemented by hyphenation codes. When eINITEX starts without reading a format file both are initially empty; when a \patterns command is executed and \savinghyphcodes has a positive value, the current \lccode values are saved as hyphenation codes for the current language. These saved hyphenation codes are later compressed together with the patterns and written to or read from a format file. When the patterns have been compressed (always true for eVIRTEX) and hyphenation codes have been saved for the current language, they are used instead of the \lccode values for hyphenation purposes (reading hyphenation exceptions and hyphenating words).

3.11 Discarded Items When TEX's page builder transfers (vertical mode) material from the 'recent contributions' to the 'page so far', it discards glue, kern, and penalty nodes (discardable items) preceding the first box or rule on the page under construction and inserts a topskip glue node immediately before that box or rule. Note, however, that this topskip glue need not be the first node on the page, it may be preceded by insertion, mark, and whatsit nodes. Similarly when the \vsplit command has split the first part off a vbox, discardable items are discarded from the top of the remaining vbox and a splittopskip glue node is inserted immediately before the first box or rule.

When ε-TEX's parameter \savingvdiscards has been assigned a positive value, these 'discarded items' are saved in two lists and can be recovered by the commands \pagediscards and \splitdiscards that act like 'unvboxing' hypothetical box registers containing a vbox with the discarded items.

The list of items discarded by the page builder is emptied at the end of the output routine and by the \pagediscards command; new items may be added as long as the new 'page so far' contains no box or rule.

The list of items discarded by the \vsplit command is emptied at the start of a vsplit operation and by the \splitdiscards command; new items are added at the end of a vsplit operation.

3.12 Expandable Commands Chapter 20 of *The TEXbook* gives complete lists of all expandable TEX commands and of all cases where expandable tokens are not expanded. For ε-TEX there are these additional conditionals:

- `\ifdefined`⟨token⟩ (test if token is defined)

True if ⟨token⟩ is defined; creates no new hash table entry.

- `\ifcsname...\endcsname` (test if control sequence is defined)

True if the control sequence `\csname...\endcsname` would be defined; creates no new hash table entry.

- `\iffontchar`⟨font⟩ ⟨8-bit number⟩ (test if char exists)

  True if `\char`⟨8-bit number⟩ in `\font`⟨font⟩ exists.

These are ε-TEX's additional expandable commands:

- `\unless`.
  The next (unexpanded) token must be a boolean conditional (i.e., not `\ifcase`); the truth value of that conditional is reversed.
- `\eTeXrevision`.
  The expansion is a list of character tokens of category 12 ('other') representing ε-TEX's revision (minor version) number, e.g., '.0' or '.1'.
- `\topmarks`⟨15-bit number⟩, `\firstmarks`⟨15-bit number⟩, `\botmarks`⟨15-bit number⟩, `\splitfirstmarks`⟨15-bit number⟩, and `\splitbotmarks`⟨15-bit number⟩.
  These commands generalize TEX's `\topmark` etc. to 32768 distinct mark classes; the special case `\topmarks0` is synonymous with `\topmark` etc.
- `\unexpanded`⟨general text⟩.
  The expansion is the token list ⟨balanced text⟩.
- `\detokenize`⟨general text⟩.
  The expansion is a list of character tokens representing the token list ⟨balanced text⟩. As with the lists of character tokens produced by TEX's `\the` and ε-TEX's `\readline`, these tokens have category 12 ('other'), except that the character code 32 gets category 10 ('space').
- `\scantokens`⟨general text⟩.
  The expansion is null; but ε-TEX creates a pseudo-file containing the characters representing the token list ⟨balanced text⟩ and prepares to read from this pseudo-file before looking at any more tokens from its current source.

These are the additional ε-TEX cases when expandable tokens are not expanded:

- When ε-TEX is reading the argument token for `\ifdefined`.
- When ε-TEX is absorbing the token list for `\unexpanded`, `\detokenize`, `\scantokens`, or `\showtokens`.
- Protected macros (defined with the `\protected` prefix) are not expanded when building an expanded token list (for `\edef`, `\xdef`, `\message`, `\errmessage`, `\special`, `\mark`, `\marks` or when writing the token list for `\write` to a file) or when looking ahead in an alignment for `\noalign` or `\omit`.[6]

6. Whereas protected macros were introduced with ε-TEX Version 1, suppression of their expansion in alignments was introduced with Version 2.

- When building an expanded token list, the tokens resulting from the expansion of `\unexpanded` are not expanded further (this is the same behaviour as is exhibited by the tokens resulting from the expansion of `\the`⟨token variable⟩ in both TEX and ε-TEX).

4 ε-TEX Enhancements

The execution of most new primitives related to enhancements is disallowed when the corresponding enhancement is currently disabled and will lead to an '`Improper...`' error message. The offending command may nevertheless already have had some effect such as, e.g., bringing ε-TEX into horizontal mode.

4.1 Mixed-Direction Typesetting This feature supports mixed left-to-right and right-to-left typesetting and introduces the four text-direction primitives `\beginL`, `\endL`, `\beginR`, and `\endR`. The code is inspired by but different from TEX-XET [6].

In order to avoid confusion with TEX-XET the present implementation of mixed-direction typesetting is called TEX--XET. It uses the same text-direction primitives, but differs from TEX-XET in several important aspects:

1. Right-to-left text is reversed explicitly by ε-TEX and is written to a normal DVI file without any `begin_reflect` or `end_reflect` commands;
2. a math node is (ab)used instead of a whatsit node to record the text-direction primitives in order to minimize the influence on the line-breaking algorithm for pure left-to-right text;
3. right-to-left text interrupted by a displayed equation is automatically resumed after that equation;
4. display math material is always printed left-to-right, even in constructions such as:

```
\hbox{\beginR\vbox{\noindent$$abc\eqno(123)$$}\endR}
```

TEX--XET is enabled or disabled by assigning a positive or non-positive value respectively to the `\TeXXeTstate` state variable. As long as TEX--XET is disabled, ε-TEX and TEX3 build horizontal lists and paragraphs in exactly the same way. Even TEX--XET will, in general, produce the same results as TEX3 for pure left-to-right text. There are, however, circumstances where some differences may arise. This is best illustrated by an example:

```
\vbox{\noindent
   $\hfil\break
   \null\hfil\break
   \null$\par
```

Here TEX will produce three lines containing the following nodes:

1. mathon, hfil glue, break penalty, and rightskip glue;
2. empty hbox, hfil glue, break penalty, and rightskip glue;
3. empty hbox, mathoff, nobreak penalty, parfillskip glue, and rightskip glue.

These lines can be retrieved via:

```
\setbox3=\lastbox
\unskip\unpenalty
\setbox2=\lastbox
\unskip\unpenalty
\setbox1=\lastbox
```

Later on these lines can be 'unhboxed' as part of a new paragraph and possibly their contents analyzed. As a consequence in TEX (and ε-TEX in compatibility mode) there may be horizontal lists where mathon and mathoff nodes are not properly paired. Therefore TEX might attempt hyphenation of 'words' originating from math mode or prevent hyphenation of words originating from horizontal mode.

Math-mode material is always typeset left-to-right by TEX--XETEX, even when it is contained inside right-to-left text. Therefore TEX--XETEX will insert additional `beginM` and `endM` math nodes such that material originating from math mode is always enclosed between properly paired math nodes. Consequently TEX--XETEX will never attempt hyphenation of 'words' originating from math mode nor prevent hyphenation of words originating from horizontal mode.

The additional math nodes introduced by TEX--XETEX are, however, transparent to operations such as `\lastpenalty` that inspect or remove the last node of a horizontal list.[7]

When TEX--XETEX is enabled or disabled during the construction of a box, that box may contain text-direction directives or math nodes that are not properly paired. Such unpaired nodes may cause warning messages when the box is shipped out. It is, therefore, advisable that TEX--XETEX be enabled or disabled only in vertical mode.

5 Syntax Extensions for ε-TEX

5.1 Mode-Independent Commands The syntax for TEX's mode-independent commands, as described in the first part of Chapter 24 of *The TEX book*, is extended by modifications of existing commands as well as by new commands.

First, ε-TEX has 32768 `\count`, `\dimen`, `\skip`, `\muskip`, `\box`, and `\toks` registers instead of TEX's 256. Thus it allows a ⟨15-bit number⟩ instead of an ⟨8-bit number⟩ in almost all syntax constructions referring to these registers; the only exception to this is the `\insert` command: insertion classes are restricted to the range 0–254 in ε-TEX as they are in TEX.

Next, ε-TEX extends the list of TEX's internal quantities:

⟨internal integer⟩ ⟶ whatever *The TEX book* defines | `\eTeXversion`
| `\interactionmode` | ⟨penalties⟩⟨number⟩
| `\lastnodetype` | `\currentgrouplevel` | `\currentgrouptype`
| `\currentiflevel` | `\currentiftype` | `\currentifbranch`
| `\gluestretchorder`⟨glue⟩ | `\glueshrinkorder`⟨glue⟩
| `\numexpr`⟨integer expr⟩⟨optional spaces and `\relax`⟩

⟨penalties⟩ ⟶ `\interlinepenalties` | `\clubpenalties`
| `\widowpenalties` | `\displaywidowpenalties`

⟨internal dimen⟩ ⟶ whatever *The TEX book* defines
| `\parshapeindent`⟨number⟩ | `\parshapelength`⟨number⟩
| `\parshapedimen`⟨number⟩
| `\gluestretch`⟨glue⟩ | `\glueshrink`⟨glue⟩
| `\fontcharht`⟨font⟩⟨8-bit number⟩ | `\fontcharwd`⟨font⟩⟨8-bit number⟩
| `\fontchardp`⟨font⟩⟨8-bit number⟩ | `\fontcharic`⟨font⟩⟨8-bit number⟩
| `\dimexpr`⟨dimen expr⟩⟨optional spaces and `\relax`⟩

⟨internal glue⟩ ⟶ whatever *The TEX book* defines | `\mutoglue`⟨muglue⟩
| `\glueexpr`⟨glue expr⟩⟨optional spaces and `\relax`⟩

⟨internal muglue⟩ ⟶ whatever *The TEX book* defines | `\gluetomu`⟨glue⟩
| `\muexpr`⟨muglue expr⟩⟨optional spaces and `\relax`⟩

The additional possibilities for ⟨integer parameter⟩ are:

7. This was not the case for some earlier TEX--XETEX implementations.

`\TeXXeTstate` (positive if mixed-direction typesetting is enabled)
`\tracingassigns` (positive if showing assignments)
`\tracinggroups` (positive if showing save groups)
`\tracingifs` (positive if showing conditionals)
`\tracingscantokens` (positive if showing the opening and closing of `\scantokens` pseudo-files)
`\tracingnesting` (positive if showing improper nesting of groups and conditionals within files)
`\predisplaydirection` (text direction preceding a display)
`\lastlinefit` (adjustment ratio for last line of paragraph, times 1000)
`\savingvdiscards` (positive if saving items discarded from vertical lists)
`\savinghyphcodes` (positive if `\patterns` saves `\lccode` values as hyphenation codes)

Note that the ε-TEX state variable `\TeXXeTstate` (the only one so far) is an ⟨integer parameter⟩. That need not be the case for all future state variables; it might turn out that some future enhancements can be enabled and disabled only globally, not subject to grouping.
The additional possibilities for ⟨token parameter⟩ are:

`\everyeof` (tokens to insert when an `\input` file ends)

Here is the syntax for ε-TEX's expressions:

⟨integer expr⟩ → ⟨integer term⟩
| ⟨integer expr⟩⟨add or sub⟩⟨integer term⟩
⟨integer term⟩ → ⟨integer factor⟩
| ⟨integer term⟩⟨mul or div⟩⟨integer factor⟩
⟨integer factor⟩ → ⟨number⟩
| ⟨left paren⟩⟨integer expr⟩⟨right paren⟩
⟨dimen expr⟩ → ⟨dimen term⟩
| ⟨dimen expr⟩⟨add or sub⟩⟨dimen term⟩
⟨dimen term⟩ → ⟨dimen factor⟩
| ⟨dimen term⟩⟨mul or div⟩⟨integer factor⟩
⟨dimen factor⟩ → ⟨dimen⟩
| ⟨left paren⟩⟨dimen expr⟩⟨right paren⟩
⟨glue expr⟩ → ⟨glue term⟩
| ⟨glue expr⟩⟨add or sub⟩⟨glue term⟩
⟨glue term⟩ → ⟨glue factor⟩
| ⟨glue term⟩⟨mul or div⟩⟨integer factor⟩
⟨glue factor⟩ → ⟨glue⟩
| ⟨left paren⟩⟨glue expr⟩⟨right paren⟩
⟨muglue expr⟩ → ⟨muglue term⟩
| ⟨muglue expr⟩⟨add or sub⟩⟨muglue term⟩
⟨muglue term⟩ → ⟨muglue factor⟩
| ⟨muglue term⟩⟨mul or div⟩⟨integer factor⟩
⟨muglue factor⟩ → ⟨muglue⟩
| ⟨left paren⟩⟨muglue expr⟩⟨right paren⟩
⟨optional spaces and `\relax`⟩ → ⟨optional spaces⟩
| ⟨optional spaces⟩`\relax`
⟨add or sub⟩ → ⟨optional spaces⟩$+_{12}$ | ⟨optional spaces⟩$-_{12}$
⟨div or mul⟩ → ⟨optional spaces⟩$*_{12}$ | ⟨optional spaces⟩$/_{12}$
⟨left paren⟩ → ⟨optional spaces⟩$(_{12}$

⟨right paren⟩ ⟶ ⟨optional spaces⟩)$_{12}$

Next, ε-TEX extends the syntax for assignments:

⟨prefix⟩ ⟶ whatever *The TEXbook* defines | `\protected`
⟨simple assignment⟩ ⟶ whatever *The TEXbook* defines
| ⟨penalties assignment⟩
| `\readline`⟨number⟩ `to` ⟨control sequence⟩
⟨penalties assignment⟩ ⟶ ⟨penalties⟩⟨equals⟩⟨number⟩⟨penalty values⟩
⟨interaction mode assignment⟩ ⟶ whatever *The TEXbook* defines
| `\interactionmode`⟨equals⟩⟨2-bit number⟩

In a ⟨penalties assignment⟩ for which the ⟨number⟩ is n, the ⟨penalty values⟩ are ⟨empty⟩ if $n \leq 0$, otherwise they consist of n consecutive occurrences of ⟨number⟩.
Finally, the remaining mode-independent ε-TEX commands:

- `\showgroups`, `\showifs`, `\showtokens`⟨general text⟩. These commands are intended to help you figure out what ε-TEX thinks it is doing. The `\showtokens` command displays the token list ⟨balanced text⟩.
- `\marks`⟨15-bit number⟩⟨general text⟩. This command generalizes TEX's `\mark` command to 32768 distinct mark classes; the special case `\marks0` is synonymous with `\mark`.

5.2 Vertical-Mode Commands The syntax for TEX's vertical-mode commands, as described in the second part of Chapter 24 of *The TEXbook*, is extended by ε-TEX as follows:

- `\pagediscards`, `\splitdiscards`. These two commands are similar to `\unvbox`. When `\savingvdiscards` is positive, items discarded by the page builder and by the `\vsplit` command are collected in two special lists. One of these special lists is appended to the current vertical list (in the same way as `\unvbox` appends the vertical list inside a vbox) and becomes empty.
- Here are the additional possibilities for ⟨horizontal command⟩:

⟨horizontal command⟩ ⟶ whatever *The TEXbook* defines
| `\beginL` | `\endL` | `\beginR` | `\endR`

5.3 Horizontal-Mode Commands The syntax for TEX's horizontal-mode commands, as described in Chapter 25 of *The TEXbook*, is extended by ε-TEX as follows:

- Here are the additional possibilities for ⟨vertical command⟩:

⟨vertical command⟩ ⟶ whatever *The TEXbook* defines
| `\pagediscards` | `\splitdiscards`

- `\beginL`, `\endL`, `\beginR`, `\endR` (text-direction commands).
The use of these commands is illegal when the TEX--XET enhancement is currently disabled; otherwise a `beginL`, etc. text-direction node (a new kind of math node) is appended to the current horizontal list. These nodes delimit the beginning and end of hlist segments containing left-to-right (L) or right-to-left (R) text. Before a paragraph is broken into lines, `endL` and `endR` nodes are added to terminate any unfinished L or R segments; when a paragraph is continued after display math mode, any such unfinished segments are automatically resumed, starting the new hlist with `beginL` and `beginR` nodes as necessary.
- `\marks`⟨15-bit number⟩⟨general text⟩. This command generalizes TEX's `\mark` command to 32768 distinct mark classes; the special case `\marks0` is synonymous

with \mark.

5.4 Math-Mode Commands The syntax for TEX's math-mode commands, as described in Chapter 26 of *The TEXbook*, is extended by ε-TEX as follows:

- \left⟨delim⟩ ⟨math mode material⟩
  \middle⟨delim⟩ ⟨math mode material⟩ . . . \right⟨delim⟩
  (generalizing TEX's \left⟨delim⟩ ⟨math mode material⟩ \right⟨delim⟩).
  For each ⟨math mode material⟩ ε-TEX begins a new group, starting out with a new math list (always in the same style) that begins with a left boundary item containing everything processed so far. This group must be terminated with either '\middle' or 'right', at which time the internal math list is completed with a new boundary item containing the new delimiter. In the case of '\middle', a new group is started again, in the case of '\right', ε-TEX appends an Inner atom to the current list; the nucleus of this atom contains the internal math list just completed.

References

[1] *A torture test for TEX*, by Donald E. Knuth, Stanford Computer Science Report 1027.

[2] *A torture test for ε-TEX*, by The NTS Team (Peter Breitenlohner and Bernd Raichle). Version 2, January 1998.

[3] *The WEB system of structured documentation*, by Donald E. Knuth, Stanford Computer Science Report 980.

[4] *How to generate ε-TEX*, by The NTS Team (Peter Breitenlohner and Phil Taylor). Version 2, January 1998.

[5] *The TEXbook* (Computers and Typesetting, Vol. A), by Donald E. Knuth, Addison Wesley, Reading, Massachusetts, 1986.

[6] *Mixing right-to-left texts with left-to-right texts*, by Donald E. Knuth and Pierre MacKay, *TUGboat* **8**, 14–25, 1987.

Bijlage 26

Generating Type 1 Fonts from Metafont Sources

Taco Hoekwater
Kluwer Academic Publishers
Dordrecht
taco.hoekwater@wkap.nl

abstract

This article makes a comparison between bitmapped and vector fonts, and presents some of the problems I encountered when I tried to convert METAFONT sources into PostScript Type 1 fonts

keywords

METAFONT, PostScript, fonts, Type 1, conversion, metafog.

The second part of this article will focus more closely on some of the problems that I faced while trying to convert METAFONTs into PostScript Type 1 fonts, but first some explanation is in order as to why one might want to do this conversion, and precisely what this conversion entails. These topics are the subjects of the first couple of paragraphs.

What are METAFONT Fonts?

How characters are created

I'll assume the reader knows the following: every TEX distribution has a program called METAFONT, that compiles font sources more or less the same way that the TEX program compiles text sources. A major difference between the two programs is that METAFONT produces *device–dependant* output (called `pk` files), whereas TEX produces *device–independant* output (also known as `dvi` files).

Let's look into the font sources that METAFONT uses, and see what kind of information they contain. These are ordinary ASCII files just like TEX sources, so it was easy to insert a listing of one of these files. The file that contains the METAFONT logo font (`logo10.mf`) suits our purpose quite well, since it is a rather simple font that probably everybody has available:

```
font_size:= 10pt#  ;
ht#        := 6pt#   ;
xgap#      := 0.6pt# ;
u#         := 4/9pt# ;
s#         := 0      ;
o#         := 1/9pt# ;
px#        := 2/3pt# ;
input logo
bye
```

What do we see here? First there are a bunch of assignments made (the lines that contain `:=`), then there is an `input` (this command functions the same way as TEXs `\input`, so it will start reading the file `logo.mf` next), and finally the last command is `bye`.

The file `logo.mf` contains the actual commands to create the characters. It helps to run METAFONT now to see what is going on. Just type the following to a system command prompt:

```
mf logo10
```

Probably, you didn't have to worry about where the `logo10.mf` file is on your harddisk, since most METAFONT implementations can do recursive directory searches just like TEX.

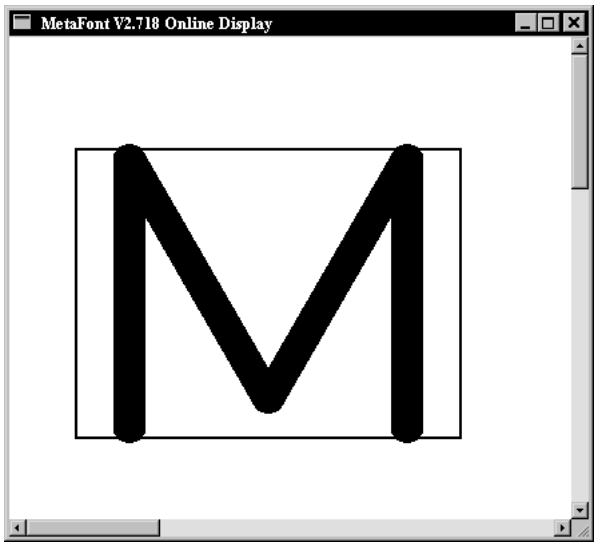


Figure 1 Metafont output window for modeless files

You should have seen a window popping up that shows characters as they are being created (like the one in fig-

ure 1, but it might look a little different on your system). Also, there should have been some terminal output, like this:

```
This is METAFONT, Version 2.718 (Web2c 7.2beta7)
(logo10.mf (logo.mf [77] [69] [84] [65] [70] [80]
[83] [79] [78]) )
Output written on logo10.2602gf (9 characters, 98
80 bytes).
Transcript written on logo10.log.
```

The numbers you see are the positions of the characters in the font. for the METAFONT logo font, these are the positions of the used characters in the ASCII table: M, E, T, A, F, P, S, O, N. As you can see, they can appear in any order within the source files.

The file METAFONT has written is not precisely the same as the output on your screen, instead it looks like figure 2. Not really usable when it comes to typesetting text, but it contains some pretty valuable information nevertheless.

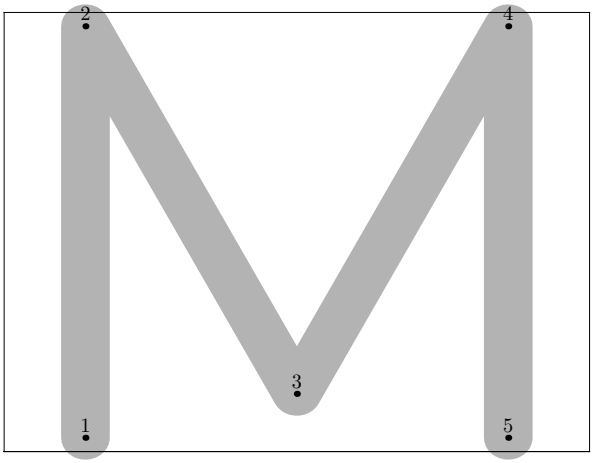


Figure 2 Metafont output file for modeless files

Have a look at the `logo.mf` file if you are interested in the nitty–gritty details. I will use only a small portion of that file to make some remarks about METAFONT. First, here is the edited program text I will use to explain things. (This is *no longer* valid METAFONT input, so don't start keying it in):

```
mode_setup;
define_pixels(s,u);

ygap#:=(ht#/13.5u#)*xgap#;
define_whole_pixels(xgap);
define_whole_vertical_pixels(ygap);
py#:=.9px#;
define_blacker_pixels(px,py);
pickup pencircle xscaled px yscaled py;
logo_pen:=savepen;

leftstemloc#:=2.5u#+s#;
define_good_x_pixels(leftstemloc);

beginlogochar("M",18);
x1 = x2 = leftstemloc;
x4 = x5 = w - x1;
x3      = w - x3;
y1      = y5;
y2      = y4;
bot y1  = 0;
top y2  = h;
y3      = ygap;
draw z1--z2--z3--z4--z5;
labels(1,2,3,4,5);
endchar;
```

Let's start at the line that begins with `beginlogochar`, because this is where the real work is done. This portion of the source defines the letter 'M' in the font. What we see here is that characters are specified by first setting up a bunch of equations (the lines that have equal signs in them), followed by a `draw` command that connects those points, actually drawing the character.

We won't go deeply into METAFONT syntax, but it is vital to understand the following: a point is defined as a pair of x and y coordinates. In METAFONT syntax, points are sequentially numbered per character, starting from 1. `z1` is the notation for point 1. Notations like `x1` and `y2` specify the x-component of point 1 and the y-component of point 2, respectively.

`beginlogochar` says that the 'M' is precisely $18u$ (units) wide, and `logo10.mf` has set up one u to be 4/9pt, so the actual character is $4/9 \times 18\text{pt} = 8\text{pt}$ wide.

TEX and METAFONT have the same author, and it shows: METAFONT can do macros just as easily as TEX can. Macros can have arguments, define other macros and assign values to things, just like in TEX. `beginlogochar` is in fact one of those macros, and it assigns some pretty important values when it gets expanded by METAFONT. For one, it defines w to be the width we calculated above, and it defines h to be the height of the character (calculated from `ht#` in `logo10.mf`, which equals 6pt).

From the values of u and s, it now follows that `leftstemloc` equals $(2.5 \times 4/9) + 0 = 10/9\text{pt}$. The last value we have left is `ygap = (ht/13.5u) * xgap`. `ht` and `xgap` have been given in the 'driver' file `logo10.mf`, and after some small calculations `ygap` becomes $(6\text{pt} \div (13.5 \times 4/9\text{pt} \times 0.6\text{pt})) = 0.6\text{pt}$.

It should now be easy to come to the conclusion that the equations fix the precise x, y locations of the five points that denote the character 'M', albeit in a slighly indirect manner. If we fill in the values we derived above, we get the following:

```
x1 = x2 = 10/9pt;
x4 = x5 = 8pt - x1;
x3      = 8pt - x3;
y1      = y5;
y2      = y4;
bot y1  = 0pt;
top y2  = 6pt;
y3      = 0.6pt;
```

After METAFONT has calculated these equalities for us, and with some minor reshuffling of the input, we get the following end-result:

```
x1 = 10/9pt  ; bot y1 = 0pt  ;
x2 = 10/9pt  ; top y2 = 6pt  ;
x3 = 4pt     ;     y3 = 0.6pt;
x4 = 62/9pt  ; top y4 = 6pt  ;
x5 = 62/9pt  ; bot y5 = 0pt  ;
```

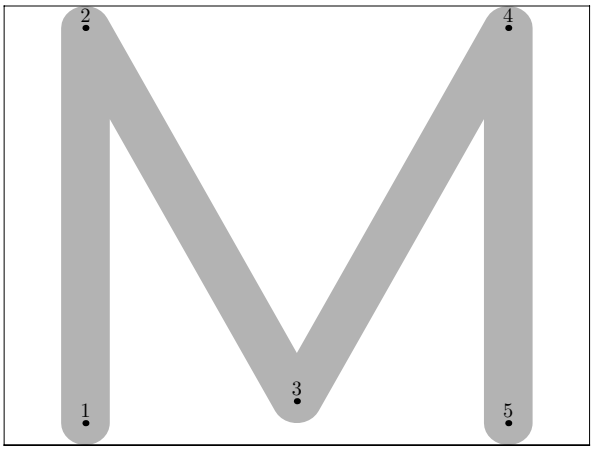


Figure 3 Metafont output file for modeless files, hand–calculated version

We could have typed this in right away, and METAFONT would have been just as happy. The end result would have been the same, as can be seen in figure 3.[1] But why do the calculus yourself if the machine can do it for you?

METAFONTs ability to do the needed calculations all by itself is one of its most important strong points. Combined with macros and separate input files, it becomes possible to use various fonts with the same shared sources. In such a 'font', the only file that is different between various versions of the font is the 'driver' file, that assigns different values to the same parameters. METAFONTs calculations will have different results, so the some of points will end up in slightly different locations. The resulting font will be similar in style but may still differ in lots of ways.

Creating a full font

Usually, fonts are not just a bunch of characters. There also is some other metric information included in almost every font. The final section of our example file (at the end, after all the characters have been defined) contains the following lines:

```
ligtable "T": "A" kern -.5u#;
ligtable "F": "O" kern -u#;
ligtable "P": "O" kern u#;

font_quad:=18u#+2s#;
font_normal_space:=6u#+2s#;
font_normal_stretch:=3u#;
font_normal_shrink:=2u#;
font_identifier:="MFLOGO" ;
font_coding_scheme:="AEFMNOPST only";
```

The first three lines belong to the 'ligature table'. Usually it will contain both real ligatures and the kerning information for the font, but because this is a very simple font, there are only three really simple kerning pairs.

The next lines define TEXs `\fontdimen` values: how wide a space will be and how much it can stretch and shrink, and some other information that will appear in the created font but is generally not used by programs.

Dealing with device dependence

Now let's have a look at the device dependant calculations that METAFONT does. Here is the relevant portion of the example again:

```
mode_setup;
define_pixels(s,u);

ygap#:=(ht#/13.5u#)*xgap#;
define_whole_pixels(xgap);
define_whole_vertical_pixels(ygap);
py#:=.9px#;
define_blacker_pixels(px,py);
```

There are, in fact, two kinds of device dependence that need to be dealt with. The `mode_setup` line takes care of the first kind of device dependance: the effects that the actual hardware of the printing engine can have on the printed

1 Actually, figure 3 is not completely identical to figure 2, because in my example I cheated with the calculations a bit to keep the explanation simple.

font.

The most obvious difference between any two printing devices is of course the resolution, but there are other problems as well. Since we prefer our output to look as close to our intended font as possible, usually a certain amount of correction is needed based on (i.e.) whether the device is going to be an inkjet printer or a lasertypesetter.

`mode_setup` cannot do this all by itself, and this is why you usually have to specify somewhere what printer you are using. Programs like `dvips` will call METAFONT with a command like:

```
mf \mode=ljfour; mag=1; input logo10
```

If we forget about that first backslash, we can see that there are two assignments and one `input` command on this line. The second assignment differs from 1 when a font is called within TEX using a command like

```
\font\logohuge = logo10 at 20pt
```

In that case, the assignment would be `mag=2`. The other assignment is far more interesting. METAFONT usually starts with a 'format' file similar to the `fmt` files TEX uses, and somewhere in the sources for those format files there are some definitions like this:

```
mode_def cx =
  mode_param (pixels_per_inch, 300);
  mode_param (blacker, 0);
  mode_param (fillin, .2);
  mode_param (o_correction, .6);
  mode_common_setup_;
enddef;
```

Figure 4 An example of two different imaging models.

All parameters besides `pixels_per_inch` are a little too technical to explain in detail in a short article like this one, but figure 4 tries to explain that these values really do depend on the printing engine. The drawing on the left shows a more or less standard inkjet, that shoots dots of (black) ink on the paper. The right drawing shows a (hypothetical) printing device with a radically different approach. This machine pours light on a photographic film through a raster, creating a negative image. There are still round dots, but they are inverted! It is easy to imagine that this radically different technique can have quite an impact on the resulting image.

One effect that is very easy to see from the (admittedly very badly drawn) figure is that the inside corners in the right drawing are a lot blacker than in the left one. This sort of thing happens all the time with in real life printing, but it often goes unnoticed because people tend to have only one printer.

The second device dependancy is not really related to printers at all, but is caused simply by the fact that METAFONT outputs a pixel bitmap. Although METAFONT does it calculations with a very high accuracy, this does not help at all if there are simply not enough pixels to display the character. The commands that look like `define_xxx_pixels` take care of this kind of dependancy, whose effects can be seen in figure 5.

Figure 5 The character on the right has been created with all the `define_xxx_pixels` commands removed from the source.

The sub-optimal distribution of pixels in this example is caused by the underlying pixel grid that can not be changed.

What are Type 1 fonts?

How PostScript fonts are created

PostScript Type 1 fonts are quite different from METAFONT fonts. Usually, Type 1 fonts are created in a wysiwyg environment with a drawing program that is only suited for the creation of fonts. Figure 6 shows the program I usually use.

The graphical user interface nicely shields off the designer from what is happening behind the scenes, so we need to look into the generated files themselves if we want to get more information. On Windows and Unix systems, the actual fonts are saved in a binary file with the extension `pfb` (short for PostScript Font Binary), and the metric information in an ascii file with extension `afm` (short for Adobe Font Metrics).

2 This section is loosely borrowed from Erik-Jan Vens' article "Incorporating PostScript fonts in TEX", EuroTEX proceedings

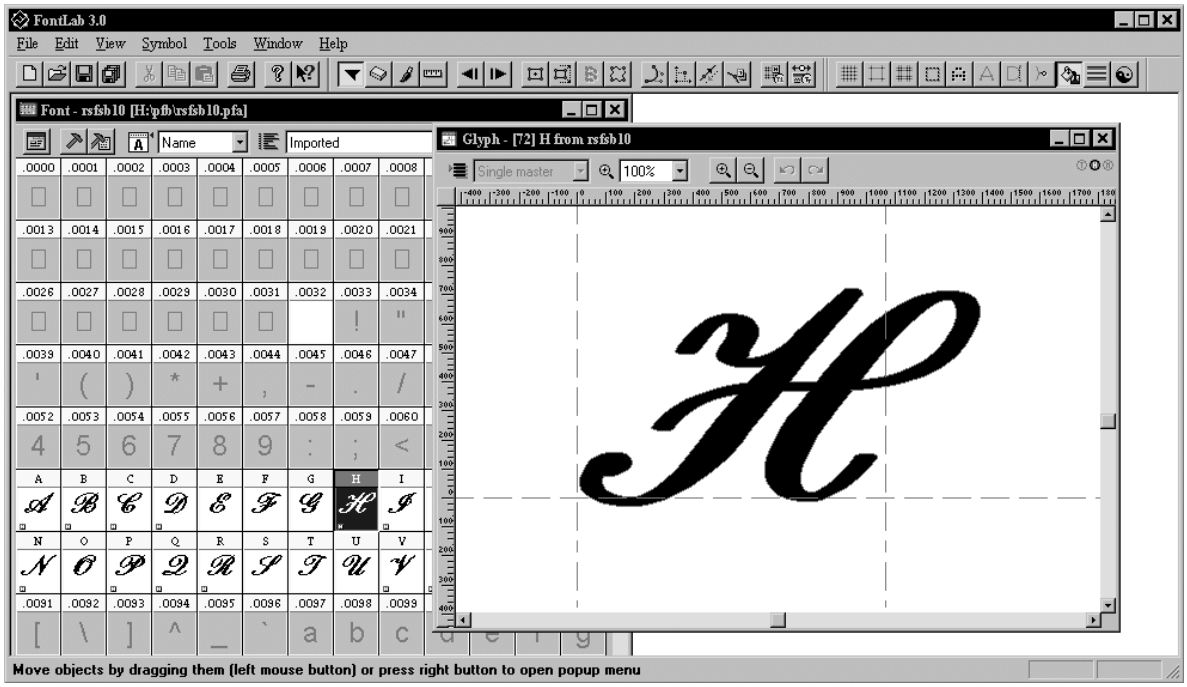


Figure 6 An interactive font editor: Fontlab version 3

What a Type 1 font looks like[2]

The binary representation of a Type 1 font is just a compressed version of the non-compressed ascii format, with extension `pfa`. So we need a program that will do the decompression for us. One of the programs that can do this is `T1ascii` from the T1Utils package. But running this programs leaves us with a hexadecimal encrypted file. In the early days, the encryption key was a trade secret of Adobe Incorporated. This key is now freely available, but the file format still reflects the past. Yet another program from the T1Utils can convert this form to real human–readable PostScript: `T1disasm`. Now we can look at the generated PostScript file to see how the 'M' is defined in Type 1 format:

```
/M {
   78 800 hsbw
   611 -20 hstem
   -11 21 hstem
   0 66 vstem
   578 66 vstem
   581 595 rmoveto
   -259 -450 rlineto
   -259 450 rlineto
   -6 9 -12 7 -12 0 rrcurveto
   -16 -17 -12 -17 hvcurveto
   -563 vlineto
   -17 15 -13 18 vhcurveto
   19 14 13 17 hvcurveto
   439 vlineto
   76 -131 75 -131 75 -131 rrcurveto
   5 -10 12 -6 13 0 rrcurveto
   14 0 8 8 8 8 rrcurveto
   75 131 75 131 76 131 rrcurveto
   -439 vlineto
   -17 14 -13 19 vhcurveto
   18 15 13 17 hvcurveto
   562 vlineto
   17 -17 13 -16 vhcurveto
   -12 0 -12 -5 -6 -11 rrcurveto
   closepath
   endchar
} ND
```

The code looks enough like normal PostScript to recognize it at first glance, but the commands themselves are not the same you would use in everyday graphics. The PostScript language uses reverse Polish notation for it's commands, so you should read backwards, starting at the end of the line. `581 595 rmoveto` means 'move to the point with coordinates (581, 595).

1992, pp. 173–181.

All values are given in a coordinate system that maps 1000 units to one `em`. The nullpoint lies at the lower left corner. When one uses a PostScript font in a PostScript language program, the coordinate system is initially scaled in a way such that 1000 units equal precisely `1 bp`. The values used to describe points and intermediate values can be negative, but never partial. This need for discrete values can be a major problem when converting METAFONT fonts, as we will see later on.

Now let's have a short look at the used commands. The command `hsbw` sets up the width information for this character (the first number is the left sidebearing distance, the second number the advance width). The commands that end in `stem` are used by the hinting system. The whole collection of commands that look like `xlineto` and `xxcurveto` are shortcuts for the ordinary PostScript commands `lineto` and `curveto`: these draw the actual outline. All of these drawing commands are always relative to the 'current point'. The last couple of commands end the character: `closepath` to close the defined path (like METAFONT's `cycle`) and `endchar` to do the actual drawing. `ND` functions as `def`: it defines the command '`M`' (from the first line) to mean 'do everything between the braces' (remember this is reverse Polish notation).

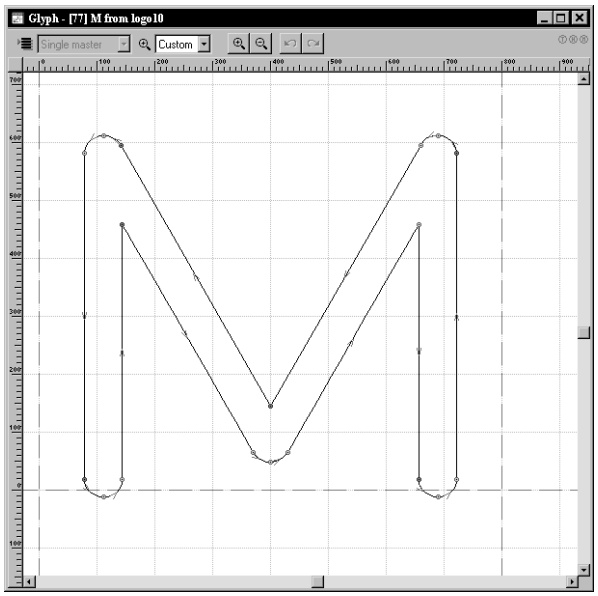


Figure 7 How the character's path is drawn.

At first sight it is a little surprising to see that the PostScript representation is rather a lot longer than the METAFONT version. This is caused by another limitation of Type 1 format: every character *has* to define an outlined path that is filled by `endchar`. Thanks to this limitation, we cannot use four stroked lines to draw the 'M' the way we did in METAFONT, but instead are forced to trace the borders of filled shape.

Dealing with device–dependancies

Adobe's Type 1 format does not supply a means of dealing with device differences directly, like METAFONT's `define_good_pixels`. But of course there has to be some means of making sure that a font looks reasonable on low-resolution devices, and this is handled by a system called 'hints'. The responsible commands are separated into two different levels: there are 'font-level' hints and 'character–level' hints. Font-level hints take care of three things:

1. Alignment zones
2. Standard stem widths
3. Extra information to control the hinting

The relevant portion of the font-file looks like this:

```
/BlueValues [ -12 0 600 611 ] ND
/BlueScale 0.04379 def
/BlueShift 7 def
/BlueFuzz 1 def
/MinFeature { 16 16 } ND
/StdHW [ 60 ] ND
/StdVW [ 66 ] ND
/ForceBold false def
```

Alignment zones First off, alignment zones are defined by the array called `/BlueValues`. The values in the array define vertical zones by specifying two y coordinates for each zone. In this case, there are only the two areas between $[-12, 0]$ and $[600, 611]$, but there may be more entries.

The first entry in the array defines an area in which the y-coordinates of points (that lie within this area) are changed into the highest (second) number. For the following entry, the y-coordinate is changed into the lowest (first) number. Together, these two areas allow characters like the 'O' to be rendered at low resolution without sticking out unacceptably below the baseline if compared to characters like the 'H' (see figure 8).

Standard stem widths Quite often, a vertical or horizontal line in a font will be just a little bit to large for one device pixel but not large enough for two pixels. Depending on the underlying pixel grid, the line may consequently be rendered as either one or two pixels.

In these problematic characters, we want to make sure at least that verticals and horizontals that are intended to have the same width throughout the font use the same amount of pixels. This is done by pre-defining the widths that are supposed to be identical. The commands that pass this in-

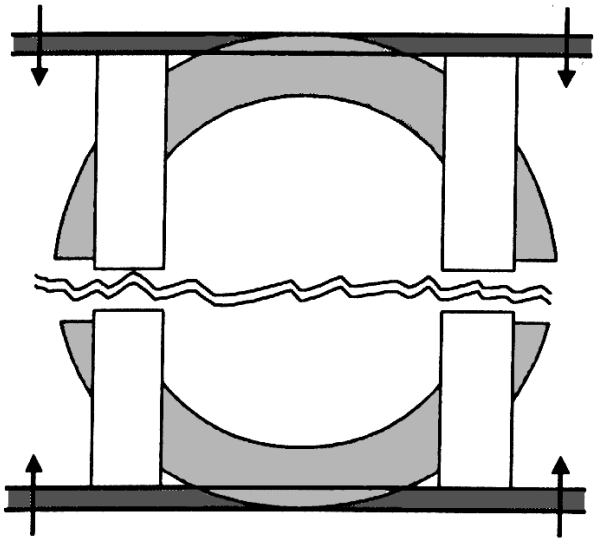


Figure 8 An example of how overshoot-suppression works in PostScript: the top and bottom of the 'O' are adjusted so that the character becomes just as high as the 'H'. (figure borrowed from the Fontlab Manual)

formation to the renderer are `StdVW` and `StdHW`. The effect that a correct setting of these values has on the rendering of the font can be seen in figure 9.[3]

Extra information to control the hinting There are some extra commands in the example that we haven't covered yet: the three `Bluexxxx` commands define (amongst other things) the pointsize below which overshoot suppression is turned on, and a fuzzy correction on the values of the alignment zones. `ForceBold` is used with bold fonts to make sure that they will stay at least two pixels wide at low resolutions (otherwise they would look identical to the non-bold version at small sizes).

The character-level hints are handled by the commands from the top of the listing given previously:

```
611 -20 hstem
-11 21 hstem
0 66 vstem
578 66 vstem
```

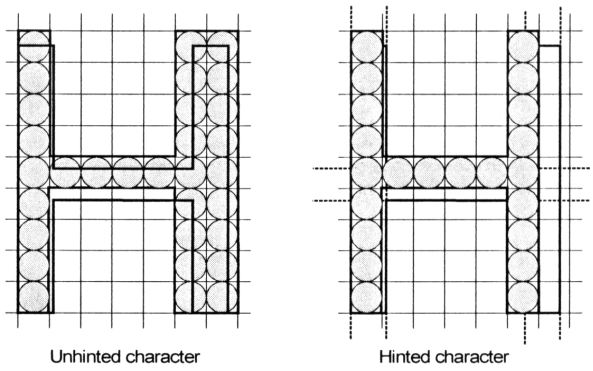


Figure 9 The desired result. (this figure is also borrowed from the Fontlab Manual)

These define horizontal and vertical stem zones. The first number says at which coordinate to start, the second number the width to use from there. In this case (remember this is an 'M') there are two vertical stems, and two 'ghost' horizontal stems. Figure 10 shows the graphical representation of this character in the font editor.

The 'ghost' stems are inserted because without them the overshoot suppression wouldn't work.

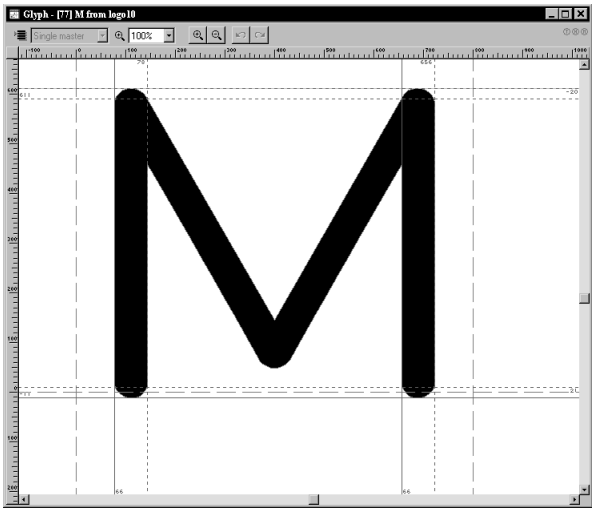


Figure 10 The character 'M' from the METAFONT logo font, with PostScript hints added

3 Like all hinting information, these values are ignored if the stem widths are larger than three device pixels (approximating 1200 dpi for the average font). As a result, output at 1200 dpi on a new device sometimes looks inferior to the 600 dpi version for the trained eye.

Why we want to convert to Type 1

Now that we have looked briefly into both formats, it is obvious that conversion from METAFONT input syntax to PostScript definitions is not going to be easy. METAFONT is apparently a lot smarter than the Type 1 interpreter, much better suited to handle device dependancies, and more accurate.

So, why bother at all? For practical reasons, of course. The most important incentive is the on-screen display of generated PDF files. Adobe's Reader is very bad at displaying bitmapped fonts, so files with only Type 1 fonts look a lot better. As it is, there are quite some METAFONT fonts that don't (yet) have a Type 1 counterpart, so necesarily lots of TEX-generated PDF files use bitmaps.

There also is another interesting motive: designing high quality fonts in METAFONT syntax is a lot easier than creating Type 1 fonts of the same quality in an interactive editor (not to mention the fact that interactive programs usually crash at every second mouse click).

Tasks to be handled by the conversion process

Various things need to be taken care of by the conversion, but the three major parts are:

1. Resolving the equations in the METAFONT sources.
2. Converting stroked paths into outlined paths.
3. Insertion of Type 1 style hinting information.

Resolving the equations in the METAFONT sources. The first item is easy to do with an already existing program: METAPOST. METAPOST is a program by John Hobby (co-author of METAFONT) that accepts METAPOST input syntax and outputs an Encapsulated PostScript picture. For example, running METAPOST on the logo fonts (using precisely the same syntax as for METAFONT) gives the following output:

```
%!PS
%%BoundingBox: 0 -1 8 7
%%Creator: MetaPost
%%CreationDate: 1998.05.10:1535
%%Pages: 1
%%EndProlog
%%Page: 1 1
0.66418 0 dtransform exch truncate
     exch idtransform pop setlinewidth
[] 0 setdash
1 setlinecap
1 setlinejoin
10 setmiterlimit
gsave
newpath
1.10696 0.18819 moveto
1.10696 5.78938 lineto
3.98503 0.78595 lineto
6.8631 5.78938 lineto
6.8631 0.18819 lineto
1 0.9 scale
stroke
grestore
showpage
%%EOF
```

The PostScript code contained in this file is not that hard. The first few lines are just comments. The two lines that end with `setlinewidth` do nothing except setting the line width for strokes. It looks complex, but the code is always the same, the only things in these two lines that ever change are the two numbers.

The next lines set up some values of the PostScript graphics state that do not always have a predefined value (this is just a security measure). These lines also never change. `newpath` is the first command that is interesting: starting from here the character is defined. Indeed, there is only one `moveto`, followed by four straight lines, and finally a `stroke`.

It would be a little bit easier if the calculated values were given in units of a thousand per *em*, and this can be done by inserting a different `mode_def`. Basically, we ask METAPOST to generate a normal font file, but at a magnification of 100.375. This gives us an end result in PostScript big points, and the generated character will now look like this (some comments and irrelevant lines stripped):

```
%!PS
%%BoundingBox: 77 -12 723 612
66.66722 0 dtransform exch truncate
 exch idtransform pop setlinewidth
gsave newpath 111.1115 18.88889 moveto
111.1115 581.11142 lineto
399.99867 78.88953 lineto
688.88585 581.11142 lineto
688.88585 18.88889 lineto
1 0.90001 scale stroke grestore
showpage
%%EOF
```

Good. This is starting to look like something we could use. Ideally, we would prefer a output in rounded numbers, but that is not possible.

Converting stroked paths into outlined paths. Mr. Kinch (author of TrueTEX) has written a program called `metafog` that converts METAPOST output as in the ex-

ample above into the format required by the Type 1 specifications. At the moment, the program is only available as an optional extra with TrueTEX, but correspondence with Mr. Kinch indicate that it is very likely that this program will soon be available separately.

Metafog reads the METAPOST EPS file, and converts this into another EPS file. It also displays some debugging information about the character on the terminal:

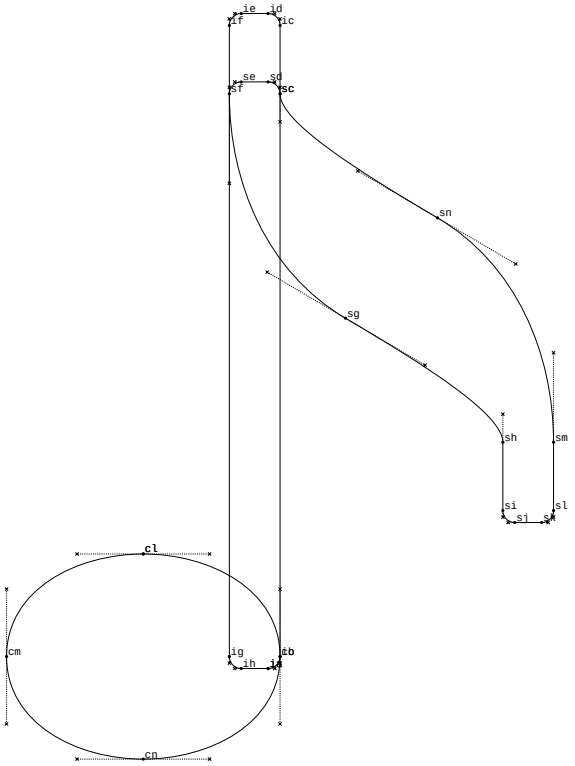


Figure 11 Metafog output file, page 1

```
interp: "scale" implies elliptical pen, 66.7 x 6\
                                                0.0
o: scaffolded TRUE
reduce: reducing shape 1 of 2
reduce: reducing shape 2 of 2
duplicate: scaffolded
try_point: 0.01 value scaffold
reduce: reducing shape 3 of 2
.....
reduce: reducing shape 5 of 4
Plotting page 1 (Initial Input Contour) ... done\
                                              plotting.
reduce: reducing shape 1 of 1
reduce: reducing shape 2 of 1
Plotting page 2 (Final Result Contour) ... done \
                                              plotting.
Total knots used: 598 (a--wz), ~ 29% indexable c\
                                              apacity
```

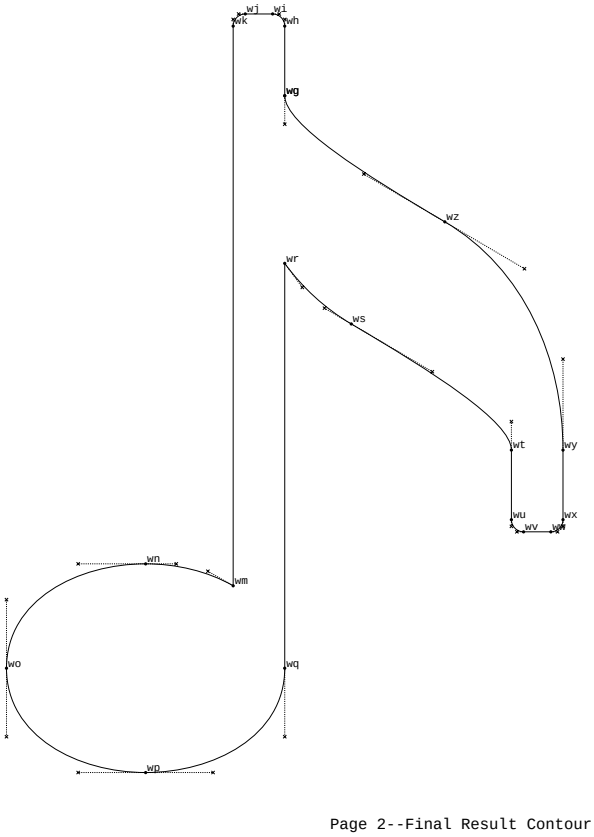


Figure 12 Metafog output file, page 2

The created files are pretty large, too large to include literally. This is because the file serves two purposes: it is the input format for another program (`makefont`, also by Kinch) *and* it also shows the work that has been done by metafog. The first page shows the result, the second page the initial input as metafog saw it. (the output of one of these files is shown in figure 11 and figure 12).

All we have to do is run metafog on all characters in the font. If everything went correctly, the next step in the process is running the `makefont` program. But this is not always the case. Metafog has its flaws, and it is especially bad at handling complex characters. One of those trickier characters is given in figure 13.

In order to handle cases like this gracefully, metafog has a special startup option that gives a half-way result: it cuts

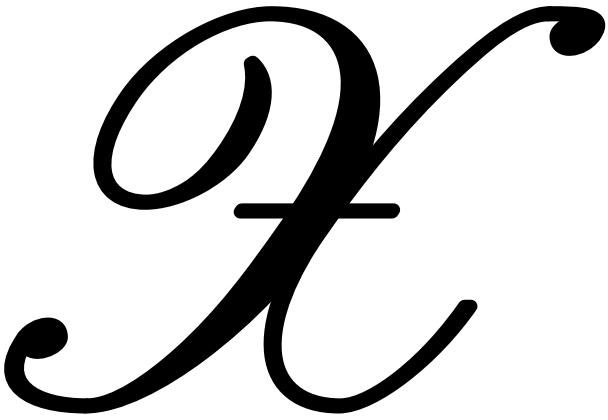


Figure 13 The character 'X' from Ralph Smith's Formal Script

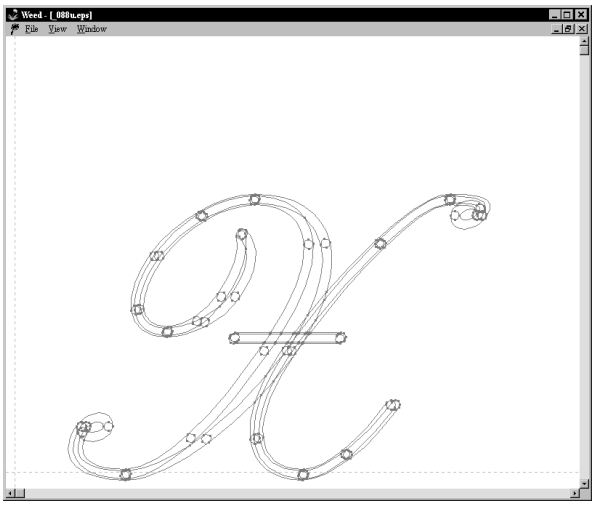


Figure 14 Screendump of the weeder's window

the supplied shape in pieces, but it does not try to remove parts that are not needed. There is yet another program in the metafog suite that helps for the problematic characters.

This program is called the 'weeder' (figure 14). It is an interactive program that reads the half-way result metafog created. The human operator now has to select the partial lines that are supposed to belong to the shape, and the weeder will write a finished file for use by makefont (just like metafog itself would have done if things had gone right the first time).

For some fonts, one has to do almost every character 'by hand', for other fonts none at all. The point of view the machine has on what precisely denotes a complicated character can be rather unexpected: sometimes a character can look very simple to you but be almost impossible to process by metafog (usually characters that use `draw` and `fill` commands that intersect somewhere). And the other way around also happens: large portions of the `nash14` (arabic) font looked exceedingly complex to me, but were in fact handled by metafog without any problems.

Either way, eventually there will be EPS files available for all characters in the font. Makefont combines all of the separate files into one PostScript file, and the last step of the actual conversion process is running the T1Utils to get a binary representation that can be fed into a commercial font editor.

Insertion of Type 1 style hinting information.

The `pfb` file created at the end of step two still has a couple of major flaws that need to be fixed. First and foremost among these: there are absolutely *no* Type 1 hints included. There *were* hints in the original METAFONT sources, but these are ignored by METAPOST, and subsequent portions of the conversion do not have access to them. This is the major reason for the need of a commercial font editor. Type 1 hinting is too complicated a process to rely on any non–interactive program to make the right choices.

Another thing that must be checked, especially for symbolic fonts, is the turning direction of the subpaths. In PostScript, whether a path will be black or white depends on how the path turns: clockwise or counterclockwise. Metafog sometimes gets confused, and outputs a character in which two concentric circles both turn leftward (like in an 'O' or the diameter symbol from the Waldi Symbol font). In those cases, the character will be completely filled, which is of course wrong.

The absolutely final step (so far I've needed to do this for every commercial font program I could find) is disassembling the `pfb` file, running a perl script to fix some incompatibilies/bugs in the used font editor and to insert a couple of workarounds for bugs in software that uses the font, and reassembling.

What I have done already and future plans.

So far, I have converted four METAFONT fonts that I needed myself. The files that are now avialable from CTAN are:

```
logo8       wasy5      stmary5      rsfs5
logo9       wasy7      stmary7      rsfs7
logo10      wasy10     stmary10     rsfs10
logosl10    wasyb10
logobf10
```

All of these files reside in a subdirectories of the METAFONT sources, named `ps-type1/hoekwater`. The logo font is located under `cm/utilityfonts` due to a weird directory structure on CTAN. Note that I have not always

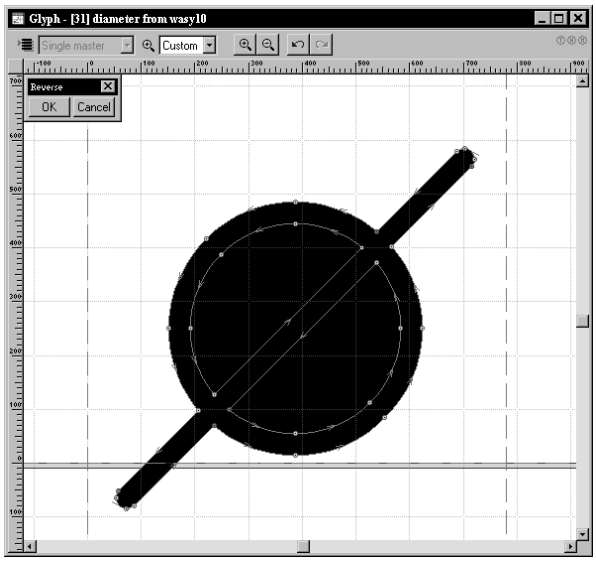


Figure 15 Paths that turn the wrong way

converted all of the METAFONT source files (for instance, from the Waldi2 set `wasy6`, `wasy8` and `wasy9` are missing), so LaTeX users probably need a changed Font Definition File.

Each directory now also contains a readme:

Hi there,

This directory contains a Type 1 version of (some of) the Metafont sources from the directory above. The pfb files are intended to be used with the original TFM files, *don't* run finst or afmtotfm on the afm files!

The afm and pfm files are provided just in case you want to install the files on your windowing system. I'm sorry for the Macintosh users amongst you, but I don't know how to create the MacType1 files.

The fonts should give exactly the same output results as the .mf originals, up to the point of sillyness. All bugs in the metafont sources have been kept (and there might be new ones).

Files are herewith donated to the public domain, and provided as is. Note that I feel that the copyright from the metafont sources still applies, my only statement here is that I do not impose extra restrictions.

Conversion process:

.mf → metapost (c) Hobby → .eps
.eps → metafog (c) Kinch → .pfb
.pfb → hinted & touched up with FontLab v3.0c

More fonts should follow in the next few months. I can be reached for propositions/bugs at:
`taco.hoekwater@wkap.nl`

Greetings,

Taco Hoekwater

Please take a note of the fact that I *only* want to give support for problems that are *intrinsically* related to the `.pfb` files themselves. I don't have enough spare time to help people with problems related to the integration of the fonts into their TeX distributions. If somebody wants to volunteer for this job, please let me know and I will add you to the readme.

I plan to add quite a couple of fonts in the near future (some of which may have been uploaded already by the time you read this). The announcement of the availability of the files that are now on CTAN almost immediately resulted in a doubling of the length of my wish list.

The following fonts are *TODO* and will definately be done before the summer:

- The rest of the METAFONT logo fonts belonging to the `mflogo` package (on a request by Ulrik Vieth).
- At least the most important fonts that are needed by the `wsuipa` package: `tipaxx` and `xipaxx` (I already started on those, but the MAPS takes precedence, sorry Sebastian!).
- The Nash font that is used by ArabTeX (but Klaus Lagally says that he needs to fix and update the METAFONT sources first).
- The Blackboard Bold font (actually, everything that is needed for the new math font encoding will be finished before the end of the year).
- At least one each of the Greek, Cyrillic and Hebrew text font families (could someone please point me to the 'best' font of those that are available?).
- The `manfnt` (requested by Phil Taylor).

Every font (presuming 256 characters) takes about one day to complete. This does not sound like too much time, but unfortunately I also have other work to do :-(

I am still open for requests, but you may have to wait a couple of months.

For further reading

On METAFONTs language:
Donald E. Knuth, “The METAFONT Book”. Addison–Wesley Publishing Company, June 1986, *361 pages*.

On using METAFONT to design real life fonts:
Donald E. Knuth, “Computer Modern Typefaces (Computers and Typesetting, volume E)”. Addison–Wesley Publishing Company, June 1986, *588 pages*.

On the PostScript Language:
Adobe Systems Inc, “The PostScript Language Reference Manual”. Addison–Wesley Publishing Company, December 1990, *764 pages*.

On METAPOST:
John Hobby, “A User’s manual for METAPOST”, AT&T Bell Laboratories Computing Science Technical Report 162, 1992. Comes as part of the METAPOST distribution.

On Type 1 fonts:
Adobe Systems Inc, “Adobe Type 1 Font Format”. Addison–Wesley Publishing Company, June 1995.

On the Metafog program:
Richard J. Kinch, “Converting METAFONT Shapes to Outlines”. Paper presented at the 1995 TUG Conference in St Petersburg, Florida, USA. Appeared in print in *Tugboat* 16.3

Bijlage 27
Vergelijking van sgml en xml

Simon Pepping
Elsevier Science
E-mail: s.pepping@elsevier.nl

XML (eXtensible Markup Language) is één van de buzz words van het afgelopen jaar op het Internet. Het wordt begroet met kreten als 'Een tweede kans voor SGML', 'Web auteurs bevrijd uit het keurslijf van HTML', 'de nieuwe Web standaard voor informatieuitwisseling'. Zowel in SGML kringen als bij de softwareindustrie roept het enthousiaste reakties en grootse verwachtingen op.

XML komt voort uit twee bewegingen. 1. In de SGML wereld beseft men dat de SGML wel mooi maar moeilijk implementeerbaar is. Daardoor zijn er maar weinig toepassingen van ontstaan. 2. In de software industrie beseft men dat HTML te beperkt is om allerlei soorten informatie over het Internet uit te wisselen. XML is de ontmoeting van die twee: Het is een vorm van SGML ontdaan van zijn moeilijkste kanten, zodat het eenvoudiger wordt om er toepassingen voor te maken. Doordat het nog steeds SGML is, is het flexibel en krachtig genoeg om velerlei typen informatie te kunnen beschrijven.

Nu al zijn er verscheidene parsers voor XML beschikbaar, en hun grootte bedraagt maar rond een vijfde tot een tiende van de James Clark's `nsgmls` parser voor volledig SGML (op mijn Linux machine 1.2MB).

De meeste informatie over XML komt uit de SGML hoek en is vaak technisch van aard. Bovendien benadert het XML vanuit het SGML gezichtspunt en benadrukt vooral de verschillen met SGML. Daar heb je niet zoveel aan als je geen SGML kent. Een informatief dokument over XML voor belangstellenden die minder van SGML af weten is Peter Flynn's 'Frequently Asked Questions about the Extensible Markup Language' (`http://www.ucc.ie/cml/faq.sgml` en `http://www.ucc.ie/xml`). Het beantwoordt zelfs de vraag: 'What is SGML?'

Op de volgende bladzijden drukken we een notitie af van één van de voormannen van SGML, James Clark. Hierin geeft hij een precieze opsomming van de verschillen tussen SGML en XML. In de volgende alinea's zal ik enkele punten naar voren halen.

Het valt me op dat een in het oog springend verschil tussen SGML en XML niet genoemd wordt: XML documenten hoeven geen DTD te hebben. Zulke documenten kunnen weliswaar niet door een parser op hun geldigheid gekontroleerd worden (ze zijn niet 'valid'), maar als ze aan redelijke eisen van markup voldoen, zoals volledig getagged, geen slechte nesting, zijn ze wel 'well-formed'.

In XML zijn veel moeilijk implementeerbare kenmerken van SGML geschrapt in het belang van een grotere praktische haalbaarheid. Zulke vereenvoudigingen zijn o.a.:

- XML heeft een vaste SGML deklaratie; dit sluit al heel veel bijzondere mogelijkheden en moeilijkheden uit.
- Alle openings- en sluittags moeten aanwezig zijn (OMITTAG NO). Ongesloten openings- en sluittags zijn niet toegestaan. Attribuutwaarden moeten altijd tussen aanhalingstekens gegeven worden ('entered as literals'). Dit maakt de markup regelmatiger en dus gemakkelijker te parsen.
- < en & mogen niet als tekstkarakter voorkomen. In SGML mag dit onder bepaalde voorwaarden wel. Weer zo'n beperking die parsen gemakkelijker maakt.
- Een aantal verfijningen in SGML zijn overboord gezet ten gunste van praktische haalbaarheid, b.v. `NUTOKEN(S)`, `NUMBER(S)` en `NAME(S)` zijn niet toegestaan als gedeklareerde waarden voor attributen.
- Zelfs vaker gebruikte en dus zeker nuttig gebleken mogelijkheden van SGML zijn verwijderd ten gunste van de haalbaarheid, b.v. `INCLUDE/IGNORE` marked sections zijn niet in een document toegestaan.

XML heeft ook enkele zaken aan SGML toegevoegd, zoals:

- Er mogen meerdere attribuut deklaraties voor één element voorkomen.
- Er is geen onderscheid meer tussen als `EMPTY` gedeklareerde elementen en elementen zonder inhoud. Beide mogen gekodeerd worden als `<fig file="fig1.eps"/>`.

Deze zaken hebben een aanpassing van de SGML specifikatie vereist, de 'Web SGML Adaptations Annex' die in het document genoemd wordt.

Toepassingen van XML staan nog in de kinderschoenen, maar er is veel aktiviteit aan het front. Er zit een XML parser in MS Internet Explorer 4 en in Netscape Navigator 5 (James Clark's `expat` parser). Er zijn verscheidene standalone XML parsers, vooral in Java geschreven. Er zijn XML modules in Perl. En er komt dagelijks wat bij. Voor informatie zie de SGML/XML web page op `http://www.sil.org` en de nieuwsgroep `comp.text.sgml`.

Bijlage 28
Comparison of SGML and XML

James Clark
E-mail: jjc@jclark.com

abstract
This document provides a detailed comparison of SGML (ISO 8879) and XML.

status of this document
This document is a NOTE made available by the W3 Consortium for discussion only. This indicates no endorsement of its content, nor that the Consortium has, is, or will be allocating any resources to the issues addressed by the NOTE. Errors or omissions in this document should be reported to the author.

1 Differences Between XML and SGML

XML allows only documents that use the SGML declaration in this note. This declares all the following SGML features as NO:

- DATATAG
- OMITTAG
- RANK
- LINK (SIMPLE, IMPLICIT and EXPLICIT)
- CONCUR
- SUBDOC
- FORMAL

Note that it differs from the reference concrete syntax in a number of ways:

- It also declares no short reference delimiters; it follows that SHORTREF and USEMAP declarations cannot occur in XML
- The PIC (processing instruction close) delimiter is ?>
- Quantities and capacities are effectively unlimited
- Names are case sensitive (NAMECASE GENERAL is NO)
- Underscore and colon are allowed in names
- Names can use Unicode characters and are not restricted to ASCII

The following constructs which are permitted in SGML when SHORTTAG is YES are not allowed in XML:

- Unclosed start-tags
- Unclosed end-tags
- Empty start-tags
- Empty end-tags
- Attribute values in attribute specifications entered directly rather than as literals
- Attribute specifications that omit the attribute name

NET delimiters can be used only to close an empty element. In SGML without the Web SGML Adaptations Annex, the NET delimiter is declared as />. With this approach, XML is not allowing null end-tags and is allowing net-enabling start-tags only for elements with no end-tag. In SGML with the Web SGML Adaptations Annex, there is a separate NESTC (net-enabling start tag close) delimiter. This allows the XML <e/> syntax to be handled as a combination of a net-enabling start-tag <e/ and a null end-tag >. With this approach, XML is allowing a net-enabling start-tag only when immediately followed by a null end-tag.

XML imposes the following restrictions not in SGML:

- Entity references
  - Entity references must be closed with a REFC delimiter
  - References to external data entities in content are not allowed
  - General entity references in content are required to be synchronous
  - External entity references in attribute values are not allowed
  - Parameter entity references are allowed in the internal subset only within a declaration separator (that is, at a point where a markup declaration could occur)
- Character references
  - Character references must be closed with a REFC delimiter
  - Named character references are not allowed
  - Numeric character references to non-SGML characters are not allowed
- Entity declarations
  - A #DEFAULT entity cannot be declared
  - External SDATA entities are not allowed

World Wide Web Consortium Note 15-December-1997, NOTE-sgml-xml-971215, available from http://www.w3.org/TR/NOTE-sgml-xml-971215.
Section 3, 'SGML declaration for XML', is not included here; see the Web document.

  - External CDATA entities are not allowed
  - Internal SDATA entities are not allowed
  - Internal CDATA entities are not allowed
  - PI entities are not allowed
  - Bracketed text entities are not allowed
  - External identifiers must include a system identifier
  - Attributes cannot be specified for an entity
  - The replacement text of general text entities and external parameter entities is required to be well-formed
  - An ampersand in a parameter literal must be followed by a syntactically valid entity reference or numeric character reference
- Attribute definition list declarations
  - Associated element type in attribute definition list declarations cannot be a name group
  - Attributes cannot be declared for a notation
  - CURRENT attributes are not allowed
  - Content reference attributes are not allowed
  - NUTOKEN(S) declared values are not allowed
  - NUMBER(S) declared values are not allowed
  - NAME(S) declared values are not allowed
  - A name token group must use the or connector
  - Attribute values specified as defaults in attribute definition list declarations must be literals (SGML allows them not to be even when SHORTTAG is NO)
- Element type declarations
  - Associated element type in element type declaration cannot be a name group
  - In an element declaration, a generic identifier cannot be specified as a rank stem and rank suffix (SGML allows this even when the RANK feature is NO)
  - Minimization parameters in element declarations are not allowed
  - RCDATA declared content are not allowed
  - CDATA declared content are not allowed
  - Content models cannot use the and connector
  - Content models for mixed content have a restricted form
  - Inclusions are not allowed
  - Exclusions are not allowed
- Comments
  - A parameter separator cannot contain comments; this means that markup declarations (other than comment declarations) cannot contain comments
  - Empty comment declarations (<!> in the reference concrete syntax) are not allowed
  - A comment declaration cannot contain more than one comment
  - In a comment declaration, an S separator is not allowed before the final MDC
- Processing instructions
  - Processing instructions must start with a name (the PI target)
  - A processing instruction whose PI target is xml can only occur at the beginning of a external entity and must be an XML declaration if it occurs in the document entity, and otherwise an text declaration
  - A PI target must not match [Xx][Mm][Ll] unless it is xml
- Marked sections
  - In marked section declarations, TEMP status keyword is not allowed
  - RCDATA marked sections are not allowed
  - INCLUDE/IGNORE marked sections are not allowed in the document instance
  - In a marked section declaration, a status keyword specification that contains no status keywords is not allowed
  - In a marked section declaration, a status keyword specification cannot contain more than one status keyword
  - Marked sections are not allowed in the internal subset
  - Parameter separators are not allowed in status keyword specifications in the document instance; in particular, parameter entity references are not allowed
- Other
  - Names beginning with [Xx][Mm][Ll] are reserved
  - The SGML declaration must be implied and cannot be explicitly present in the document entity
  - When < and & occur as data, they must be entered as < and &
  - A parameter separator required by the formal syntax must always be present and cannot be omitted when it is adjacent to a delimiter

XML predefines the semantics of the attributes xml:space and xml:lang. It also reserves all attribute, element type and notation names beginning with [Xx][Mm][Ll].

XML requires that an SGML parser use an entity manager that behaves as follows:

- Lines are terminated by newline (Unicode code #X000A) rather than being delimited by RS and RE as with a typical SGML entity manager
- System identifiers are treated as URLs
- The entity manager must support entities encoded in UTF-16 and UTF-8, and must be able automatically to detect which encoding an entity uses based on the presence of the byte order mark
- The entity manager should be able to recognize the encoding declaration in the XML declaration and

encoding PI and use it to determine the encoding of entity

XML imposes requirements on the information that a parser must make available to an application.

XML depends on the following changes to SGML made by Web SGML Adaptations Annex:

- HCRO delimiter (for hex numeric character references); for XML this is &#x
- EMPTYNRM feature that allows elements declared EMPTY to have end-tags
- NESTC delimiter
- Duplicate enumerated attribute tokens are allowed
- Relaxation of rules on use of parameter entity references inside groups
- Multiple ATTLIST declarations for a single element type
- ATTLIST declarations which don't declare any attributes
- KEEPRSRE feature that turns off SGML's rules for ignoring RSs and REs
- Fully-tagged SGML documents; a document that is fully-tagged but not type-valid is a conforming SGML document; this makes all XML documents, including those that are well-formed but not valid, conforming SGML documents
- Predefined data character entities in the SGML declaration (for lt, amp and so on)
- Unlimited capacities and quantities

The Web SGML Adaptations Annex also enables some XML restrictions to be enforced in SGML:

- SHORTTAG is unbundled, so the SGML declaration can allow attribute defaulting and NET without allowing other SHORTTAG constructs
- The SGML declaration can assert that a document is integrally stored, which disallows improperly nested entity references in content

2 Transforming SGML to XML

For most restrictions in XML that go beyond SGML, it is possible to transform an SGML document automatically into a document that meets the restrictions, and is equivalent in the sense that it has the same ESIS. There are a number of restrictions for which this is not the case:

External SDATA entities, external CDATA entities These could be transformed into NDATA entities.

Subdocument entities These could be converted into NDATA entities with a notation that indicates that they are SGML or XML.

References to external data entities in content These could be transformed into an empty element with an attribute whose declared value is ENTITY.

Data attributes Since an external data entity can only be used in an ENTITY or ENTITIES attribute on an element, these could be transformed into other attributes on the element.

Internal SDATA entities References could be transformed into numeric character references to the appropriate Unicode character; if used in an entity or entities attribute, the entity will have to be made external.

Internal CDATA entities If used in an ENTITY or ENTITIES attribute, the entity will have to be made external (references to CDATA entities are not part of ESIS).

PI entities If they contain ?>, they cannot be converted into an XML PI. It could be an application convention that entity references are replaced in PIs. Also if they do not start with a name, they cannot be converted into a well-formed XML PI.

names An SGML document can have a concrete syntax which allows characters in names that XML does not allow in names.

Bijlage 29
Comparing CONTEXT and LATEX

Taco Hoekwater
CONTEXT Task Force
ntg-context@ntg.nl
taco.hoekwater@wkap.nl

abstract
Some aspects of CONTEXT and LATEX are compared: the political decisions, the offered functionality, size of the system, and relative speed.

keywords
CONTEXT, LATEX, comparison, copyright, size, speed, functionality

Over the past couple of months, CONTEXT has received a lot of attention and publicity in the TEX world. It seems like every second person I meet these days wants to learn about CONTEXT and is dying to know what the precise differences between CONTEXT and LATEX are. Here is a short article that tries to give an overview of the most user–visible differences.

Some larger categories that seem worth mentioning are the following:

- Functionality & Design
- Size
- Speed
- Politics & Support

Let's go over these differences. I am writing this mostly from the point-of-view of a person that currently uses LATEX and is trying to decide whether or not CONTEXT will be worth the trouble of going through yet another leaning curve.

Functionality & Design

There is one structural design decision that is most important: where LATEX is designed as a run-time extensible system (through packages) with lots of separate files that are loaded on startup, CONTEXT is conceived as a monolithic system, where all functionality is included into the format file. This difference has a large influence on the maintenance and development of the system.

Small extensions to the LATEX run-time system are relatively easy to implement, and as a consequence lots of people have indeed done so in the past. This is obviously an important advantage of LATEX from the viewpoint of a the 'casual user' who only has to ask a guru what the package is called that offers the desired functionality. One does not even have to know TEX to use LATEX.

But there is also a down-side: extensibility implemented this way forces the burdon of backward compatibility on the maintainers of the LATEX kernel. They cannot change the interface all of a sudden, breaking lots of existing macro code, simply because they found a better implementation of a certain problem. It is highly unlikely that packages like amsmath, fancyhdr and multicol will ever become more closely integrated with the LATEX kernel, although these packages (and a lot of other ones) could gain a lot of functionality if they would have better support in the kernel.

In CONTEXT, the situation is very much the other way around. Adding functionality is almost never simple, and even has to be approved by the system maintenance group. A more or less monolithic system needs a higher level of control to prevent 'trojan horse' code from building up in its kernel. In general, this means that extensions cannot be done by an 'average hacker': if you need functionality that is not catered for already, you probably have to wait around for a while until somebody considers it worth implementing.

On the other hand, CONTEXT does not have to stay compatible with lots of previous versions (if you want to save the older version, just make a copy of the format file), so really large improvements are a lot easier (read: might actually be implemented in a reasonable timeframe). CONTEXT is also a lot larger than the LATEX kernel is, so the functionality you ask for might indeed be present, even without you knowing.

Changing the layout

Here is an interesting difference for people that design layouts (called classes in LATEX and environments in CONTEXT): There is no low-level interface to CONTEXT. Everything is taken care of by high-level setup commands. This may sound limiting, but in practise these commands accept so much parameters that it is usually easy to get any desired effect.

For example, here are the definitions for `\section` in both systems:

In LATEX, one uses `\@startsection` to set up and define

the sectioning mechanism (examples have been taken from the MAPS definition files in both systems):

```
\def\section{\@startsection{section}{1}{\z@}
    {-1.5\baselineskip}{.5\baselineskip}{\large\bfseries}}
```

`\@startsection` takes six arguments, that specify the following things:

#1	name of the sectioning command
#2	structural level of the command
#3	indentation from left margin
#4	space above (negation blocks indentation)
#5	space below (negation gives a run-in heading)
#6	font style used

The syntax is concise, reasonably clear, and fairly flexible. But: if you want to do something that is not possible within the arguments of `\@startsection` (which is not unlikely) you have to devise your own commands. That means learning a lot about LaTeX internals and TeX macro programming, and you get only marginal support in the form of the `\@secdef` and `\addcontentsline` commands.

In CONTEXT, almost all sectioning commands are predefined, and those already defined commands accept parameters using a key–value system. Setting the parameters used while typesetting goes like this:

```
\setuphead
  [section]
  [style=\bfa,
   before={\blank[line,halfline]},
   after={\blank[halfline]}]
```

`\setuphead` has only two arguments: the sectioning command it applies to, and a list of settings. In the second argument, the following keywords are recognised (you have to guess the meanings, this article is not a manual): *style, textstyle, numberstyle, page, continue, head, before, after, command, numbercommand, textcommand, prefix, placehead, ownnumber, variant, color, distance, incrementnumber, indentnext.*

Since a lot of these arguments take TeX commands as settings, it is quite unlikely that you will need to define your own commands. It follows that in CONTEXT style design is a lot simpler than in LaTeX. In fact, lots of CONTEXT input files I have start off with twenty or thirty lines of code that setup the layout, without using any external files.

Run time loaded files

This brings us to the next difference: LaTeX typically loads somewhere between five and forty files at runtime, whereas CONTEXT usually only needs the layout and font specifications for the current document as a separate file.

This has a major impact on the structure of the filesystem, of course. LaTeX needs a tree of subdirectories to store the often needed configuration and extension files, where CONTEXT does all with only one `\input` directory, that can be virtually empty. A quick count on my harddisk showed 417 files in the LaTeX 'packages' tree, and only 17 CONTEXT run-time files (most of those are font specification files).

Functional comparison

Almost all of the functionality from the LaTeX kernel is supplied in CONTEXT. The only thing I can think of that is really missing is the `picture` environment (and one can consider that a feature: CONTEXT actively promotes the use of METAPOST instead).

Of course, a monolithic system is not as flexible as the run-time approach, but the following functionality from various LaTeX packages I am aware of (Piet van Oostrum could probably increase this list a bit) is supplied within CONTEXT (albeit usually taking a slightly different approach): *a3, a4, a5, abbrev, afterpage, alltt, amssymb, amstext, babel, bm, boxedminipage, chapterbib, changebar, color, dcolumn, doc, doublespace, draftcopy, endfloat, endnotes, enumerate, example, exscale, fancybox, fancyhdr, fancyvrb, flafter, fleqn, float, fnpara, fontenc, footnote, ftnright, graphic*, graphpap, here, hhline, hyperref, ifthen, indentfirst, inputenc, keyval, lablst, layout, leqno, letterspace, longtable, lscape, ltxdoc, lucida*, makeindx, marks, metapost, mfnfss, minitoc, moresizes, moreverb, multicol, multidx, multirow, picins, picinpar, psnfss, samepage, showkeys, sidefloat, slides, syntonly, sub*, tabularx, testpage, theorem, trace*, verbatim, vpage, wrapfig, xspace*

There are some things, however, that are not (yet) implemented. I consider the following to be the most important packages, but the same remark as above applies to this list as well: *amsmath, array, bib*, breqn, natbib.*

And then there is some extra functionality that doesn't have a parallel in LaTeX as far as I know: the multilingual interface, interactive menu's, syntax highlighted verbatims, selective processing, run–time generated METAPOST drawings, reader profiles, collated and positioned output, two-pass page break/float optimalization, synonyms/glossaries, 'real' plain TeX compatibility, text buffers, page and paragraph backgrounds, color palettes, extended list item support, parallel documents, floats in multiple column output and finally grid snapping.

Size

Measurements of the relative run-time sizes of the two packages can be done on a couple of different levels. First and foremost, here are the TEX capacities needed to process a trivial 'Hello World' file in both systems. The files used to obtain the numbers are given below:

For LATEX:

```
\documentclass{minimal}
\begin{document}
Hello, World!
\end{document}
```

For CONTEXT:

```
Hello, world!
\bye
```

```
Here is how much of TeX's memory you used:
 13 strings
 191 string characters
 541649 words of memory
 2948 multiletter control sequences
 3640 words of font info for 14 fonts
 14 hyphenation exceptions
 14i,4n,10p,107b,137s stack positions
```

Figure 1 Parameter usage for LATEX.

```
Here is how much of TeX's memory you used:
 210 strings
 2081 string characters
 718520 words of memory
 16358 multiletter control sequences
 9049 words of font info for 28 fonts
 15 hyphenation exceptions
 50i,16n,65p,85b,898s stack positions
```

Figure 2 Parameter usage for CONTEXT.

The LATEX version resulted in the log file given in figure 1, the CONTEXT version gave figure 2. It is interesting to note that CONTEXT doesn't use all that much strings yet, although it almost exclusively uses a key–value system for options. On the other hand, CONTEXT uses a *lot* of control sequences, and puts quite a strain on the 'save stack' (that's the `898s`). Remember these are values for a minimal file, for a full production document the CONTEXT values will look more like those given in figure 3.

```
Here is how much of TeX's memory you used:
 1559 strings
 16366 string characters
 744952 words of memory
 17674 multiletter control sequences
 52309 words of font info for 104 fonts
 15 hyphenation exceptions
 53i,17n,91p,127b,1731s stack positions
```

Figure 3

What we see here is that apparently there are quite some macros that either create new strings or define other control sequences (since these definately have not been created by the loading of extra macro files).

Both the hash table size and the save stack are easily flooded in old–fashioned systems, but with the advent of web2c 7.0 it luckily became very easy to change the values of TEX's various memory paramaters. CONTEXT definately needs a *large* TEX. The values I use are given below (some of those parameters are set high because of other things I do, not for CONTEXT):

```
main_memory   =1500000 % words of inimemory
extra_mem_top = 500000 % extra high memory
extra_mem_bot = 500000 % extra low memory
font_mem_size = 200000 % Words of font info
font_max      =   1000 % Total number of fonts
hash_extra    =  30000 % Extra Hash table entries
pool_size     = 500000 % String values
string_vacancies=50000
max_strings   =  25000
pool_free     =  10000
trie_size     =  64000 % hyphenation trie
hyph_size     =   1000 % hyphenation exceptions
nest_size     =    200 % semantic groups
max_in_open   =     40 % input files
param_size    =   1000 % macro parameters
save_size     =  10000 % for saving values
stack_size    =    600 % input sources
```

Some other size numbers are the ones that are related to the size of the macro packages on disk, both the format file and the format's sources and support macros.

The typical CONTEXT format file is slightly over 2.3 Megabytes, compared to about 550 Kilobytes for LATEX. This 5 : 1 ratio reflects itself in the sources as well. Comparing the disk occupation roughly gives the

same ratio for the base system (not counting LaTeX's various 'standard' packages), and indeed the printed sources of ConTeXt are about five times as large a pile of paper as the LaTeX pile (you are not advised to try this unless you have a *lot* of paper and a very fast printer available).

Speed

Now let's talk about speed. ConTeXt offers quite a lot of functionality that is not available in LaTeX, and it is completely parameter–driven, so it seems quite reasonable to expect that it will be a bit slower. But how much is a bit?

Here are some timings, done with a very plain input file that creates 200 pages of impressively boring text. The ConTeXt file has been processed three times, to show the relative time spent in the output routine. For LaTeX, this did not seem to make much sense, since LaTeX was rather fast already and also because setting up LaTeX to use 6 point body fonts is not all that trivial.

System	Pointsize	Pages	Time
plain	10pt	181	21s
LaTeX	10pt (default)	244	27s
ConTeXt	12pt (default)	259	02m59s
ConTeXt	10pt	185	02m17s
ConTeXt	6pt	83	01m22s

We see that ConTeXt is indeed quite a lot slower. The second and third ConTeXt runs indicate that it spends a far larger amount of time in the output routine. (This test assumes that TeX's paragraph processing will not have a large impact on the timings. Because the DVI files stay roughly the same size, file I/O can be ignored as well. What is left over is mostly time spent in `\output`.)

It follows that the output routine of ConTeXt is a lot more complicated than LaTeX's (not really a surprise) and far less optimized to deal with trivial cases like the demonstration file (which is an interesting observation). Further tests showed that ConTeXt's version of `\reset@font` (which is called `\restoreglobalbodyfont`, by the way) really takes a lot of time. Disabling this macro gave a timing of 1m50s for the 10 point version, a speed gain of almost 30 seconds![1]

I decided to run the trivial file from the previous example with `\tracingmacros=1` to see what would happen, and indeed: whereas the LaTeX version resulted in a log file of only 72 kilobytes, the ConTeXt log became 1.3 megabytes! Interestingly, ConTeXt's `\output` specifications in the sources are almost trivial, and after reading them only a couple of times you actually understand what the output routine is doing.

An impressive part of the ConTeXt sources that shows off the amazing amount of structure is the following piece of code that comes from the end of the output routine:

```
\addpagecutmarks  0
\replicatepagebox 0
\scalepagebox     0
\mirrorpaperbox   0
\rotatepaperbox   0
\centerpagebox    0
\mirrorprintbox   0
\rotateprintbox   0
\offsetprintbox   0
\pagegoal=\dimen0
\box0{}}
```

This shows the next major difference between LaTeX and ConTeXt that I consider worth noting: LaTeX is optimized for speed of processing, whereas large portions of ConTeXt are optimized for ease of comprehension (even a lot of the optimized macros contain a literate programming comment that shows the non–optimized version of the macro and explains the algorithm used, including why and how the macro was optimized).

Politics & Support

There seems to be some confusion amongst people with regard to the copyright issues and politics of both packages. The following paragraphs give the 'definitive' answers for LaTeX and ConTeXt respectively.

Usage restrictions

The LaTeX2e copyright (`legal.txt`) says absolutely nothing about usage restrictions. The result is that legally there are no such restrictions, and LaTeX is effectively *free software.* However, donations to the LaTeX3 project are actively encouraged. The file `ltx3info.tex` says:

> "Although LaTeX may be distributed freely, the production and maintenance of the system does require expenditure of reasonably large sums of money. The LaTeX3 Project Fund has therefore been set up to channel money into this work. We know that some users are aware of this fund as they have already contributed to it—many thanks to all of them! If you want to know more about how you can help the project, see Page [..]—and thanks in advance for your generosity in the future."

1 This is no longer true in the latest beta version of ConTeXt, most because Hans proof–read this article for me.

There *are* usage restrictions on CONTEXT, however. Every file in the CONTEXT distribution begins with the following text:

> "This module is part of the CONTEXT macro–package and is therefore copyrighted by PRAGMA. Non–commercial use is granted."

This is a little bit on the terse side, but a more verbose official statement is in the readme file:

> "[On public use:] The most recent stable version of CONTEXT is available in the public domain and may be used by everyone, except by direct competitors of PRAGMA ADE.
>
> [On commercial use:] Systematic large scale commercial use of CONTEXT is permitted, given permission by PRAGMA ADE, conforming the conditions as stated below.
>
> With commercial use, we mean systematic document processing for third parties as well as large scale in–company use. Using CONTEXT in an iterative process towards an optimal document (for instance an article) is of course permitted.
>
> [...]
>
> Commercial users ... should pay a decent contribution."

It should be clear that you can use CONTEXT freely if you are a private person or non–commercial institution. Hans Hagen explained the situation to me as follows (the words are as I remember them, so please don't cite me):

> "We (PRAGMA) have spent a lot of time developing CONTEXT. We are a rather small company, and this has been a major investment for us. We are quite happy to offer our macros to the general public, but we would hate to find ourselves in the situation where we actually loose income because customers or competitors start using CONTEXT themselves.
>
> If you think you might belong in one of those two categories, you should contact PRAGMA for licensing information."

Some further discussion with Hans resulted in the following two lists.

Here are some groups of users that are expressly allowed to use CONTEXT without contacting PRAGMA:

1. Companies that have to work with unsolicited author–supplied CONTEXT input files.
2. Typesetting companies (small or large) that use CONTEXT internally, but only on an incidental basis and that do not advertise this fact.
3. Non-profit organisations.
4. In all other cases, barring those enumerated below: if the total annual production of pages is less than 2.000, it is safe to presume that you belong to this group.

And here are some groups that are expressly forbidden to use CONTEXT without prior written consent by PRAGMA:

1. Publishing houses.
2. Typesetting companies that intend to advertise with the fact that they use and/or accept CONTEXT input.
3. Typesetting companies that intend to use CONTEXT as their base means of production.
4. Governmental institutions.

Distribution restrictions

Both systems share restrictions on distribution of the system and distribution of changed files. The following text has been adopted from `legal.txt` from the LATEX distribution (with some exceptions deleted, this in *not* the legal version!).

> Redistribution of unchanged files is allowed provided that all files that belong to the system are distributed.
>
> The distribution of changed versions of certain specified files (most notably, the font definition files) included in the system are allowed under the following restrictions:
>
> - You rename the file before you make any changes to it. Any such changed files should be distributed under conditions that ensure that those files, and any files derived from them, will never be redistributed under the names used by the original files in the distribution.
> - You change the 'identification string' to clearly indicate that the file is not part of the standard system.
> - You change the 'error report address' so that we do not get error reports for files not maintained by us.
> - You acknowledge the source and authorship of the original version in the modified file.
> - You also distribute the unmodified version of the file.
>
> The above restrictions are not intended to prohibit, and hence do not apply to, the updating, by any method, of a file so that it becomes identical to the latest version of that file in the Standard system.

I have the distinct impression that this is actually more restrictive than most people seem to think it is (I know it was a lot more restrictive than I myself thought it was). Of course, the restrictions serve only two purposes: protection of the rights of people that have written the original work, and protection of the integrity of LaTeX as a system.

CONTEXT has the second problem in an even higher exponent: since almost all source files are included into the format, there is no telling what would happen if their contents cannot be guaranteed. To make sure CONTEXT will stay consistent without requiring a huge amount of maintenance, there is a very simple rule: You are *not* allowed to distribute changed or derived files at all, and distribution is only allowed under control of one of the TeX User Groups.

Support & Availability

Support for LaTeX is typically handled by the various user groups (for dutch users: the `tex-nl` mailing list. For subscription information, see elsewhere in this MAPS), and globally on the `comp.text.tex` newsgroup. Apart from these, there is a separate address for bug reports. See `bugs.txt` from the distribution for more information.

All support for CONTEXT is handled by the 'CONTEXT Task Force', a group of 'pioneer users' from different countries that serve as a filter between users and PRAGMA.

This group of people monitors a mailing list kindly provided by the Dutch TeX User Group: `ntg-context@ntg.nl`. Subscription requests should be sent to `majordomo@ntg.nl` with body "subscribe ntg-context". The mailing list can also be used for bug reports and feature requests.

LaTeX and the various contributed files are available from CTAN, and the base system together with lots of packages is currently part of (almost?) all TeX distributions. CONTEXT is distributed from a central location: `http://www.ntg.nl/context`. Soon, the sources will also be uploaded to CTAN, and CONTEXT is quickly becoming part of the current TeX distributions.

A final word

It should be clear that there are some major differences between the two packages. In fact, maybe the only thing that they *do* have in common is that they are both large, independantly developed TeX macro packages.

They may appeal to completely different types of users: Creating text books and manuals (including interactive ones) with CONTEXT is surprisingly easy, but LaTeX can handle highly demanding scientific articles a little better and faster.

Hans Hagen
pragma@pi.net

Bijlage 30

Pretty printing TEX, MetaPost, Perl and JavaScript

Keywords
verbatim, MetaPost, Perl, JavaScript, CONTEXT

abstract
Although for real pretty printing of sources one has to use CWEB like environments, TEX can also do a pretty job rather well. The CONTEXT verbatim environment has pretty printing built in. One can either use colors of fonts. The latter is used in the MAPS, the former in this article.

A few years ago I implemented a verbatim environment that supports coloring of TEX. The source code can be found in `supp-ver.tex` and was presented in MAPS 16. Although primary meant for CONTEXT, this module is rather generic, which is proved by the fact that it is used for typesetting verbatim in the MAPS.

In this article I will introduce the successor of this module: `verb-ini.tex`. This new module is backward compatible, but offers a more general solution for pretty printing. The main enhancement is that the pretty printing interpreter is generalized and supports not only TEX, but also METAFONT and METAPOST, as well as PERL and JAVASCRIPT code. These filters are defined in the files `verb-*.tex`.

I needed the extensions because in CONTEXT metagraphics can be included in the document source. That way users can produce run time and layout dependant graphics. The PERL pretty printer was needed after I reimplemented TEXUTIL in PERL. Pretty printing of JAVASCRIPT came around when I wrote a calculator demo for the PDF platform that demonstrates how TEX, METAPOST and JAVASCRIPT have come together.

Before I go into detail, I'll show some examples of pretty printed code. First a TEX example.

```
\def\SomethingBoxed#1%
  {\framed[width=10cm,offset=.25cm]{#1}}

\SomethingBoxed{Something Boxed}
```

In METAPOST the visualization involves special treatment of reserved words. As one can see, both colors and fonts are used. Keywords like `btex` are handled according to their meaning and disable the interpretation of the following tokens until `etex` is met.

```
beginfig(1);
  draw fullcircle xscaled 200 yscaled 100;
  label(btex draw a {\em test} shape etex, (0,0));
endfig;
```

In TEXUTIL, the next piece of PERL deals with stripping unwanted characters from strings. Making strings healthy is needed for proper sorting of index entries.

```
sub SanitizedString
  { my ($string) = $_[0]; # my o my
```

```
    if ($ProcessQuotes)
      { $string =~ s/\\([\^\"\'\’\~\,])/$1/gio;
        $copied = $string;
        $copied =~ s/([\^\"\'\’\~\,])([a-zA-Z])/$ASCII{$1}/gio;
        $string =~ s/([\^\"\'\’\~\,])([a-zA-Z])/$2/gio;
        $string=$string.$copied }
    $string =~ s/\\-|\|\||/\-/gio;
    $string =~ s/\\[a-zA-Z]*|  |\{|\}//gio;
    return $string }
```

When defined, the PERL interpreter also understands special functions, like:

```
use Getopt::Long;
```

The JAVASCRIPT interpreter is implemented on top of the PERL one. The main difference lays in the way comments are handled: `//`, `/*` and `*/` versus `#`.

```
i = 1;
while (i<=100)
 { Stack[i] = ""; // We're talking about a stack of strings!
   i++ }          // This is in fact i = i+1;
if (Done)
  { Stack[1] = "in a while" }
else              /* such an else is optional */
  { Stack[1] = "at once" }
```

Due to the fact that the verbatim environment is sort of object oriented, each pretty printer get's it own commands:

```
\startTEX ··· \stopTEX
\startMP  ··· \stopMP
\startPL  ··· \stopPL
\startJV  ··· \stopJV
```

For plain TEX users these commands are available after loading the macros:

```
\input verb-ini
```

The interpreter is enabled by saying:

```
\setcolorverbatim
```

The current meaning of this macro takes care of POSTSCRIPT colors, but tuning the visualization to his or her personal needs, is not that hard.

In CONTEXT, users can not only use the `\start..` commands mentioned, but also adapt some characteristics of each individual verbatim environment, like:

```
\setuptyping[MP][margin=2em,space=on]
```

Of course one can use the commands `\starttyping`, `\typefile`, and `\typebuffer`. These obey the settings of the general typing environment, like:

```
\setuptyping[option=TEX,margin=1em]
```

The command `\typefile` uses the file extension to automatically determine the pretty interpreter to be used.

The colorization is implemented using the CONTEXT palet mechanism. This mechanism enables users to define collections of colors that can be switched as a whole. So rather than redefining a specific shade of red, yellow or whatever, one just enables ano-

ther palet that has flavors of them defined. The default colors are defined as:

```
\definecolor [colorprettyone]   [r=.9, g=.0, b=.0] % red
\definecolor [colorprettytwo]   [r=.0, g=.8, b=.0] % green
\definecolor [colorprettythree] [r=.0, g=.0, b=.9] % blue
\definecolor [colorprettyfour]  [r=.8, g=.8, b=.6] % yellow

\definecolor [grayprettyone]    [s=.30]
\definecolor [grayprettytwo]    [s=.45]
\definecolor [grayprettythree]  [s=.60]
\definecolor [grayprettyfour]   [s=.75]
```

There are two main palets, one for color and one for gray printing:

```
\definepalet
  [colorpretty]
  [  prettyone=colorprettyone,
     prettytwo=colorprettytwo,
   prettythree=colorprettythree,
    prettyfour=colorprettyfour]

\definepalet
  [graypretty]
  [  prettyone=grayprettyone,
     prettytwo=grayprettytwo,
   prettythree=grayprettythree,
    prettyfour=grayprettyfour]
```

These palets are inherited by the specific pretty palets, for instance:

```
\definepalet [MPcolorpretty]  [colorpretty]
\definepalet [MPgraypretty]   [graypretty]
```

By default we have:

```
\setuptyping[MP][palet=MPcolorpretty]
```

but when needed, one can specify another palet. Of course, such a palet should be defined first in terms of `prettyone` upto `prettyfour`.

In CONTEXT one could (and still can) make spaces visible, obey tabs and embed TEX commands (using a different escape character). New however is the way multiple empty lines and breaking paragraphs are handled. From now on, by default, multiple empty lines are concatenated into one. Also, by default, the first two and last two lines are always kept together.

Another new feature is dedicated to Kees van der Laan, who, as a MAPS author, would like to see the pretty printer adapt its interpretation to TEX's current active character state. The next piece of TEX code shows this feature:

```
\bgroup
\catcode`\|=\@@escape %%\|\
\catcode`\\=\@@active %%\\+
|gdef|dohandlenewpretty#1%
  {|def|dodohandlenewpretty##1%
     {|getprettydata{\}%
      |let|newprettytype=|prettytype
      |getprettydata{##1}%
      |ifnum|prettytype=|newprettytype
```

```
      |let|next=|newpretty
    |else
      |def|next{|newprettycommand{#1}##1}%
    |fi
    |next}%
  |def|donohandlenewpretty##1%
    {|newprettycommand{#1}##1}%
  |handlenextnextpretty
    |dodohandlenewpretty|donohandlenewpretty}
|egroup
```

Without the switch, the first few lines would look like:

```
\bgroup
\catcode`\|=\@@escape
\catcode`\\=\@@active
|gdef|dohandlenewpretty#1%
  {|def|dodohandlenewpretty##1%
     {|getprettydata{\}%
      |let|newprettytype=|prettytype
```

The redefinitions are invoked by the double comment sign, followed by a backslash. The next (non space) token will be interpreted as the one following it. In our example the | will be visualized as the \ and the \ as the + token.

When followed by a space, the double comment takes the next token as an interpreter command to be executed. An example demonstrates this feature.

```
\ziezo{test}          %%\ P   ##\ B##\ T % enter PERL mode       %%\ E
if $test eq "test"    ##\ B   ##\ B##\ T % begin group (\bgroup) %%\ E
if $test eq "test";   ##\ T   %%\ B%%\ T % enter TEX mode        %%\ E
\ziezo{test}          %%\ M   %%\ B%%\ T % enter METAPOST mode   %%\ E
draw (0,0)--(10,10);  %%\ E   ##\ B##\ T % end group (\egroup)   %%\ E
if $test eq "test";
```

this was typed in as (forget the comments):

```
\ziezo{test}          %%\ P
if $test eq "test"    ##\ B
if $test eq "test";   ##\ T
\ziezo{test}          %%\ M
draw (0,0)--(10,10);  %%\ E
if $test eq "test";
```

When in CONTEXT one wants to pass data from TEX to JAVASCRIPT, one can use the prefix TEX. This prefix is interpreted as *pretty print the next string as a TEX one*. Of course the keyword TEX is stripped before the JAVASCRIPT is shipped out. So:

```
var MinLevel    = -TEX \MinLevel;
var MaxLevel    = TEX \MaxLevel;
var Level       = 1;
```

becomes in pretty typography:

```
var MinLevel    = -TEX \MinLevel;
var MaxLevel    = TEX \MaxLevel;
var Level       = 1;
```

Bijlage 31

The Calculator Demo

Integrating TEX, MetaPost, JavaScript and PDF

Hans Hagen
pragma@pi.net

Keywords
MetaPost, JavaScript, PDF, CONTEXT, pdfTEX

abstract

Due to its open character, TEX can act as an authoring tool. This article demonstrates that by integrating TEX, METAPOST, JAVASCRIPT and PDF, one can build pretty advanced documents. More and more documents will get the characteristics of programs, and TEX will be our main tool for producing them. The example described here can be produced with PDFTEX as well as traditional TEX.

Introduction

When Acrobat Forms were discussed at the PDFTEX mailing list, Phillip Taylor confessed: "... they're one of the nicest features of PDF". Sebastian Ratz told us that he was "... convinced that people are waiting for forms.". A few mails later he reported: "I just found I can embed JAVASCRIPT in forms, I can see the world is my oyster" after which in a personal mail he challenged me to pick up the Acrobat Forms plugin and wishing me "Happy JavaScripting".

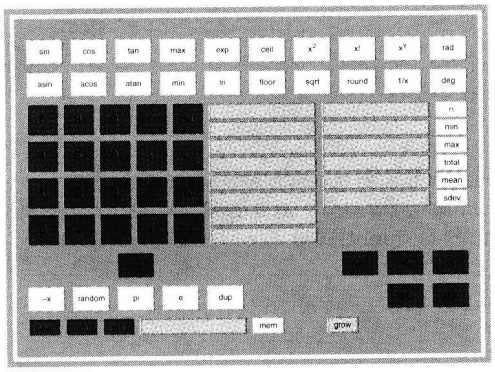


Figure 1 The calculator demo.

At the moment that these opinions were shared, I already had form support ready in CONTEXT, so picking up the challenge was a sort of natural behaviour. In this article I'll describe some of the experiences I had when building a demo document that shows how forms and JAVASCRIPT can be used from within TEX. I also take the opportunity to introduce some of the potentials of PDFTEX, so let's start with introducing this extension to TEX.

Where do we stand

While ε-TEX extends TEX's programming and typographic capabilities, PDFTEX primarily acts at the back end of the TEX processor. Traditionally, TEX was (and is) used in the production chain:

ASCII → TEX → DVI → whatever

The most versatile process probably is:

ASCII → TEX → DVI → POSTSCRIPT

or even:

ASCII → TEX → DVI → POSTSCRIPT → PDF

All functionality that TEX lacks, is to be taken care of by the DVI postprocessing program, and that's why TEX can do color and graphic inclusion. Especially when producing huge files or files with huge graphics, the POSTSCRIPT → PDF steps can become a nuisance, if only in terms of time and disk space.

With PDF becoming more and more popular, it will be no surprise that Han The Thanh's PDFTEX becomes more and more popular too among the TEX users. With PDFTEX we can reduce the chain to:

ASCII → TEX → PDF

The lack of the postprocessing stage, forces PDFTEX (i.e. TEX) to take care of font inclusion, graphic inserts, color and more. One can imagine that this leads to lively discussions on the PDFTEX mailing list and thereby puts an extra burden on the developer(s). Take only the fact that PDFTEX is already used in real life situations while PDF is not stable yet.

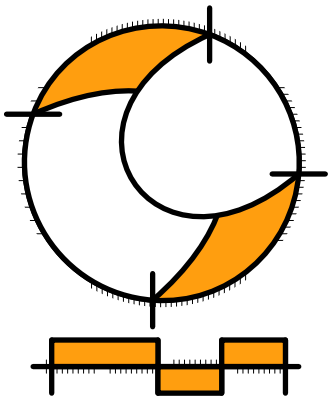
To those who know PDF, it will be no surprise that PDFTEX also supports all kind of hyper referencing. The version[1] I

Integrating TEX, MetaPost, JavaScript and PDF

Hans Hagen

Spring 1998

Due to it's open character, TEX can act as an authoring tool. This article demonstrates that by integrating TEX, METAPOST, JAVASCRIPT and PDF, one can build pretty advanced documents. More and more documents will get the characteristics of programs, and TEX will be our main tool for producing them. The example described here can be produced with PDFTEX as well as traditional TEX.



PRAGMA

Advanced Document Engineering | Ridderstraat 27 | 8061GH Hasselt NL
tel: +31 (0)38 477 53 69 | e-mail: pragma@wxs.nl | ConTEXt: www.pragma-ade.nl

This article was first published in the *Minutes and Appendices* of the NTG (Nederlandstalige TEX Gebruikersgroep), Issue 98.1. It was presented as paper at the 1998 annual meeting of the (international) TEX User Group that took place in Toruń (Poland).

Introduction

When Acrobat Forms were discussed at the pdfTeX mailing list, Phillip Taylor confessed: "... they're one of the nicest features of PDF". Sebastian Ratz told us that he was "... convinced that people are waiting for forms.". A few mails later he reported: "I just found I can embed JavaScript in forms, I can see the world is my oyster" after which in a personal mail he challenged me to pick up the Acrobat Forms plugin and wishing me "Happy JavaScripting".

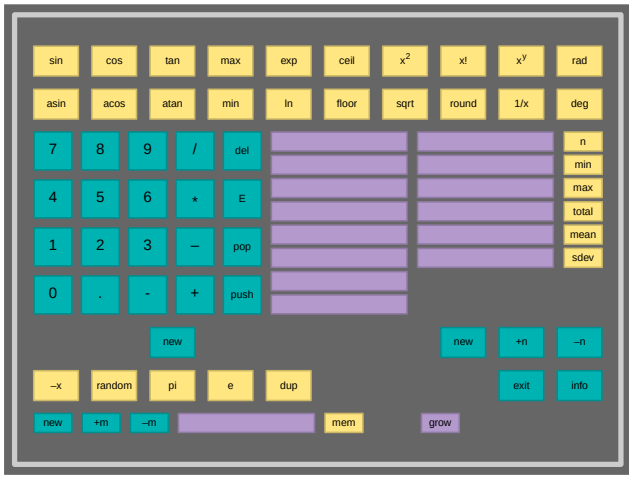


Figure 1 The calculator demo.

At the moment that these opinions were shared, I already had form support ready in ConTeXt, so picking up the challenge was a sort of natural behaviour. In this article I'll describe some of the experiences I had when building a demo document that shows how forms and JavaScript can be used from within TeX. I also take the opportunity to introduce some of the potentials of pdfTeX, so let's start with introducing this extension to TeX.

Where do we stand

While ε-TeX extends TeX's programming and typographic capabilities, pdfTeX primarily acts at the back end of the TeX processor. Traditionally, TeX was (and is) used in the production chain:

ASCII → TeX → DVI → whatever

The most versatile process probably is:

ASCII → TeX → DVI → PostScript

or even:

ASCII → TeX → DVI → PostScript → PDF

All functionality that TeX lacks, is to be taken care of by the DVI postprocessing program, and that's why TeX can do color and graphic inclusion. Especially when producing huge files or files with huge graphics, the PostScript → PDF steps can become a nuisance, if only in terms of time and disk space.

With PDF becoming more and more popular, it will be no surprise that Hàn Thế Thành's pdfTeX becomes more and more popular too among the TeX users. With pdfTeX we can reduce the chain to:

ASCII → TeX → PDF

The lack of the postprocessing stage, forces pdfTeX (i.e. TeX) to take care of font inclusion, graphic inserts, color and more. One can imagine that this leads to lively discussions on the pdfTeX mailing list and thereby puts an extra burden on the developer(s). Take only the fact that pdfTeX is already used in real life situations while PDF is not stable yet.

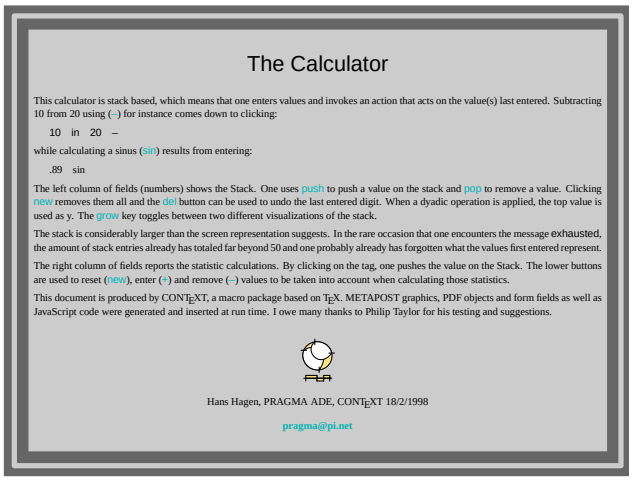


Figure 2 The help information screen.

To those who know PDF, it will be no surprise that pdfTeX also supports all kind of hyper referencing. The version[1] I used when writing this article supports:

1. link annotations
2. screen handling
3. arbitrary annotations

[1] Currently I'm using β-version 1.12g.

where especially the last one is accompanied by:

4. form objects
5. direct objects

and of course there is also:

6. extensive font support

Be prepared: PDFTEX's font support probably goes (and certainly will go) beyond everything DVI drivers as well as ACROBAT supports!

TEX stands in the typographic tradition and therefore has unsurpassed qualities. For many thousands of years people have trusted their ideas to paper and used glyphs for communication. The last decades however there has been a shift towards media like video, animations and interactive programs and currently these means of communication meet in hyper documents.

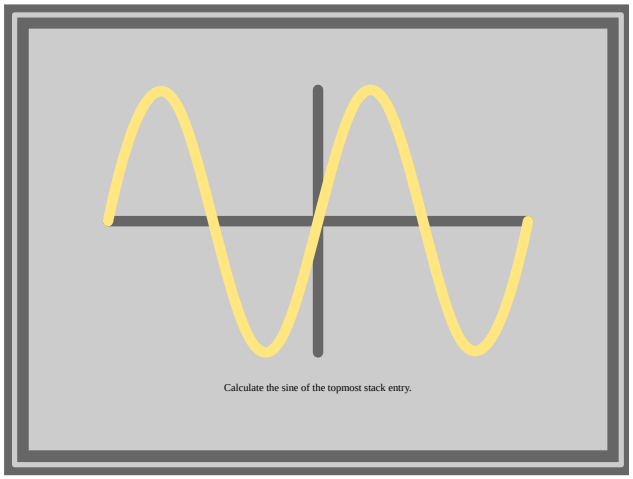


Figure 3 The $\sin(x)$ screen.

Now what has this to do with PDFTEX. Recently this program started to support the PDF annotations other than the familiar hyperlink ones. As we will see later on, this enables users of TEX to enhance their documents with features that until now had to be programmed with dedicated tools, which could not even touch TEX's typographic quality. This means that currently TEX has become a tool for producing rather advanced documents within the typographic and (largely paper based) communication traditions. Even better, by using PDF as medium, one can produce very sophisticated interactive documents that are not bound to ill documented standards and programs and thereby stand a better chance to be accessible for future generations.

The calculator demo

The document described here is produced with CONTEXT. This document represents a full featured calculator which took me about two weeks to design and build. Most of the time was spend on defining METAPOST graphics that could explain the functionality of the buttons.[2] Extending CONTEXT for supporting JAVASCRIPT took me a few days and the rest of the time was spend on learning JAVASCRIPT itself.

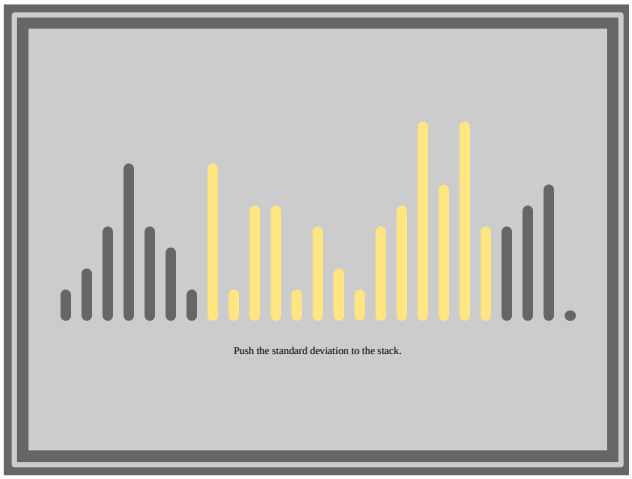


Figure 4 The standard deviation screen.

The calculator demo was first developed using DVIPSONE and ACROBAT. At that moment, PDFTEX did not yet provide the hooks needed, and the demo thereby served as a source of inspiration of what additional functionality was needed to let PDFTEX produce similar documents.

Throughout this article I show some of the screens that make up the calculator demo. These graphics are no screen dumps but just POSTSCRIPT inclusions. Just keep in mind that when using TEX, one does not need bitmap screen dumps, but can use snapshots from the real document. A screen, although looking as one graphic, consist of a background with frame, a centered graphic, some additional text and an invisible active area the size of the gray center.

The demo implements a stack based calculator. The stack can optionally grow in two directions, de-

[2] This included writing some auxiliary general purpose METAPOST macros.

pending on the taste of the user. Only the topmost entries of about 50 are visible.

The calculator demo, called `calculator.pdf`, itself can be fetched from the PDFTEX related site:

```
http://www.tug.org/applications/pdftex
```

or from the CONTEXT repository at:

```
http://www.ntg.nl/context
```

The calculator is defined in one document source file, which not only holds the TEX code, but also contains the definitions of the METAPOST graphics and the JAVASCRIPT's. I considered including a movie (video) showing an animation of our company logo programmed in METAPOST and prepared in Adobe Premiere, but the mere fact that movies are (still) stored outside the PDF file made me remove this feature.

Now keep in mind that, when viewing the calculator PDF file, you're actually working with a document, not a program. A rather intelligent document for that matter, but still a document.

Forms and annotations

Before I go into details, I'll spend some words on forms and annotations in PDF. To start with the latter, annotations are elements in a PDF file that are not related to (typo)graphic issues, like movies and sound, hyper things, navigation and fill–in–forms. Formally annotations are dealt with by drivers plugged into the graphic engine, but in practice some annotations are handled by the viewer itself.

Forms in PDF are more or less the same as in HTML and once filled in can be send over the net to be processed. When filling in form fields, run time error checking on the input can prevent problems later on. Instead of building all kind of validation options into the form editor, such validations are handled by either a dedicated plugin, or better: by means of JAVASCRIPT. Therefore, one can attach such scripts to all kind of events related to form editing and one can launch scripts by associating them to active, that is clickable, areas on the screen.

So we've got fields, which can be used to let users provide input other than mere clicks on hyper

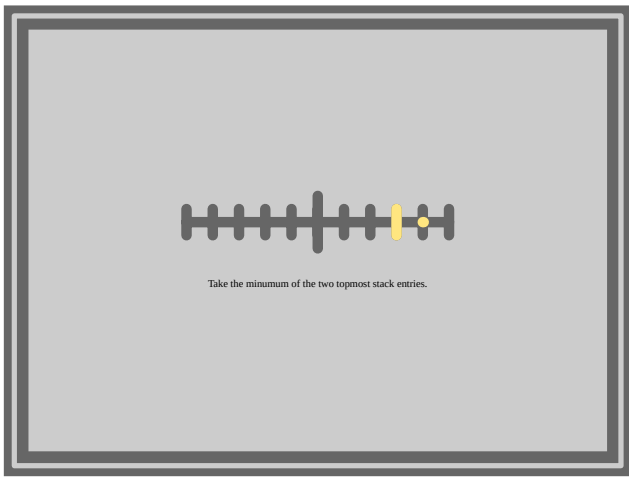


Figure 5 The $\min(x, y)$ screen.

links, we've got run time access to those fields using JAVASCRIPT, and we can let users launch such scripts by mouse events or keystrokes, either when entering data or by explicit request.

Currently entering data by using the keyboard is prohibited in the calculator. The main reason for this is that field allocation and access are yet sort of asynchronic and therefore lead to confusion.[3]

So, what actually happens in the calculator, is that a user clicks on a visualized key, thereby launching a JAVASCRIPT that in turn does something to field data (like adding a digit or calculating a sine), after which the field data is updated.

JAVASCRIPT

Writing this demo at least learned me that in fact support for JAVASCRIPT is just another sort of referencing and therefore needed incorporation in the general cross referencing scheme. The main reason is that for instance navigational tools like menus and buttons must have access to all cross reference mechanisms.

Consider for instance buttons. We already supported:

```
\button{...}[the chapter on whatever]
\button{...}[otherdoc::some topic]
\button{...}[previouspage]
\button{...}[PreviousJump]
```

[3] Initializing a field from within JAVASCRIPT is not possible unless the viewer has (at some dubious moment) decided that the field indeed exists.

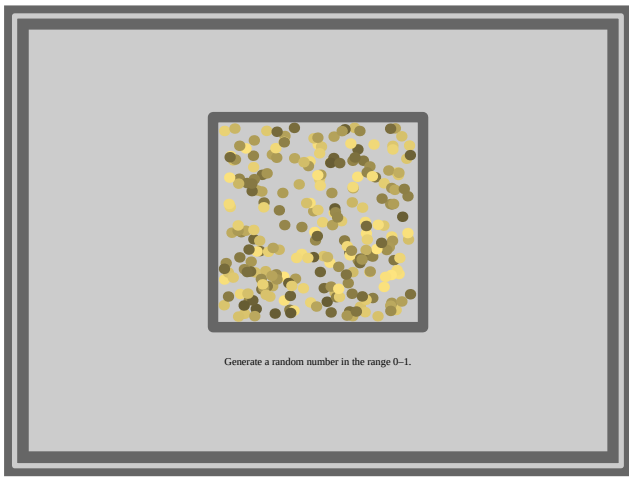


Figure 6 The random number screen.

Here the first reference is an internal one, often a chapter, a table or figure or a bibliography. The second one extends this class of references across documents. The third reference is a predefined internal one and the last reference gives access to viewer controls. As we can see: one scheme serves different purposes.

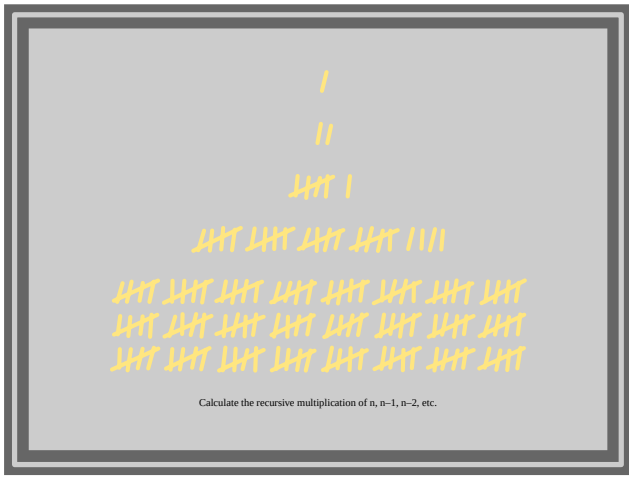


Figure 7 The period (.) screen.

Launching applications and following threads can quite easily be included in this scheme, but JAVASCRIPT support is different. In the calculator there are for instance 10 digit buttons that all do the same action and only differ in the digit involved. Here we want just one JAVASCRIPT to be reused 10 times. So instead of saying:

```
\button{0}[javascript 0]
\button{0}[javascript 1]
```

we want to express something like:

```
\def\SomeDigit#1%
  {\button{0}[javascript #1]}

\SomeDigit{4}
```

This means that in practice we need a referencing mechanism that:

- is able to recognize JAVASCRIPT
- is able to pass arguments to these scripts

So finally we end up with something:

```
\button{7}[JS(digit{7})]
```

This call tells the reference mechanism to access the JAVASCRIPT called `digit` and pass the value `7` to it. Actually defining the script comes down to just saying:

```
\startJScode{digit}
  Stack[Level] += String(JS_S_1);
  do_refresh(Level);
\stopJScode
```

One can pass as much arguments as needed. Here `JS_S_1` is the first string argument passed. Passing cross reference arguments is also possible. This enables us to let users jump to locations depending on their input. Such arguments are passed as `R{destination}` and can be accessed by `JS_R_1`.

Figure 8 The digit 7 screen.

In practice one will separate functions and calls by using preambles. Such preambles are document wide pieces of JAVASCRIPT, to be used whenever applicable.

```
\startJSpreamble{functions}
  // begin of common functions

  function do_digit(d)
    { Stack[Level] += String(d);
      do_refresh(Level) }

  // end of common functions
\stopJSpreamble
```

and:

```
\startJScode{digit}
  do_digit(JS_S_1);
\stopJScode
```

From these examples one can deduce that indeed the actual JAVASCRIPT code is included in the document source. It's up to TEX to pass this information to the PDF file, which in itself is not that trivial given the fact that one (1) has to strip comments, (2) has to convert some characters into legal PDF ones and (3) must pass arguments from TEX to JAVASCRIPT.

Simple cases like the digit code fragment, can also be passed as reference: `JS(digit{1})`. By default CONTEXT converts all functions present in the preambles into such references. One can organize JAVASCRIPTS into collections as well as postpone inclusion of preambles until they are actually used.

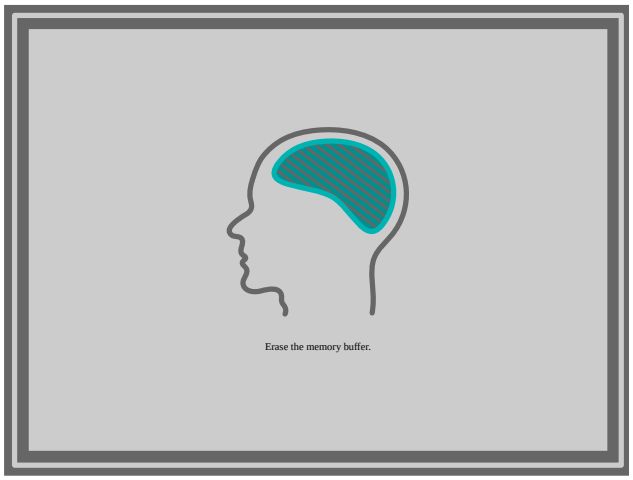


Figure 9 The memory erase screen.

Currently the only problem with including preambles lays in the mere fact that ACROBAT pdfmarks[4] not yet offer a mechanism to enter the JAVASCRIPT entries in the appropriate place in the document catalog, without spoiling the collected list of named destinations. Because CONTEXT can be instructed to use page destinations when possible, I could work around this (temporary) ACROBAT pdfmark and PDFTEX limitation. At the time this article is published, PDFTEX probably handles this conceptual weak part of PDF in an adequate way.

METAPOST graphics

All graphics are generated at run time using METAPOST. Like the previous mentioned script, METAPOST code is included in the source of the document. For instance, the graphic representing π is defined as:

```
\startuseMPgraphic{pi}
  pickup pencircle scaled 10;
  draw fullcircle
    scaled 150
    withcolor .4white;
  linecap := butt;
  ahlength := 25;
  drawarrow halfcircle
    scaled 150
    withcolor \MPcolor{action};
\stopuseMPgraphic
```

and called

```
\useMPgraphic{pi}
```

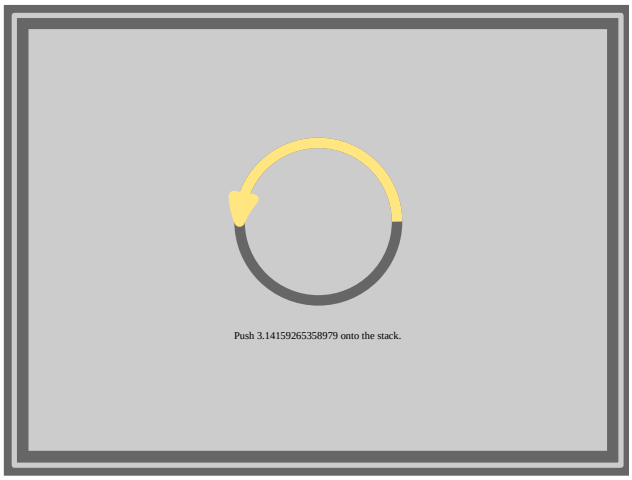


Figure 10 The π screen.

Just like the JAVASCRIPT preamble we can separate common METAPOST functions by defining inclusions. The next one automatically loads a module with some

[4] These are extensions to the POSTSCRIPT language.

auxiliary macros.

```
\startMPinclusions
  input mp-tool;
\stopMPinclusions
```

The mechanism for including METAPOST graphics is also able to deal with reusing graphics and running METAPOST itself from within TEX. In CONTEXT all processed METAPOST graphics are automatically translated into PDF by TEX itself, colors are converted to the current color space, and text is dealt with accordingly. Of course one needs to take care of proper tagging, but the next macro does this well:

```
\def\SomeShape#1#2%
  {\startreuseMPgraphic{shape:#1#2}
     draw fullcircle
       xscaled #1
       yscaled #2
   \stopreuseMPgraphic
   \reuseMPgraphic{shape:#1#2}}
```

Now we can say:

```
\SomeShape{100pt}{200pt}
\SomeShape{150pt}{180pt}
\SomeShape{120pt}{110pt}
```

Which just inserts three graphics with different sizes but similar line widths.

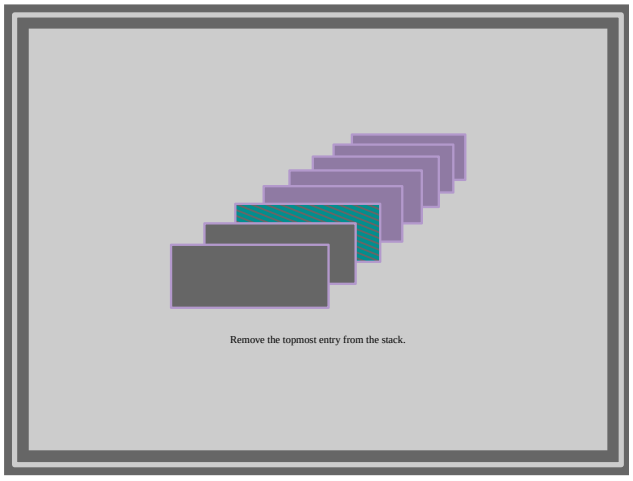


Figure 11 The pop stack screen.

Backgrounds

Now how do we attach such shapes to the buttons? Here we introduce a feature common to all framed things in CONTEXT, called overlays. Such an overlay is defined as:

```
\defineoverlay
  [shape]
  [\MPshape
     {\overlaywidth}
     {\overlayheight}
     {\overlaycolor}]
```

The shape called \MPshape is defined as:

```
\def\MPshape#1#2#3%
  {\startreusableMPgraphic{fs:#1#2#3}
     path p ;
     p := unitsquare
       xscaled #1
       yscaled #2;
     color c ;
     c := #3 ;
     fill p
       withcolor c ;
     draw p
       withpen pencircle scaled 1.5
       withcolor .8c ;
   \stopreusableMPgraphic
   \reuseMPgraphic{fs:#1#2#3}}
```

Such an overlay is bound to a particular framed thing by saying:

```
\setupbuttons[background=shape]
```

Here the right dimensions are automatically passed on to the overlay mechanism which in turn invokes METAPOST.

The calculator demo proved me that it is rather useful to have stacked backgrounds. Therefore the buttons, which have both a background (the METAPOST drawn shape) and behind that a sort of help button that is activated by clicking on the surroundings of the button, have their backgrounds defined as:

```
\setupbuttons
  [background={infobutton,shape}]
```

Actually we're stacking from back to top: an info button, the key bound button, the background graphic and the text. One rather tricky side effect is that stacked buttons interfere with the way active areas are

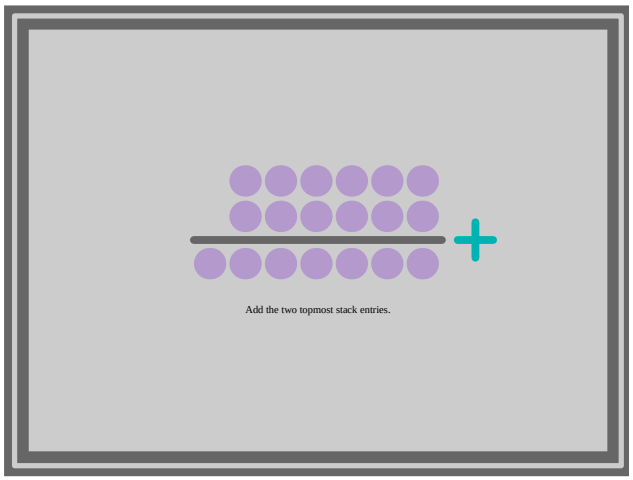


Figure 12 The addition (+) screen.

output. In this particular case we have to revert the order of the active areas by saying `\reversegototrue`.

Object reuse

The button and background graphics are generated once and used more than once. We already mentioned that METAPOST graphics can be reused. In practice this comes down to producing the graphic once and including it many times. In PDF however, one can also include the graphic once and refer to it many times. In PDF such reused objects are called forms, a rather unfortunate naming. So, in the calculator demo, all buttons with common shapes as well as the backgrounds are included only once. One can imagine that extending TEX with such features leads to interesting discussions on the PDFTEX discussion list.

Forms

Although still under construction, CONTEXT supports PDF fill-in-forms. The calculator demo demonstrates that such forms can be used as a (two way) communication channel to the user. Stack values, statistics and memory content are stored and presented in form fields, defined by saying something like:

```
\definefield[Stack.1][line][Results]
```

followed by

```
\field[Stack.1]
```

The characteristics of this line field are set by:

```
\setupfield
  [Results]
  [horizontal,frame]
  [width=fit,
   height=.5\ButtonWidth,
   background=shape,
   backgroundcolor=\MPcolor{stack},
   frame=off]
  [width=3.5\ButtonWidth,
   frame=off]
  [width=3.5\ButtonWidth,
   frame=off]
```

The reader needs some fantasy to grab the meaning of this rather overloaded setup. The first argument tags the characteristics, and can be considered something like a class in object oriented languages. The second argument tells CONTEXT how to typeset the field when labels are used, while the last three arguments specify the way fields, their labels and the envelop that holds them both together are typeset. In the calculator, the labels are suppressed.

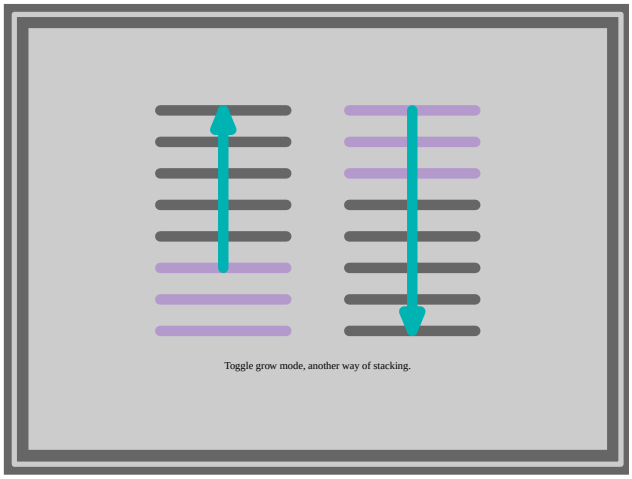


Figure 13 The grow mode screen.

One reason for decoupling definition and setup, that is, not attaching characteristics to individual fields, lays in the fact that I have applications in mind with thousands of fields and saving characteristics at the field level that would definitely overload TEX.

Where do we go

The previous examples show us quite clearly that, although being of old age in terms of computer programs, TEX is among the few applications that are able to adapt themselves rather fast to current develop-

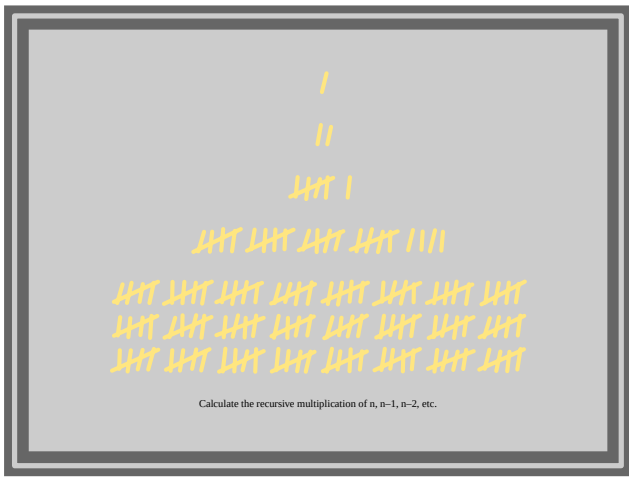


Figure 14 The *n*! screen.

ments while at the same time preserving the high quality and stability its users are accustomed to. As TEX gave mathematicians the means of circumventing the often lousy text editing and desk top publishing output in the early days of computing, TEX can give its users the high quality and stable authoring platform they need in this multi–media age. As demonstrated here, TEX can do a wonderful job not only in producing interactive documents, but in producing intelligent documents too.

Bijlage 32

Contending with Office suites

Siep Kroonenberg
Faculteit der Economische Wetenschappen
Rijksuniversiteit Groningen
n.s.kroonenberg@eco.rug.nl

abstract
The author is tried beyond endurance by current Office software.

keywords
Windows, suites, wordprocessors, graphics, export, import, OLE, frogs

Even a diehard TEX user cannot always disregard the World of Windows. I regret to admit that Windows NT still is my primary operating system. Now and then I am forced to contend with Office suites, and invariably am appalled at the shoddy engineering, the amount of silliness and the terrible PostScript support.

Quality engineering

Implementing a letterhead in Word and WordPerfect

The other day we had to implement our department's letterhead in the major wordprocessing applications in use at our department. My colleague Erik Frambach took care of the LATEX version, and I did the commercial word processors: two versions each of Word and WordPerfect.

I used to be quite enthusiastic about the desktop-publishing capabilities of WordPerfect 5.1. Eventually, though, print- and file corruption problems forced me to abandon WP for TEX. I find it hard to believe that people entrust their theses to such a house of cards, but they do, and they even manage to get their theses into print. So I had some hope that this would be a reasonably straightforward job. But it wasn't. The layout required exact positioning of items on the page and placing graphics in headers and footers close to the margin. Wordprocessors aren't designed for this. Neither is LATEX, but there the workarounds still work the next day.

With the newer Windows versions of Word and especially WordPerfect I ran into all the old aggravations: erratic printer output, file corruption, misbehaving graphics and also program crashes. It also turned out that neither Word nor WordPerfect were capable of creating a reasonable export for older versions. In the case of WordPerfect I even had to recreate the template for the older version from scratch. I ended up spending most of my time working around bugs and shortcomings in program design.

After I thought I was done with the job, I got a mail from a Word user that whenever he reopened an old letter, the logos had resized themselves to something enormous. Since there was an obvious and convenient workaround, I made no attempt to solve this problem.

Corel's latest and greatest: WordPerfect 8

WordPerfect 8 defeated me altogether. I managed to sort of import the WP 6.1 version but that was the end of my luck; crashes prevented me from making any headway.

The problems had started with installation of the program. It proved impossible to put together a consistent but reasonably compact installation. The Borland Database Engine, requiring about 8MB of disk space, was considered an essential component, as was the address book utility. In spite of all the unwanted installed items which were deemed essential, I still ran repeatedly into missing files, which had to be copied manually from the CD. And the number of registry entries was mind-boggling (but still a good deal less than those for Microsoft Office). The File/New dialogue presented me with a long list of templates which I hadn't installed and didn't want to know about. Attempts to clean up this list caused more program crashes. I know, this flame won't pass as a serious bug report. I just had to get it off my chest.

Exchanging graphics

Exporting from Excel

A question which crops up now and then on TEX-NL is how to get a chart from Excel into TEX. The trouble is that Excel relies on the clipboard and OLE exclusively for export. There are some workarounds: you can copy–and–paste to a vector drawing program such as CorelDRAW, and export from there to eps (Encapsulated PostScript). Or you can print to an eps file[1]. Such an eps file gets the full page as bounding box, so you should make sure beforehand that the chart covers as much of the page as possible, and that there are no unwanted page headers or -footers.

1. The Microsoft PostScript driver for Windows NT doesn't offer this option, but an Adobe PostScript driver with this option has finally become available. This driver only supports PostScript level 2 and higher.

About a year ago, I needed this, or so I thought. I needed to import them into a vector drawing program to edit colors, line style and type style.

Both the printer route and the clipboard route resulted in total fragmentation: each line segment of each graph and each letter of each word had become a separate object. Since the charts concerned were line graphs with lots of data points, usually two or three thoroughly intertwined line graphs per chart, rejoining the fragments was not practical: the converted files were useless.

Luckily, in this case I could bypass MS Office: the charts were also available as eps files created by a non-office application. The ps2ai untility bundled with Ghostscript turned these into perfectly good Illustrator files.

OLE

The O-word is OLE, Microsoft's protocol to make one program behave as a module within another. OLE is a very expensive but nevertheless ineffective way to let programs exchange data. Expensive, because use of OLE requires both the calling and the called program to be loaded in memory at the same time. Ineffective, because it is limited by the clipboard formats both programs understand. As to graphics, Bézier curves are likely to be reduced to polylines, and colors to rgb or worse. Between programs from one suite, it is usually not a problem, since they can support additional common proprietary formats. But exchange between a suite program and an external application can be very problematic indeed. I already cited my troubles with exporting charts from Excel.

Contrast this with exchange of graphics data via the eps (Encapsulated PostScript) format. For the exporting program, creating an eps file should be no more complicated than printing to a PostScript printer, which it must be able to do anyway. All that the importing application needs to understand about the file is its bounding box. For all its simplicity, this scheme allows import and printing of externally-generated graphics with all features intact, and works very well for TEX and for serious publishing applications. PostScript-hostile Office applications still manage to screw this up.

Exporting from Presentations

Although Presentations[2] does support eps export, this option is of limited use. If you create simple charts from spreadsheets in Presentations and export them in Encapsulated PostScript format, you have a fair chance of success. Even so, you may find that the size and placement of objects in the eps file is far from exact. Any font which is not Times or Helvetica though, even Symbol, gets replaced with Helvetica.

You may be tempted by Presentations' powerful drawing features to use it for other diagrams than spreadsheet charts. Be forewarned that more often than not, eps exports of such drawings are corrupted in some way, for example:

- lost rotation of text objects
- bogus rotation
- invalid PostScript
- stray objects
- altered line and fill attributes

You may have less trouble creating an eps file with the PostScript driver.

The role of the press

If computer magazines would be doing their job, they would put new software through some stress testing and check the software against lists of known bugs. Inevitably they would then in their reviews demolish the newer Office programs, and so force software vendors to produce better quality. Instead, they complain that Presentations has fewer multimedia gadgets than PowerPoint.

There is the story that if you put a frog in cold water and then you proceed to boil the water, and the frog with it, then the frog won't have the sense to jump out of the pan and save himself. This is how I feel with respect to Windows software. Bugs and bloat are making our work impossible, but since it happens incrementally, we fail to realize what is being done to our computing environment.

2. I am talking about versions 2 and 3. I have barely looked at version 8. I noticed, though, that there still were problems with fonts.

Bijlage 33

4TEX5.0: TDS, Web2C, and Windows 95/NT

Wietse Dol

1 Introduction

4TEX started of as a workbench for TEX under MS-DOS. Making a workbench for TEX, all its friends and many other useful programs needed for text processing, the 4TEX team decided to make the 4TEX systems as open as possible. This means that users could change the workbench, update the used files/programs and add new stuff whenever the user felled necessary (or when bugs had to be resolved). These conditions resulted in the decision to write 4TEX with the 4DOS batch programming language. Since 1991 4TEX has been enthusiastically used by people who needed a tool for writing letters, articles, reports, and books...

WINDOWS 95/NT gradually took over MS-DOS and many people asked for a Windows version of 4TEX. In December 1997 the 4TEX team (Erik Frambach and Wietse Dol) decided to (re)write a 4TEX for Windows that should have the same functionality as the MS-DOS version. The 4TEX for WINDOWS 95/NT also should be an open system, i.e. users should be able to add utilities and make changes to the system they prefer. The 4TEX for Windows is at this moment under beta testing and we hope that in a few months time Addison Wesley will publish the book and double CD-rom.

The 4TEX main program was written with Borlands Delphi 3.0 and uses Windows programs/screens whenever possible. However, 4TEX still uses the MS-DOS COMMAND.COM substitute 4DOS(4DOS.COM). There are two reasons for this:

- 4TEX now uses the Windows Web2C TEX distribution (written by Fabrice Popineau). This is a 32-bits distribution with the possibility to use the TEX Directory Structure (TDS) but the executables still use a MS-DOS session to run. This immediately gives 4TEX the possibility to start 4DOS instead of MS-DOS and use the 4DOS batch command language to do extra checks, set some environment variables etc.
- When creating an open system, the user should be able to change/add programs. This isn't fully possible with the Delphi part of 4TEX (you need a lot of programming knowledge and the complete commercial Delphi environment). Using 4DOS still offers WINDOWS 95/NT users the same possibilities as with the (old) MS-DOS version. 4DOS comes with a complete help menu and digging through this and looking at the examples 4TEX supplies, you can learn to do advanced programming within 4DOS.

In the following sections we will specify and explain some features of 4TEX for windows.

2 The main menu

In figure 1 the main menu of 4TEX is shown. You can simply choose the TEX file you want to create/edit. 4DOS saves for each TEX file the 4TEX options you have selected (i.e. printer options, previewer options, TEX format,spellchecker language, bibTEX file etc.). When selecting a main TEX file these options are loaded (from the file with the file extension .PAR).

The first example of how 4TEX makes it possible to add/change the functionality of the menu without any programming is when selecting a TEX format. Pressing the "choose format" button a menu appears as shown in figure 2. This menu is the result of a file called US_FRM.LST. This .LST file contains plain ASCII text. Two lines in the file define a format, i.e. the first line is the title that is displayed in the format selection menu, the second line is the program (with options) that is executed when pressing the "compile" button. So this means if you want to add/change formats you only need to change the US_FRM.LST file.

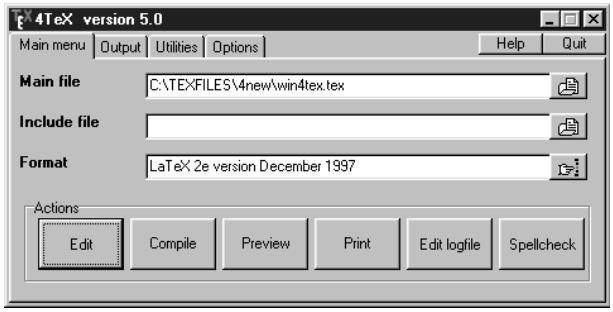


Figure 1. The 4TEX main menu

The method explained for format files also works for the user utilities menu, the select printer menu, the select previewer menu etc. The second way to configure 4TEX to your own taste is by changing the 4TEX.INI file. In this ini-file all path settings, program preferences (e.g. the editor)

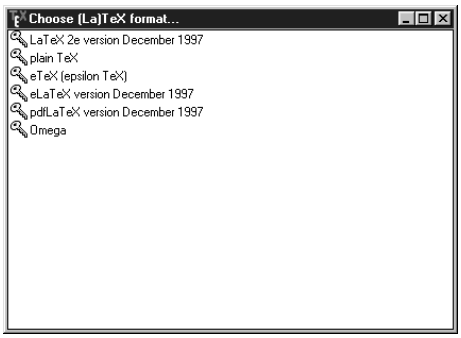


Figure 2. The 4TEX format menu

etc. can be changed.

4TEX tries to be multilingual, i.e. in the options menu (or when installing 4TEX) you can switch to another language. Switching to another language will result in a 4TEX menu with text/buttons etc. in the selected language. At the moment we support: English, German, and Dutch. We are working on: Polish, French, Russian,...

Of course the main menu has a lot of nice windows tricks like: clicking with the right-mouse button on the "select main TEX" file will result in a popdown list of the last 10 selected files (makes it possible to quickly get recently edited files). Under windows there is of course a windows help file with hypertext facilities. You can only open one-instance of 4TEX (i.e. you can not have multiple copies of 4TEXrunning) etc.

3 Editor

The editor that is by default used by 4TEX is the PFE editor (written by Alan Phillips). This editor is freeware and has all the functionality you need from an editor (except perhaps syntax coloring..). When starting the editor, also the program LATEX macros (written by Juan Aguirregabiria) is started. This LATEX macros tool makes is easy to insert LATEX commands, text commands, Greek letters, mathematics etc. by simply pressing a button. Figure 3 shows you an example of the PFE editor in combination with the macro tool.

The PFE editor has extensive macro possibilities and has the possibility to use Dynamic Data Exchange (DDE). This DDE feature is used when pressing the "compile" button in 4TEX. By pressing this button a DDE command is issued to save the file in the PFE editor, and hence you do not have to save the file before you compile it. Also the DDE is used for block compilation, i.e. marking a block in your text and by right-clicking the "compile" button, only the marked block is compiled and previewed. This block-compilation makes it easy to check complex mathematics and tables without compiling the whole document. When

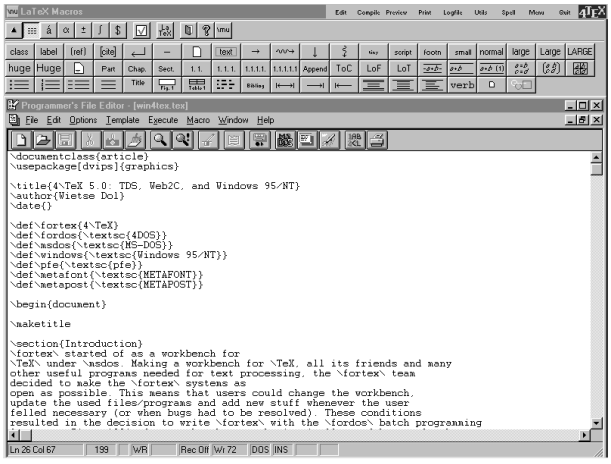


Figure 3. The PFE editor in combination with the LATEX macro tool

an error occurs during compilation you can jump to the line of the error by typing the 'e' and pressing the "enter".

Of course you can change the complete functionality of the editor and the LATEX macro tool by changing the 4TEX.INI file.

4 Printers and previewers

At the moment the following previewers are supported:

- XDVIWIN or WINDVI by Fabrice Popineau. Complete functionality of Web2C distribution, still under development but useful to view DVI files.
- DVIWIN 2.9 by H. Sendoucas. Virtual fonts are automatically replaced and automatic font generation is supported under 4TEX. Viewer has nice `\special` commands to include graphics but is not well suited for TDS structure and VF fonts.
- GSview by Russel Lang. Ghostscript (GNU-ware) in combination with GSview makes it possible to view/print PostScript and PDF files under windows. An extremely useful tool and the best viewer/printer approach when your TEX document has graphics (EPS as well as bitmaps).When this previewer is selected 4TEX will automatically generate a .PS or .PDF file from the .DVI file.

4TEX supports many printers (specified in the file US_PRN.LST). The best functionality/printer support is for PS printers and Ghostscript printing. For all printers the 4TEX printer options screen can be used to easily set all kind of printer options/parameters (e.g. page range, printer resolution, paper size etc.).

5 Utilities

The utilities/tools for TEX like bibTEX, MAKEINDEX, METAFONT, and METAPOST are all handled by windows screens. This makes entering the parameters and running the programs easy.

Under MS-DOS 4TEX had some useful tools like “automatic format generation” and “conversion tools”. Of course this functionality also can be found in 4DOS under WINDOWS 95/NT. 4TEX uses windows screens and can be extended by updating some .LST files or by adding some extra 4DOS batch files to a “utility” directory.

6 Conclusion

Our years of experience with TEX under MS-DOS makes it possible to write/create a system under WINDOWS 95/NT with the same functionality (and more...). Together with you we can extend this functionality even further and offer experienced as well as new TEXies a tool/workbench for making TEX-documents.

Bijlage 34

Perl and TEX a simple application

Gilbert van den Dobbelsteen
gilbert@login.iaf.nl

abstract

A simple application where perl is used to extract data from log-files and create output using TEX. The perl-scripts in this article run under perl 4.036 and should also run under perl 5.

1 Introduction

I am a software engineer programming embedded systems. Embedded systems are found everywhere in your environment, From a programmable calculator to dish-washers, coffee-machines, electric razors and GSM telephones.

Most of the time our projects involve communications with another system, usually a PC. Communication software usually involves one or more large state-machines which should ensure reliable transmission and reception. But since it's still software an implementation could be faulty. Timers could be to tight or retransmissions occur to frequently with no cause or the checksum algorithm fails on some weird occasion.

Since communications (and real-time error-free communications in specific) are difficult to debug we use log files where usually the raw data on the link is logged. The log-files are kept simple and straight and usually they are hard to read for normal humans.

Last year I engineered a project for Douwe Egberts Coffee Systems which involved communications with several other parties. The protocol used was a standard protocol used by many parties in the vending industry. The protocol is used to configure the vending machine and to read out the transactions made. Our company made a data-logger and some other party included similar functionality inside their vending machine.

Both parties made a simple PC for verifying the communications protocol. The specifications were not clear on all points so sometimes we needed to make choices in the implementation. After we both finished our product and our tools, we exchanged software and hardware to cross-verify each-others efforts.

That's where the trouble started. Their tool couldn't talk with our product, and vice-versa. So what to do? The log-files where extensive and I didn't had the time to spend hours explaining to the other party what was going on. So I decided to create a perl script which extracted vital information from the log-file and generated a file that could easily be processed by TEX.

I had some experience with perl and text-processing so this would be reasonably easy to do. It seems to me that perl is very well suited for data processing and TEX is a decent tool to create visual output.

2 Log-files

Almost all the programs we develop generate log-files. These are handy when a client calls with some problem. If it can't be supported by phone I ask them to email the log-file. Examining the log-files usually unfolds a bug in the software or a bug in somebody else's software.

Since most applications deal with communications I always have trouble examining the log-files. Sometimes the information is so large it is possible to overlook an error.

Here's an example of the data from the log-file:

```
> 04 05 01 56 A8 FA
< C4 67 13 62 E3 CC 00 00 2A 3F
```

The angular brackets indicate the direction of the transfer. Examining this data is not easy because most log-files are thousands of lines in size.

Perl is not only an excellent tool for processing this type of data, it can do more. It can actually analyze the protocol and point out some positions where the data is not right. This is very handy if the log-file is a few megabytes in size. You could even use colored output (in red?) for pin-pointing potential errors.

3 Basic protocol stuff

Skip this section if you're not interested, it contains some background information.

So what's a communication protocol? How do you transfer data from one system to antoher and make sure no errors occur? Many books have been written on this subject, and many examples are available. I'll stick with some simple things in a point-to-point[1] connection.

First of all you need to detect if the data you receive is correct. The easiest way to do this is to add redundant information. We always use cyclic redundancy checks. This

1. point-to-point: In this article a communication link between two computers.

is a reliable method of determining if received data is correct. Incorrect data is simply discarded.

Then comes the real trouble: When you send data, you want to make sure your data arrives at the other side. How to do that? Although this is easy, many machines I've seen have an incorrect implementation. The solution is to send an acknowledge message back, informing the sender the data arrived OK. The sender then sends the next available data.

Several things can go wrong here. The acknowledgement could get lost due to some error on the link (for example you manually disconnected the modem). The sender is waiting for an acknowledgement and when it does not arrive within a certain amount of time it simply re-transmits the data. This can go wrong too. Suppose you have a very slow link[2]. The acknowledgement arrives, but it is simply too late. The sender retransmits the data and you end up with the same data twice. When the data contains a command such as *turn on the coffee machine* this is no problem. The coffee machine is turned on while it was already on. But if the data contains a command like *pour a can of coffee* you end up with two cans of coffee when in fact you wanted only one. You can imagine what mess this gives since the person requesting a can of coffee expected only one.

How to solve that? Again this is simple. And it is the last thing I'll tell you about protocols. Make sure each data packet carries a unique number. When you re-transmit the data it contains that same number. The acknowledments from the receiver also return the number of the data acknowledged. So now the sender sends *1: pour a can of coffee*, the receiver replies: *ack:1* and pours a can of coffee. If the acknowledgement is lost, the sender re-transmits: *1: pour a can of coffee*. The receiver *knows* it has already seen data #1 so it simply sends back *ack:1* and does *not* pour another can of coffee. Is this simple or what? The unique numbers can easily be generated by a counter, which is incremented each time new data is sent. The receiver also has a counter so it knows which number should be the next to receive. Data arriving with another number is simply acknowledged and discarded.

So that is a simple way to create error-free communications. There are several other aspects but I won't bother you with the details.

4 Processing log-files

See the PERL script below for processing the data.

```
while($line = <STDIN>) {
  if($line =~ /^\>.*/) {
    &process_right(substr($line, 1));
  }
  elsif($line =~ /^\<.*/) {
    &process_left(substr($line, 1));
  }
}
```

The above script processes input from standard input and calls the function `process_right` if the line starts with a > or `process_left` if the line starts with <. Before the call is made, it strips of the first character using `substr` (similar to basic's `MID$`). Don't be alarmed by amount of rubbish present. The $ sign indicates a simple string variable, the & indicates a function call. The forward slashes indicate a regular expression. The rest looks like a C program.

So what to do now? I'll give the details of the `process_right` function, the `process_left` is omitted since it is similar:

```
sub process_right{
  local($data) = @_;
  local(@values);
  local($local_ns,$local_nr);

  $data =~ s/^\s+//; # omit leading white space

  @values = split(/[\s\n]+/, $data);

  # @values is an array with the seperate hexa-
  # decimal values. Convert them to decimal.

  foreach $value (@values){
    $value = hex($value);
  }

  if($values[0] == 1){     # START command?
    print "\\r {START}\n";
  }elsif($values[0] == 2){ # ACK   command?
    $local_nr = $values[1];
    print "\\r {ACK $local_nr}\n";
  }elsif($values[0] == 3){ # NACK  command?
    $local_nr = $values[1];
    print "\\r {NACK $local_ns}\n";
  }elsif($values[0] == 4){ # DATA
    $local_ns = $values[1];
    print "\\r {DATA $local_ns}\n";
  }else{                   # Unknown command
    print "\\r {UNKNOWN}\n";
    &print_data("\\rt", @hexvalues);
  }
}
```

I omitted the various checking functions here which are present in the actual perl-script. This is just to illustrate how to do such things. First the leading white-space is stripped with a regular expression. Then `$data` is converted into

2. I know you use internet so you know what I'm talking about.

an array of `values`. Arrays in perl are arrays of scalars (variables that start with a $-sign). Scalars can be strings or numbers. When you apply an operation, perl automagically converts them to the correct type.

So now we are left with an array of hexadecimal values. This is not too bad, perl provides the `hex` gunction for conversion. The `foreach` statement converts the values to decimal.

After that the program finds out what the command is and prints a converted line on standard out. That's all there is to it. Note: you can use variables directly in print strings, perl expands the variables for you.

5 Output

The output of the script contains lines like this:

```
\STARTPROTOCOL
\l {START}
\lt{Start of protocol}
\r {STACK}
\rt{Ackowledge}
\STOPPROTOCOL
```

So the arrow drawing stuff is left to TEX. These are simple macro's:

```
\startTEX
\def\BOXSIZE{\hsize}
\def\STARTPROTOCOL{\bgroup
  \def\l##1{\hbox to \BOXSIZE{\leftarrowfill
                              \ {\tt ##1}}}
  \def\r##1{\hbox to \BOXSIZE{{\tt ##1}\
                              \rightarrowfill}}
  \def\lt##1{\hbox to \BOXSIZE{\hfill ##1}}
  \def\rt##1{\hbox to \BOXSIZE{##1\hfill}}
  \obeylines }
\def\STOPPROTOCOL{\egroup}
```

6 Example

Below is an example of a complete and correct communication session:

RECEIVER *SENDER*

← START

Communication start

STACK →

Acknowledgement

← WhoAreYou

Sec: 0x2342, Pass: 0x7326

WhoAreYou Accepted →

Serial#: 385682445

← FINISH

ACK →

The `WhoAreYou` command identifies the terminal, and after it is accepted configuration commands can be given.

Here's an extensive example with several errors in it:

RECEIVER *SENDER*

← START

Communication start

STACK →

Start Acknowledged

← START

Communication start (**duplicate!**)

STACK →

Start Acknowledged (**duplicate!**)

← DATA 1

WhoAreYou

← DATA 1

WhoAreYou (**retry!**)

ACK 1 →

DATA 1 →

WhoAreYou Accepted)

← ACK 1

← DATA 2

Finish

NACK 1 →

(Finish) not received!

← DATA 2

Finish

ACK 2 →

As you can see the tool finds errors which are hard to spot by humans. The `STACK` response is not received by the sender so the `START` command is sent again. The `WhoAreYou` command is not received by the receiver so the sender sends a retry. Near the end, the `Finish` command was received but it contained an error. This is signalled by the `NACK`. The sender retries the operation.

7 Conclusion

Perl is a good tool to process ASCII data. When used in combination with TEX, the tool can create nice output. The power of perl is completely unlike C. Although it looks a lot like C it has much more power. In specific the string processing is supurb. For almost anything you want there is a perl internal function available. And if there's not you can roll your own.

I didn't discuss perl 5 which adds many more features and object oriented programming. Perhaps I'll discuss that the next time. Perl is also well equipped for systems programming. It is possible to write a complete web-server in perl.

The program described in this article helped us locating bugs quick and easy. It's a convenient way to look at data. Since the original specifications included similar diagrams, verifying the diagrams was easy too.

Bijlage 35
Some funny macro's

Hans Hagen
pragma@pi.net

Keywords
dropped caps, CONTEXT

abstract
Sometimes documents can be enhanced with special typography for first characters or lines of chapters. In this article I present some macros for dropped caps and first line treatment. Although more advanced solutions are possible, the examples show at least how things work. Users can derive their own macros from them.

```
1  \unprotect
```

This module implements some typographics tricks that can be fun when designing document layouts. The examples use macros that are typical to CONTEXT, but non CONTEXT users can use the drop caps and first line treatment macros without problems. This module will be extended when the need for more of such tricks arises.

```
2  \ifx \undefined \writestatus \input supp-mis.tex \relax \fi
3  \writestatus{loading}{Context Support Macros / Fun Stuff}
```

Let's start with dropped caps, those blown up first characters of a paragraph. It's hard to implement a general mechanism that suits all situations, but dropped caps are so seldomly used that we can permit ourselves a rather user unfriendly implementation. \DroppedCaps

```
\DroppedCaps
  {\color[green]} {cmbx12}
  {2.2\baselineskip} {2pt} {\baselineskip} {2}
  Let's start
```

As we will see, there are 7 different settings involved. The first argument takes a command that is used to do whatever fancy things we want to do, but normally this one will be empty. The second argument takes the font. Because we're dealing with something very typographic, there is no real reason to adopt complicated font switching schemes, a mere name will do. Font encodings can bring no harm, because the alphanumeric characters are nearly always located at their natural position in the encoding vector.

This simple case shows us what happens when we apply minimal values. Here we used:

```
\DroppedCaps
  {\color[red]} {cmbx12}
  {\baselineskip} {0pt} {0pt} {1}
  This simple
```

In this ugly example the third argument tells this macro that we want a dropped capital scaled to the baseline distance. The two zero point arguments are the horizontal and vertical offsets and the last arguments determines the hanging indentation. In this paragraph we set the height to two times the baselinedistance and use two hanging lines:

```
\DroppedCaps
  {\color[red]} {cmbx12}
  {2\baselineskip} {0pt} {\baselineskip} {2}
  In this ugly
```

Here, the first character is moved down one baseline. Here we also see why the horizontal offset is important. The first example (showing the L) sets this to a few points and also used a slightly larger height.

Of course common users (typist) are not supposed to see this kind of fuzzy definitions, but fortunately TEX permits us to hide them in macros. Using a macro also enables us to garantee consistency throughout the document:

```
\def\MyDroppedCaps%
  {\DroppedCaps
     {\color[green]} {cmbx12}
     {5\baselineskip} {3pt} {3\baselineskip} {4}}

\MyDroppedCaps The implementation
```

The implementation of the general macro is rather simple and only depends on the arguments given and the dimensions of the strut box. We explicitly load the font, which is no problem because TEX does not load a font twice. We could have combined some arguments, like the height, vertical offset and the number of lines, but the current implementation proved to be the most flexible. One should be aware of the fact that the offsets depend on the design of the glyphs used.

4

```
\def\DroppedCaps#1#2#3#4#5#6#7%
  {\par
   \vskip#6\baselineskip
   \penalty-200
   \vskip-#6\baselineskip
   \setbox0=\hbox
     {\font\temp=#2 at #3%
      \temp#1{#7}\hskip#4}%
   \setbox0=\hbox
     {\lower#5\box0}%
   \ht0=\ht\strutbox
   \dp0=\dp\strutbox
   \hangindent\wd0
   \hangafter-#6%
   \noindent
   \hskip-\wd0
   \vbox{\forgetall\box0}%
   \nobreak}
```

Before we go to the next topic, we summarize this command:

```
\DroppedCaps
   {command} {font}
   {height} {hoffset} {voffset} {lines}
```

\TreatFirstLine

INSTEAD OF LIMITING ITS ACTION TO ONE TOKEN, THE NEXT MACRO TREATS THE whole first line. This paragraph was typeset by saying:

```
\TreatFirstLine {\sc} {} {} {}
Instead of limiting its action to one token, the next macro
treats the whole first line. This paragraph was typeset by
saying:
```

The combined color and font effect is also possible, although one must be careful in using macros that accumulate grouping, but the commands used here are pretty save in that respect.

```
\TreatFirstLine {\startcolor[red]\bf} {\stopcolor} {} {}
The combined color and font effect is also possible,
```

```
although one must be careful in using macros that accumulate
grouping, but the commands used here are pretty save in that
respect.
```

Before we explain the third and fourth argument, we show the implementation. Those who know a bit about the way TEX treats tokens, will probably see in one glance that this alternative works all right for most text–only situations in which there is enough text available for the first line, but that more complicated things will blow. One has to live with that. A workaround is rather trivial but obscures the principles used.

```
\def\TreatFirstLine#1#2#3#4% before, after, first, next
  {\leavevmode
   \bgroup
   \forgetall
   \bgroup
   #1%
   \setbox0=\box\voidb@x
   \setbox2=\box\voidb@x
   \def\grabfirstline##1 %
     {\setbox2=\hbox
        {\ifvoid0
           {#3{\ignorespaces##1}}%
         \else
           \unhcopy0\ {#4{##1}}%
         \fi}%
      \ifdim\wd2=\!!zeropoint
        \setbox0=\box\voidb@x
        \setbox2=\box\voidb@x
        \let\next=\grabfirstline
      \else\ifdim\wd2>\hsize
        \hbox to \hsize{\strut\unhbox0}#2\egroup
        \break##1\
        \egroup
        \let\next=\relax
      \else
        \setbox0=\box2
        \let\next=\grabfirstline
      \fi\fi
      \next}%
   \grabfirstline}
```

individual words. Of course one needs ... to know a bit more about the macro package used to get real nice effects, but this example probably demonstrates the principles well.

```
\gdef\FunnyCommand
  {\getrandomfloat\FunnyR{0}{1}%
   \getrandomfloat\FunnyG{0}{1}%
   \getrandomfloat\FunnyB{0}{1}%
   \definecolor[FunnyColor][r=\FunnyR,g=\FunnyG,b=\FunnyB]%
   \color[FunnyColor]}

\TreatFirstLine {\bf} {} {\FunnyCommand} {\FunnyCommand}
The third and fourth argument can be used to gain special
effects on the individual words. Of course one needs ...
```

Like in dropped caps case, one can hide such treatments in a macro, like:

```
\def\MyTreatFirstLine%
  {\TreatFirstLine{\bf}{}{\FunnyCommand}{\FunnyCommand}}
```

```
\beginofshapebox
When using \CONTEXT, one can also apply this funny command
to whole lines by using the reshape mechanism. Describing
this interesting mechanism falls outside the scope of this
module, so we only show the trick. This is an example of
low level \CONTEXT\ functionality: it's all there, and it's
stable, but not entirely meant for novice users.
\endofshapebox

\reshapebox{\FunnyCommand{\box\shapebox}} \flushshapebox
```

This mechanism permits hyphenation and therefore gives better results than the previously discussed macro `\TreatFirstLine`.

`\TreatFirstCharacter`

Just to be complete we also offer a very simple one character alternative, that is not that hard to understand:

6

```
\def\TreatFirstCharacter#1#2% command, character
  {{#1{#2}}}
```

A previous paragraph started with:

```
\TreatFirstCharacter{\bf\color[green]} Just to be
```

`\StackCharacters`

The next hack deals with vertical stacking.

7

```
\def\StackCharacters#1#2#3#4% sequence vsize vskip command
  {\vbox #2
     {\forgetall
      \baselineskip0pt
      \def\StackCharacter##1{#4{##1}\cr\noalign{#3}}%
      \halign
        {\hss##\hss&##\cr
         \handletokens#1\with\StackCharacter\cr}}}
```

Such a stack looks like:

and is typeset by saying:

```
\StackCharacters{CONTEXT}{}{\vskip.2ex}{\FunnyCommand}
```

An alternative would have been

```
\StackCharacters {CONTEXT} {to 5cm} {\vfill} {\FunnyCommand}
```

At a lower level horizontal and vertical manipulations are already supported by: \processtokens

```
\processtokens {begin} {between} {end} {space} {text}
```

This macro is able to typeset:

L E T ' S H A V E F U N

which was specified as:

```
\processtokens
  {\hbox to .5\hsize\bgroup} {\hfill}
  {\egroup} {\space} {LET'S HAVE}

\processtokens
  {\vbox\bgroup\raggedcenter\hsize1em}
  {\vskip.25ex} {\egroup} {\strut} {FUN}
```

Next we introduce some font manipulation macros. When we want to typeset some text spread in a well defined area, it can be considered bad practice to manipulate character and word spacing. In such situations the next few macros can be of help: \NormalizeFontHeight \NormalizeFontWidth

```
\NormalizeFontHeight \name {sample text} {height} {font}
\NormalizeFontWidth  \name {sample text} {width}  {font}
```

These are implemented using an auxilliary macro:

```
8  \def\NormalizeFontHeight%
     {\NormalizeFontSize\ht}
9  \def\NormalizeFontWidth%
     {\NormalizeFontSize\wd}
10 \def\NormalizeFontSize#1#2#3#4#5%
     {\setbox0=\hbox{\font\temp=#5 at 10pt\temp#3}%
      \dimen0=#10
      \dimen2=10000pt
      \divide\dimen2 by \dimen0
      \dimen4=#4%
      \divide\dimen4 by 1000
      \dimen4=\number\dimen2\dimen4
      \edef\NormalizedFontSize{\the\dimen4}%
      \font#2=#5 at \NormalizedFontSize}
```

Consider for instance:

```
\NormalizeFontHeight \tmp {X} {2\baselineskip} {cmr10}
{\tmp To Be Or Not To Be}
```

This shows up as (we also show the baselines):

1 To Be Or Not To Be

The horizontal counterpart is:

```
\NormalizeFontWidth \tmp {This Line Fits} {\hsize} {cmr10}
\hbox{\tmp This Line Fits}
```

The calculated font scale is avaliable in the macro \NormalizedFontSize.

One can of course combine these macros with the ones described earlier, like in:

```
\NormalizeFontHeight \DroppedFont {2\baselineskip} {cmbx12}

\def\NicelyDroppedCaps%
  {\DroppedCaps
     {\kleur[groen]}
     {\DroppedFont}
     {2pt}
     {\baselineskip}
     {2}}
```

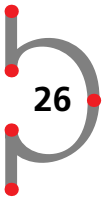
It's up to the reader to test this one.

```
\protect
\endinput
```

Bijlage 36

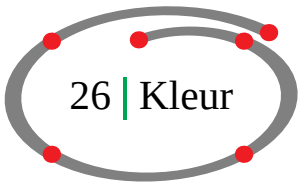
CONTEXT
een rondleiding

Ton Otten
Hans Hagen
PRAGMA ADE
Ridderstraat 27
8061GH Hasselt
ntg-context@ntg.nl



abstract
This is the second half of the CONTEXT manual for beginners. Those who want an index and a quick reference guide, can download the complete manual from the NTG server. The layout is adapted to the MAPS layout.

Keywords
CONTEXT



Teksten kunnen in kleur worden gezet met:

```
\kleur[...]
```

Het gebruik van kleuren wordt geactiveerd door:

```
\stelkleurenin[status=start]
```

Vanaf dat moment zijn de basiskleuren beschikbaar. Basiskleuren zijn rood, groen en blauw.

```
\startkleur[rood]
Hasselt is een \kleur[groen]{kleurrijke} stad.
\stopkleur
```

Hasselt is een kleurrijke stad.

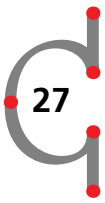
Op een zwart–wit printer ziet u alleen maar grijswaarden. In een elektronisch document verschijnen de kleuren zoals bedoeld.

Het is ook mogelijk uw eigen kleuren te definiëren met:

```
\definieerkleur[...][..,..=..,..]
```

Bijvoorbeeld:

```
\definieerkleur[donkerrood]   [r=.5,g=.0,b=.0]
\definieerkleur[donkergroen]  [r=.0,g=.5,b=.0]
```



Na de definitie zijn de kleuren donkerrood en donkergroen beschikbaar als de commando's `\donkerrood` en `\donkergroen`.

27 | Achtergronden bij tekst

Een alinea of paragraaf kan worden benadrukt met bijvoorbeeld een achtergrond. Een achtergrond wordt aangemaakt met het commando–paar:

```
\startachtergrond ... \stopachtergrond
```

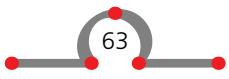
Een voorbeeld kan het gebruik toelichten:

```
\stelachtergrondin[achtergrond=raster,hoek=rond]

\startachtergrond
Hasselt heeft een aantal prominente mensen voortgebracht.
Recent is aan het licht gekomen dat Kilian van Rensselaer
een belangrijke rol heeft gespeeld bij de oprichting van de
staat New York.
\stopachtergrond
```

Dit wordt weergegeven als:

Hasselt heeft een aantal prominente mensen voortgebracht. Recent is aan het licht gekomen dat Kilian van Rensselaer een belangrijke rol heeft gespeeld bij de oprichting van de staat New York.

Een achtergrond kan over meerdere pagina's doorlopen. Met:



```
\stelachtergrondin[..,..=..,..]
```

kunnen de instellingen van achtergronden worden gewijzigd.

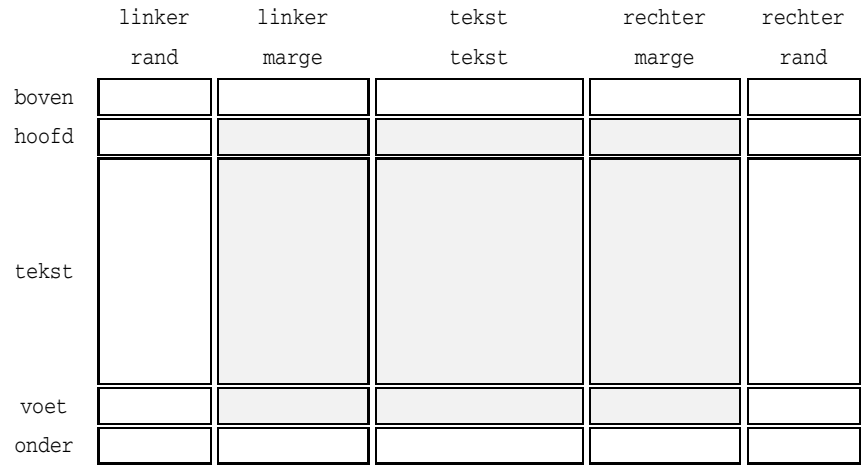
28 | Achtergronden op paginavlakken

28

De achtergrond van ieder paginavlak kan worden ingesteld. Het commando daarvoor is:

```
\stelachtergrondenin[.1.][..,.2.,..][..,..=..,..]
```

De eerste twee paren haken worden gebruikt om de paginavlakken te definiëren. Het laatste paar wordt gebruikt om de instellingen vast te leggen.



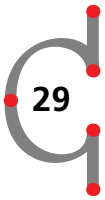
Figuur 28.1 De paginavlakken ingesteld met `\stelachtergrondenin`.

Indien u de achtergronden in de grijze gebieden van figuur 28.1 wilt wijzigen, typt u:

```
\stelachtergrondenin
  [hoofd,tekst,voet]
  [linkermarge,tekst,rechtermarge]
```

```
[achtergrond=raster]
```

29 | Uitlijnen



Horizontaal en verticaal uitlijnen wordt ingesteld met:

```
\steluitlijnenin[...]
```

Afzonderlijke regels kunnen worden uitgelijnd met:

```
\regelrechts{}
\regellinks{}
\regelmidden{}

\regellinks{Hasselt is gebouwd op een zandheuvel.}
\regelmidden{Hasselt ligt aan een kruising van twee rivieren.}
\regelrechts{Hasselt is vernoemd naar een hazelaar.}
```

Na het verwerken ziet dit er als volgt uit:

Hasselt is gebouwd op een zandheuvel.
Hasselt ligt aan een kruising van twee rivieren.
Hasselt is vernoemd naar een hazelaar.

Uitlijnen van paragrafen worden gedaan met het commando–paar:

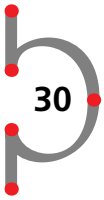
```
\startuitlijnen ... \stopuitlijnen[...]
```

Bij het uitlijnen kan een tolerantie en de richting (verticaal of horizontaal) worden ingesteld. Normaal is de tolerantie `zeerstreng`. In kolommen kan het uitlijnen soepeler worden ingesteld `zeersoepel`. De uitlijntolerantie in deze handleiding is als volgt ingesteld:

```
\steltolerantiein[horizontaal,zeer streng]
```

30 | Interactiviteit in elektronische documenten

30.1 Inleiding



Documenten kunnen elektronisch worden uitgegeven, zodat ze op een computer kunnen worden geraadpleegd en op een scherm kunnen worden weergegeven.

Interactiviteit betekent dat specifieke gebieden in het document actief (hyperlinks) zijn gemaakt. Dit houdt in dat die gebieden (meestal met de muis) kunnen worden geselecteerd en aangeklikt. Het aanklikken resulteert in een sprong naar het aangewezen doelgebied. Bij het raadplegen van een index kan bijvoorbeeld op een ingang worden geklikt, waarna naar de corresponderende pagina wordt gesprongen.

Interactie heeft betrekking op:

- actieve hoofdstuknummers in de inhoudsopgave
- actieve paginanummers in registers
- actieve paginanummers, hoofdstuknummers en figuurnummers in verwijzingen naar pagina's, hoofdstukken, figuren enz.
- actieve titels, paginanummer, hoofdstuknummers in externe verwijzingen naar andere interactieve documenten
- actieve menu's ten behoeve van navigatie hulpmiddelen

De interactiviteit hangt af van het programma dat wordt gebruikt voor het bekijken van het document. In deze handleiding wordt ervan uitgegaan dat u Acrobat Distiller gebruikt voor het produceren van PDF–documenten vanuit PS–files. Vervolgens kunt u die documenten bekijken of raadplegen met Acrobat Reader of Acrobat Exchange.

CONTEXT is een zeer goed hulpmiddel voor de produktie van elektronische of interactieve PDF–documenten. In deze handleiding wordt slechts een klein deel van de functionaliteit besproken. De auteurs hebben echter besloten alle documenten rond CONTEXT tevens elektronisch beschikbaar te maken, zodat u inzicht heeft in de mogelijkheden van CONTEXT.

30.2 Interactie

Interactiviteit wordt geactiveerd door:

```
\stelinteractiein[..,..=..,..]
```

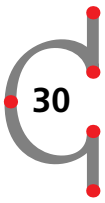
Bijvoorbeeld:

```
\stelinteractiein
```

```
[status=start,
 kleur=groen,
 letter=vet]
```

De zogenaamde hyperlinks worden nu automatisch gegenereerd en actieve woorden worden vetgroen weergegeven.

Het interactieve document is aanzienlijk groter (in MegaBytes) dan zijn papieren tegenhanger, omdat hyperlinks ruimte in beslag nemen. Ook de verwerkingstijd van een document neemt toe. Het is daarom verstandig de interactie pas te activeren als het document zijn eindstadium heeft bereikt.



30.3 Interactie binnen een document

In hoofdstuk 25 heeft u gezien dat u verwijzingen kunt aanmaken met `\in` en `\op`. U zult zich wellicht hebben afgevraagd waarom u ook *hoofdstuk* moest intypen bij een verwijzing als `\in{hoofdstuk}[introductie]`. In de eerste plaats worden *hoofdstuk* en het corresponderende nummer niet van elkaar gescheiden bij regelovergangen. In de tweede plaats worden zowel het woord *hoofdstuk* als het hoofdstuknummer in de interactieve toestand afwijkend gezet (meestal vet groen) en worden beide aanklikbaar. Hierdoor kan de gebruiker makkelijker een doelgebied selecteren.

Er is een commando dat alleen betekenis heeft in een interactief document.

```
\naar{.1.}{.2.}[ref]
```

De accolades bevatten tekst en de haken omsluiten de verwijzing.

```
In \naar{Hasselt}[fig:cityplan] zijn de straten
cirkelvormig aangelegd.
```

In het interactieve document is *Hasselt* groen en actief. Er wordt een sprong gerealiseerd naar een kaart van Hasselt.

30.4 Interactie tussen documenten

Het is mogelijk om van en naar meerdere documenten te springen. Allereerst dient u het document te definiëren waarnaar u wilt verwijzen.

```
\gebruikexterndocument[.1.][.2.][.3.]
```

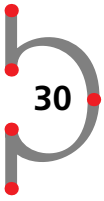
De eerste haken bevatten een logische naam voor het externe document, het tweede paar de filenaam zonder extensie en het derde paar wordt gebruikt voor een titel van het document.

Vervolgens kunt u refereren naar het externe document met:

```
\uit{...}[ref]
```

De accolades bevatten tekst en de vierkante haken de verwijzing. Hierna volgt een voorbeeld.

```
\gebruikexterndocument[hia][hasboek][Hasselt in augustus]

De meeste toeristische attracties worden beschreven in
\uit[hia]. Een beschrijving van het Eui||feest wordt gegeven
in \uit[hia::euifeest]. Een beschrijving van het
\naar{Eui--feest}[hia::euifeest] vindt u in \uit[hia]. Het
eui||feest is beschreven op \op{pagina}[hia::euifeest] in
\uit[hia]. Zie voor meer informatie
\in{hoofdstuk}[hia::euifeest] in \uit[hia].
```



Het commando `\gebruikexterndocument` wordt meestal in het instelgebied van de invoerfile gedefinieerd.

De dubbele `::` geven aan dat het gaat om een referentie naar een extern document.

Na het verwerken van uw invoerfile en de file `hasboek.tex` (allebei ten minste twee maal ten behoeve van de referenties) kunt u twee PDF–documenten aanmaken met Acrobat Distiller. De referenties hierboven hebben de volgende betekenis:

- `\uit[hia]` produceert een actieve titel die u in het derde hakenpaar van het commando `\gebruikexterndocument` heeft gedefinieerd en is gelinked (verwijst) naar de eerste pagina van `hasboek.pdf`
- `\uit[hia::euifeest]` produceert een actieve titel en is gelinked (verwijst) naar de pagina waar hoofdstuk *Eui–feest* begint
- `\naar{Eui--feest}[hia::euifeest]` produceert een actief woord *Eui–feest* en is gelinked (verwijst) naar de pagina waar hoofdstuk *Eui–feest* begint
- `\op{pagina}[hia::euifeest]` produceert een actief woord *pagina* en paginanummer en is gelinked (verwijst) naar die pagina
- `\in{hoofdstuk}[hia::euifeest]` produceert een actief woord *hoofdstuk* en hoofdstuknummer en is gelinked aan dat hoofdstuk

Zoals u ziet scheidt de `::` de (logische) filenaam en het doelgebied.

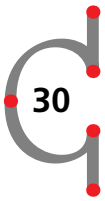
30.5 Menu's

U kunt navigatiehulpmiddelen definiëren met:

```
\definieerinteractiemenu[.1.][.2.][..,..=..,..]
```

De eerste haakjes zijn bedoeld voor een logische naam van het menu, waarmee het menu in een later stadium kan worden opgeroepen. Het tweede paar wordt gebruikt om de plaats op het scherm vast te leggen. Het derde paar bevat de instellingen.

Een menudefinitie kan er als volgt uitzien:




```
\stelkleurenin
  [status=start]

\stelinteractiein
  [status=start,
   menu=aan]

\definieerinteractiemenu
  [mijnmenu]
  [rechts]
  [status=start,
   uitlijnen=midden,
   achtergrond=raster,
   kader=aan,
   breedte=\margebreedte,
   letter=kleinvet,
   kleur=]

\stelinteractiemenuin
  [mijnmenu]
  [{Inhoud[inhoud]},
   {Index[index]},
   {\vfill},
   {Stoppen[VerlaatViewer]}]
```

Deze definitie produceert een menu aan de rechterkant van ieder scherm. De menuknoppen bevatten de teksten *Inhoud*, *Index* en *Stoppen* en hebben respectievelijk de volgende functies: een sprong naar de inhoudsopgave, een sprong naar de index en het verlaten van de viewer. De labels `inhoud` en `index` zijn voorgedefinieerd. Andere voorgedefinieerde locaties zijn `EerstePagina`, `LaatstePagina`, `VolgendePagina` en `VorigePagina`.

Een actie als `VerlaatViewer` is nodig om het elektronische document zo onafhankelijk mogelijk te maken van de viewer. Andere voorgedefinieerde acties zijn `VorigeSprong`, `DoorzoekDocument` en `PrintDocument`. De betekenis van deze acties spreken voor zich.

Menu's worden ingesteld met:

```
\stelinteractiemenuin[..,...,..][..,..=..,..][..,{..[ref]},..]
```

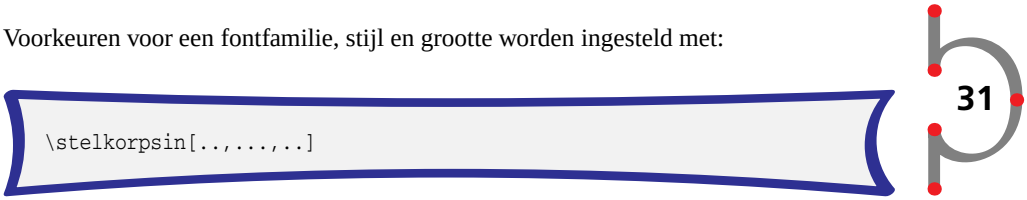
31 | Fonts en fontovergangen

31.1 Introductie

De standaard font in CONTEXT is *Computer Modern Roman* (cmr). Bovendien is Lucida Bright (lbr) een volwaardig alternatief en zijn symbolen van *American Society* (ams) beschikbaar. Verder kunnen PostScript fonts (pos) worden gebruikt.

31.2 Fontstijl en grootte

Voorkeuren voor een fontfamilie, stijl en grootte worden ingesteld met:



Indien u in het instelgebied typt `\stelkorpsin[sansserif,9pt]` komt de tekst in uw document er ongeveer zo uit te zien.

Veranderingen in de font op een willekeurige plaats in het document kunnen worden gedaan met:

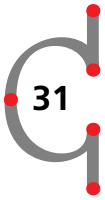

```
Op 10 november, een dag voor Sint Maarten, trekt de jeugd
van Hasselt van deur tot deur om een speciaal liedje te
zingen en zichzelf te begeleiden op de {\em foekepot}. Ze
gaan niet weg voordat ze wat geld of wat snoepgoed hebben
gekregen. Het liedje gaat als volgt:

\startsmaller
\switchnaarkorps[klein]
\startregels
Foekepotterij, foekepotterij,
Geef mij een centje dan ga'k voorbij.
Geef mij een alfje dan blijf ik staan,
'k Zak nog liever naar m'n arrenmoeder gaan.
Hier woont zo'n rieke man, die zo vulle gèven kan.
Gèf wat, old wat, gèf die arme stumpers wat,
'k Eb zo lange met de foekepot elopen.
'k Eb gien geld om brood te kopen.
Foekepotterij, foekepotterij,
Geef mij een centje dan ga'k voorbij.
```

```
\stopregels
\stopsmaller
```

Hierbij wordt opgemerkt dat `\startsmaller ··· \stopsmaller` ook het begin en het einde aangeven van de fontovergang. De functie van `\startregels` en `\stopregels` in dit voorbeeld spreekt voor zich.

Op 10 november, een dag voor Sint Maarten, trekt de jeugd van Hasselt van deur tot deur om een speciaal liedje te zingen en zichzelf te begeleiden op de *foekepot*. Ze gaan niet weg voordat ze wat geld of wat snoepgoed hebben gekregen. Het liedje gaat als volgt:

Foekepotterij, foekepotterij,
Geef mij een centje dan ga'k voorbij.
Geef mij een alfje dan blijf ik staan,
'k Zak nog liever naar m'n arrenmoeder gaan.
Hier woont zo'n rieke man, die zo vulle gèven kan.
Gèf wat, old wat, gèf die arme stumpers wat,
'k Eb zo lange met de foekepot elopen.
'k Eb gien geld om brood te kopen.
Foekepotterij, foekepotterij,
Geef mij een centje dan ga'k voorbij.



Indien u een overzicht wilt van de fontfamilie kunt u het volgende commando invoeren:

```
\toonkorps[cmr]
```

[cmr]													
	\tf	\sc	\sl	\it	\bf	\bs	\bi	\tfx	\tfxx	\tfa	\tfb	\tfc	\tfd
\rm	Ag	AG	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag
\ss	Ag	Ag	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag
\tt	Ag	Ag	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag

31.3 Fontstijl- en grootte–overgang in commando's

In enkele commando's kan men de `letter` instellen. Bijvoorbeeld:

```
\stelkopin[hoofdstuk][letter=\tfd]
```

In dit geval wordt de fontgrootte voor het zetten van de hoofdstukken aangegeven met het commando `\tfd`. In plaats van een dergelijk commando kunnen ook de volgende opties van het actuele font worden ingegeven:

```
normaal  vet  schuin  vetschuin  type  mediaeval
klein  kleinvet  kleinschuin  kleinvetschuin  kleintype
kapitaal kap
```

31.4 Locale fontstijl- en fontgrootte–overgang

In de tekst kunt u de stijl veranderen in roman, sans serif en teletype met `\rm`, `\ss` en `\tt`. De lettertypen italic en boldface worden veranderd met `\sl` en `\bf`. De grootte kan

variëren van 4pt tot 12pt en wordt veranderd met `\switchnaarkorps`.

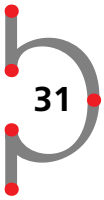
Het actuele font wordt steeds aangeduid met `\tf`. Indien u naar een grotere letter wilt overgaan, kunt u `\tfa`, `\tfb`, `\tfc` en `\tfd` typen. In aanvulling op `a`, `b`, `c` en `d` mag u ook `\sl`, `\it` en `\bf` gebruiken.

```
{\tfc Muntslag}

In de periode van {\tt 1404} tot {\tt 1585} had Hasselt een
eigen muntatelier en mocht het zelf munten slaan. Dit recht
werd door andere steden aangevochten, maar de
{\switchnaarkorps[7pt] bisschop van Utrecht} ging niet in op
deze {\slb protesten}.
```

de accolades geven het begin en eind van de fontovergangen aan.

Muntslag

In de periode van `1404` tot `1585` had Hasselt een eigen muntatelier en mocht het zelf munten slaan. Dit recht werd door andere steden aangevochten, maar de bisschop van Utrecht ging niet in op deze *protesten*.



31.5 Herdefiniëren fontgrootte

Voor speciale toepassingen kunt u de fontgrootte herdefiniëren.

```
\definieerkorps[..,.1.,..][.2.][..,..=..,..]
```

Een definitie kan er als volgt uitzien:

```
\definieerkorps[10pt][rm][tfe=lbr at 36pt]
{\tfe Hasselt!}
```

Vervolgens produceert `\tfe` de 36pt grote letters: ασσελτη

31.6 Klein kapitaal

Afkortingen als PDF (Portable Document Format) worden gezet in pseudo klein kapitaal. Een klein kapitaal is iets kleiner dan kapitaal van het actuele font. Pseudo klein kapitaal worden als volgt gemaakt:

```
\kap{}
```

Als u `PDF club`, `\kap{PDF club}` en `\sc PDF club` vergelijkt:

PDF club en PDF CLUB en PDF CLUB

dan ziet u de verschillen. Het commando `\sc` toont een 'echte' klein kapitaal. De reden

voor het gebruik van pseudo klein kapitaal heeft te maken met persoonlijke voorkeuren.

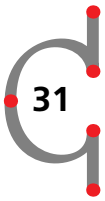
31.7 Benadrukken

Om consistent tekstfragmenten te kunnen benadrukken bestaat het commando:

```
\em
```

Benadrukte woorden worden schuin gezet.

```
Als u door Hasselt loopt, moet u uitkijken voor {\em
Amsterdammers}. Een {\em Amsterdammer} is {\bf \em geen}
inwoner van Amsterdam maar een kleine stenen pilaar die
wordt gebruikt om trottoir en straat te scheiden. Wandelaars
zouden door die {\em Amsterdammers} beschermd moeten
worden, maar heel vaak verwonden zij zich omdat ze over de
paaltjes struikelen.
```



Dit wordt: Als u door Hasselt loopt, moet u uitkijken voor *Amsterdammers*. Een *Amsterdammer* is ***geen*** inwoner van Amsterdam maar een kleine stenen pilaar die wordt gebruikt om trottoir en straat te scheiden. Wandelaars zouden door die *Amsterdammers* beschermd moeten worden, maar heel vaak verwonden zij zich omdat ze over de paaltjes struikelen.

Een benadrukt woord binnen een benadrukte zin wordt weer normaal *gedrukt en vet benadrukken zou er* **als volgt uit moeten *zien***.

31.8 Typeletters / verbatim

Indien tekst in een typeletter moet worden weergegeven, gebruikt u:

```
\starttypen ... \stoptypen
```

In een tekst typt u:

```
\type{...}
```

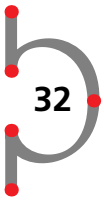
De accolades omsluiten de tekst die in een typeletter moet worden weergegeven. Een waarschuwing is op zijn plaats. Bij het werken met `\type` moeten regelovergangen extra worden gecontroleerd, omdat het afbreekmechanisme niet werkt.

U kunt met betrekking tot typen het een en ander instellen met:

```
\steltypenin[...][..,..=..,..]
```

```
\steltypein[..=..]
```

32 | Samengestelde karakters



32.1 Inleiding

In hoofdstuk 3 heeft u gezien dat u voor speciale karakters meer dan een karakter moet intypen. Dit geldt voor # $ % & _ { en }. Karakters met accenten moeten worden samengesteld om uiteindelijk het juiste karakter te krijgen.

Ook in de mathematische mode bestaan geaccentueerde karakters. Het valt buiten het bestek van deze handleiding om daar op in te gaan. Zie voor dit onderwerp het TEXBook van Donald E. Knuth.

32.2 Geaccentueerde karakters

Geaccentueerde letters moeten in CONTEXT worden samengesteld. Tabel 32.1 toont hoe dit gebeurt. Het karakter *u* is hier slechts een voorbeeld.

U typt	U krijgt	U typt	U krijgt
\'{u}	ù	\u{u}	ŭ
\’{u}	ú	\v{u}	ǔ
\^{u}	û	\H{u}	ű
\"{u}	ü	\t{uu}	u͡u
\~{u}	ũ	\c{u}	ų
\={u}	ū	\d{u}	ụ
\.{u}	u̇	\b{u}	u̲

Tabel 32.1 Geaccentueerde karakters.

Omdat ì of j́ ongewenst is voor een geaccentueerde *i* en *j* worden deze letters als volgt samengesteld:

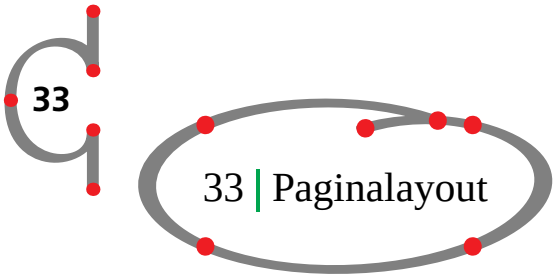
```
\"{\i}   ï
\^{\j}   ĵ
```

32.3 Buitenlandse karakters

De wijze van samenstellen van enkele buitenlandse karakters wordt in tabel 32.2 getoond.

U typt	U krijgt	U typt	U krijgt
\oe	œ	\O	Ø
\OE	Œ	\l	ł
\ae	æ	\L	Ł
\AE	Æ	\SS	ß
\aa	å	?‘	¿
\AA	Å	!‘	¡
\o	ø		

Tabel 32.2 Buitenlandse karakters.



33 | Paginalayout

De paginalayout van dit document is gedefinieerd met:

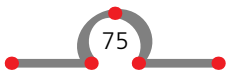
```
\stellayoutin[..,..=..,..]
```

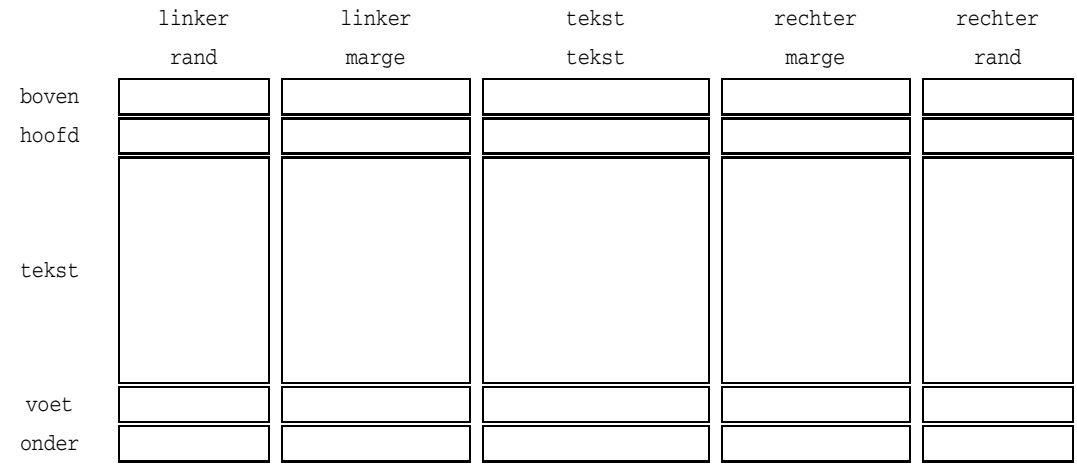
U dient bekend te zijn met de parameters waarmee de paginalayout kan worden ingesteld. Een pagina is ingedeeld in vlakken die worden aangeduid met tekst, marge, hoofd, voet enz.

De verschillende vlakken worden in figuur 33.1 schematisch weergegeven.

De paginalayout kan worden opgeroepen met `\toonkader`. Na verwerking wordt de layout met kaders weergegeven. Het commando `\tooninstellingen` geeft de instelwaarden weer. Een combinatie van beide commando is `\toonlayout`.

De waarde van de layout parameters zijn beschikbaar als commando’s (zie tabel 33.2). Dit maakt het mogelijk nauwkeurig te werken bij het definiëren van afmetingen van bijvoor-

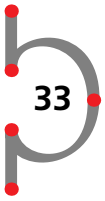




Figuur 33.1 De vlakverdeling van een pagina.

Commando	Betekenis
\zetbreedte	breedte van zetgebied
\zethoogte	hoogte van het zetgebied
\tekstbreedte	breedte van tekst vlak
\teksthoogte	hoogte van tekst vlak

Tabel 33.1 Een aantal parameters die als commando beschikbaar zijn.



beeld kolommen, figuren en tabellen. Een aantal van deze waarden wordt in tabel 33.1 toegelicht.

Indien u een breedte van een kolom of een figuur wilt definiëren is het verstandig om deze te relateren aan de \zetbreedte of \zethoogte. Bij verandering van deze waarden worden de breedte of hoogte van de kolom of figuur proportioneel meeveranderd.

```
\plaatsfiguur
  [hier]
  [fig:trapgevel]
  {Een trapgevel.}
  {\externfiguur[hass19g][type=eps,breedte=.6\zetbreedte]}
```

Na verwerking wordt figuur 33.2 geplaatst.

De overige afstanden en maten worden in tabel 33.2 getoond.

Het commando \stellayoutin wordt gedefinieerd in het instelgebied van de invoerfile, dus voor het \starttekst–commando. Dit betekent dat de waarden een globaal karakter hebben en betrekking hebben op het volledige document. Kleine wijzigingen in die layout op lokaal niveau worden gedaan met:

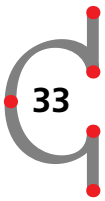
```
\paslayoutaan[21,38][hoogte=+.5cm]
```

In dit geval wordt op pagina 21 en 38 de standaardhoogte met 0,5 cm verhoogd.

Figuur 33.2 Een trapgevel.

Voor lokale aanpassingen in de layout kunt u gebruik maken van:

```
\startlokaal ... \stoplokaal
```




```
\start

\startlokaal
  \stellayoutin[hoogte=+.5cm]
\stoplokaal

Hasselt heeft een compleet andere vormgeving dan de meeste
andere steden als gevolg van de versterkingen en
verdedigingswerken.

\stop
```

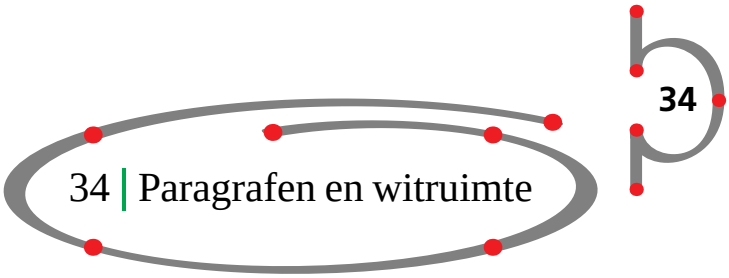
Het wordt afgeraden dergelijke tijdelijke aanpassingen te vaak uit te voeren.

Parameter	Beschikbaar als commando
bovenafstand	\bovenafstand
bovenhoogte	\bovenhoogte
hoofdafstand	\hoofdafstand
hoofdhoogte	\hoofdhoogte
kopniveau	\kopniveau
kopwit	\kopwit
rugwit	\rugwit
margeafstand	\margeafstand
margebreedte	\margebreedte
linkermargebreedte	\linkermargebreedte

Tabel 33.2a Parameters voor pagina layout.

Parameter	Beschikbaar als commando
rechtermargebreedte	\rechtermargebreedte
randafstand	\randafstand
randbreedte	\randbreedte
linkerrandbreedte	\linkerrandbreedte
rechterrandbreedte	\rechterrandbreedte
papierbreedte	\papierbreedte
papierhoogte	\papierhoogte
zetbreedte	\zetbreedte
zethoogte	\zethoogte
tekstbreedte	\tekstbreedte
teksthoogte	\teksthoogte
voetafstand	\voetafstand
voethoogte	\voethoogte
onderhoogte	\onderhoogte
onderafstand	\onderafstand

Tabel 33.2b Parameters voor pagina layout.



34.1 Inleiding

In TEX en CONTEXT is de belangerijkste eenheid van tekst een paragraaf. Een nieuwe paragraaf wordt gestart met:

- een lege regel
- het TEX–commando \par

In de ASCII invoerfile worden lege regels gebruikt als paragraafscheiders. Dit heeft als voordeel dat er een goed leesbare tekst ontstaat, waarin fouten makkelijk kunnen worden opgespoord.

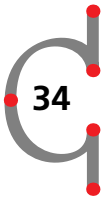
Bij het gebruiken van commando's waarin paragrafen expliciet moeten worden afgesloten, moet \par worden gebruikt.

```
Tijdens een van de oorlogen die rond Hasselt werden
uitgevochten werd Hasselt belegerd. Na enige tijd ontstond er
een voedselprobleem en brak er een hongersnood uit in de
```

```
stad. Alles wat eetbaar was werd opgegeten. Op één koe na.
Deze koe werd in leven gelaten en zelfs zeer goed
verzorgd.\par
Eén keer per dag werd de koe over de verdedigingswerken van
Hasselt geleid en de bewoners zorgden ervoor dat de
belegeraars de koe goed konden zien. Zo leek het dat er
genoeg voedsel in de stad was en dat de belegering nog lang
kon duren. De belegeraars werden hierdoor zo ontmoedigd dat
ze het beleg opbraken.\par
In de Hoogstraat in Hasselt staat een huis met een gevelsteen
waarop een koe is afgebeeld. Deze steen herinnert aan de
belegering en de slimheid van de Hasselternaren.
```

Deze tekst kan ook zonder `\par`s worden ingevoerd als er met lege regels worden gewerkt.

```
Tijdens een van de oorlogen die rond Hasselt werden
uitgevochten werd Hasselt belegerd. Na enige tijd ontstond er
een voedselprobleem en brak er een hongersnood uit in de
stad. Alles wat eetbaar was werd opgegeten. Op een koe na.
Deze koe werd in leven gelaten en zelfs zeer goed verzorgd.

Een keer per dag werd de koe over de verdedigingswerken van
Hasselt geleid en de bewoners zorgden ervoor dat de
belegeraars de koe goed konden zien. Zo leek het dat er
genoeg voedsel in de stad was en dat de belegering nog lang
kon duren. De belegeraars werden hierdoor zo ontmoedigd dat
ze de belegering opbraken.

In de Hoogstraat in Hasselt staat een huis met een gevelsteen
waarop een koe is afgebeeld. Deze steen herinnert aan de
belegering en de slimheid van de Hasseltenaren.
```



34.2 Witruimte tussen paragrafen

De verticale witruimte tussen paragrafen wordt ingesteld met:

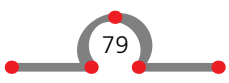
```
\stelwitruimtein[...]
```

Dit document wordt geproduceerd met `\stelwitruimtein[middel]`.

Wanneer de witruimte tussen paragrafen is ingesteld, zijn de volgende commando's beschikbaar, hoewel ze zelden hoeven te worden gebruikt:

```
\geenwitruimte
\witruimte
```

Indien paragrafen lijnen bevatten dan verdient witruimte extra aandacht, bijvoorbeeld bij:
8061GH Hasselt



moet een correctie worden uitgevoerd. Deze correctie kan worden uitgevoerd met:

```
\startregelcorrectie ... \stopregelcorrectie
```

Indien u zou intypen:

```
\startregelcorrectie
\omlijnd{8061GH Hasselt}
\stopregelcorrectie
```

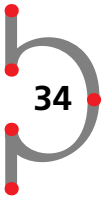
dan krijgt u een beter resultaat.

8061GH Hasselt

Een ander commando dat betrekking heeft op verticale witruimte is:

```
\blanko[..,...,..]
```

Het hakenpaar is optioneel en u kunt tussen haken de hoeveelheid witruimte opgeven. De mogelijke opties zijn veelvouden van: `klein`, `middel` en `groot` en zijn gerelateerd aan de korpsgrootte.




```
In officiële aanduidingen gaat de naam Hasselt altijd
vergezeld van de afkorting Ov. Dit is een afkorting van de
provincie Overijssel.
\blanko[2*groot]
Het grappige is dat er in Nederland geen tweede Hasselt is.
De toevoeging is daarom overbodig.
\blanko
De toevoeging is een overblijfsel uit de tijd dat Nederland
en België nog tot hetzelfde koninkrijk behoorden.
\blanko[2*groot]
Hasselt in België ligt in de provincie Limburg. Zouden de
Belgen hun brieven adresseren met Hasselt (Li)?
```

Het commando `\blanko` zonder haken is de standaard witruimte. Het voorbeeld komt er als volgt uit te zien:

In officiële aanduidingen gaat de naam Hasselt altijd vergezeld van de afkorting Ov. Dit is een afkorting van de provincie Overijssel.

Het grappige is dat er in Nederland geen tweede Hasselt is. De toevoeging is daarom overbodig.

De toevoeging is een overblijfsel uit de tijd dat Nederland en België nog tot hetzelfde

koninkrijk behoorden.

Hasselt in België ligt in de provincie Limburg. Zouden de Belgen hun brieven adresseren met Hasselt (Li)?

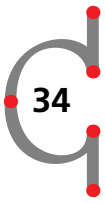
De witruimte kan worden ingesteld met:

```
\stelblankoin[...]
```

Verticale witruimte kan worden onderdrukt met het commando–paar:

```
\startopelkaar[..,...,..] ... \stopopelkaar
```

```
\alinea Hasselt (Ov) \\ Overijssel \\
\alinea Hasselt (Li) \\ Limburg    \\

\startopelkaar
\alinea Hasselt (Ov) \\ Nederland \\
\alinea Hasselt (Li) \\ België \\
\stopopelkaar
```



Hasselt (Ov) Overijssel

Hasselt (Li) Limburg

Hasselt (Ov) Nederland
Hasselt (Li) België

De tegenhanger hiervan is:

```
\startvanelkaar ... \stopvanelkaar
```

Een verticale verplaatsing over een bepaalde afstand kan worden afgedwongen met `\omlaag`. De verschuiving wordt tussen de vierkante haken ingesteld.

```
\omlaag[...]
```

34.3 Inspringen

Indien u de eerste regel van een paragraaf wilt laten inspringen, typt u:

```
\inspringen[...]
```

in het instelgebied van de invoerfile. Tussen haken worden de voorkeuren aangegeven. Standaard staat deze voorkeur op `nooit`.

Als inspringen aan staat, zult u expliciet moeten aangeven wanneer een paragraaf *niet* hoeft in te springen. Dit wordt gedaan met:

```
\nietinspringen
```

De afstand waarmee wordt ingesprongen, wordt ingesteld met:

```
\stelinspringenin[...]
```

Instellingen van commando's worden in het instelgebied van de invoerfile geplaatst. De commando's hebben een globaal karakter en zijn van toepassing op het volledige document.

In bijlage D is een compleet overzicht gegeven van alle commando's en de mogelijke parameters en instellingen.

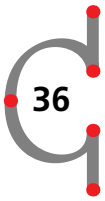
De `stel...in`–commando's hebben allen dezelfde structuur en zien er bijvoorbeeld als volgt uit:

```
\stelalineasin[.1.][.2.][..,..=..,..]

.1.           naam
.2.           getal elk
letter        normaal vet schuin vetschuin type kap klein... commando
breedte       maat
hoogte        maat
uitlijnen     links rechts midden breedte
tolerantie    zeerstreng streng soepel zeersoepel
afstand       maat
voor          commando
na            commando
binnen        commando
commando      commando
lijn          aan uit
```

Een instellingscommando bestaat uit een min of meer logische naam en een aantal vierkante haken. De vierkante haken kunnen optioneel zijn. In dat geval zijn de `[ ]` in de commandodefinitie schuin gedrukt *[ ]*.

```
\steleencommandoin[.1.][.2.][..,..=..,..]
```

De komma's geven aan dat er een lijst van parameters kan worden ingegeven. De lijst met opties die bij de definitie is opgenomen begint met `.1.` en `.2.`. Deze geven de mogelijke opties aan die in het eerste en tweede paar haken kunnen worden opgenomen. Vervolgens worden parameters en hun mogelijke waarden in het derde paar haken geplaatst.



De standaardopties en -waarden zijn in de definitie onderstreept. Bovendien zijn enkele waarden schuin gedrukt: *sectie*, *naam*, *maat*, *getal*, *commando* en *tekst*. Deze waarden kunt u zelf invoeren.

sectie	verwacht een sectienaam, zoals hoofdstuk, paragraaf enz.
naam	verwacht een logische naam
maat	verwacht een getal met eenheid in `cm pt em ex sp in`
getal	verwacht een getal
commando	verwacht een commando, omgeven door `{}`
tekst	verwacht tekst

36 | Definiëren van commando's / macro's

CONTEXT is een set macro's gebaseerd op TEX. TEX is zowel een typografisch systeem als een programmeertaal. Dit betekent dat u ook zelf programma's cq. macro's kunt schrijven

indien u een dergelijke flexibiliteit nodig heeft.

Een nieuw commando wordt gedefinieerd met:

```
\definieer[.1.]\commando{.2.}
```

Het een en ander wordt gedemonstreerd door middel van een voorbeeld.

U heeft een rijk geïllustreerd document en u wordt er moe van om steeds bij iedere figuur het volgende in te typen:

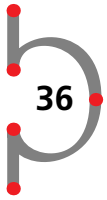
```
\plaatsfiguur
  [hier,forceer]
  [fig:logische naam]
  {Bijschrift.}
  {\externfiguur[filenaam][type=eps,breedte=5cm]}
```

U kunt een eigen commando maken waarin een aantal variabelen worden opgenomen, zoals:

- logische naam
- bijschrijft
- filenaam

De commandodefinitie zou er als volgt uit kunnen zien:

```
\definieer[3]\plaatsmijnfiguur%
  {\plaatsfiguur
     [hier,forceer]
     [fig:#1]
     {#2}
     {\externfiguur[#3][type=eps,breedte=5cm]}}

\plaatsmijnfiguur
   {leeuw}
   {De Nederlandse leeuw houdt de wacht.}
   {hass13g}
```



Het tussen haakjes geplaatste `[3]` geeft aan dat het commando drie variabelen verwacht: `#1`, `#2` en `#3`. In de commando–aanroep van `\plaatsmijnfiguur` staan de variabelen tussen accolades. Het resultaat kan er als volgt uitzien:

Op deze manier kunnen zeer geavanceerde commando's worden gedefinieerd, maar dat wordt aan uw eigen inventiviteit overgelaten.

In aanvulling op het definiëren van commando's kunt u ook zelf `\start` ··· `\stop` paren definiëren.

Figuur 36.1 De Nederlandse leeuw houdt de wacht.

```
\definieerstartstop[...][..,..=..,..]
```

Bijvoorbeeld:

```
\definieerstartstop
   [stars]
   [commandos={\inlinker{\hbox to \linkermargebreedte
                {\leaders\hbox{$\star$}\hfill}}},
    voor=\blanko,
    na=\blanko]

\startstars {\em Hasselter Juffers} zijn een soort zoete
koekjes en hun naam berust niet op toeval. Op 21 juli 1233
werd het Zwartewaterklooster opgericht. Het klooster was
bedoeld voor ongetrouwde meisjes en vrouwen van stand. Deze
meisjes en vrouwen werden {\em juffers} genoemd. \stopstars
```

Dit resulteert in:

****

Hasselter Juffers zijn een soort zoete koekjes en hun naam berust niet op toeval. Op 21 juli 1233 werd het Zwartewaterklooster opgericht. Het klooster was bedoeld voor ongetrouwde meisjes en vrouwen van stand. Deze meisjes en vrouwen werden *juffers* genoemd.

37 | Diversen

37.1 Zwevende blokken / Floating blocks

Een blok is in CONTEXT een tekstelement, zoals een tabel of een figuur die op een speciale manier wordt afgehandeld. U heeft al het gebruik van \plaatsfiguur en \plaatstabel gezien. Beide zijn voorbeelden van floating blocks of zwevende blokken. Het floatmechanisme is beschreven in hoofdstuk 10 en 11.

Dergelijke blokken kunnen worden gedefinieerd met:

```
\definieerplaatsblok[.1.][.2.]
```

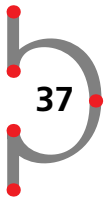
Tussen de vierkante haken wordt de naam in enkel- en meervoud vermeld. Bijvoorbeeld:

```
\defineerplaatsblok[intermezzo][intermezzos]
```

Na deze definitie zijn de volgende commando's beschikbaar:

```
\plaatsintermezzo[][]{}{}
\startintermezzotekst ... \stopintermezzotekst
\plaatslijstmetintermezzos
\volledigelijstmetintermezzos
```

Het nieuw gedefinieerde (zwevende) blok kan worden ingesteld door middel van:




```
\stelplaatsblokin[...][..,..=..,..]
```

De layout van dergelijke blokken wordt ingesteld met:

```
\stelplaatsblokkenin[..,..=..,..]
```

De nummering en de labels worden ingesteld met:

```
\stelblokkopjesin[..,..=..,..]
```

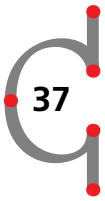
Deze commando's worden meestal in het instelgebied van de invoerfile geplaatst en zijn geldig voor alle floating blocks in het document.

```
\stelplaatsblokkenin[plaats=midden]
\stelblokkopjesin[plaats=onder,kopletter=vetschuin]

\plaatsintermezzo{Een intermezzo.}
\startkadertekst
Aan het begin van deze eeuw liep er een tramlijn van Zwolle
naar Blokzijl via Hasselt. Andere vormen van transport
werden belangrijker en net voor de Tweede Wereldoorlog werd
de lijn opgeheven. Tegenwoordig zou zo'n tramlijn best
rendabel kunnen draaien.
\stopkadertekst
```

> Aan het begin van deze eeuw liep er een tramlijn van Zwolle naar Blokzijl via Hasselt. Andere vormen van transport werden belangrijker en net voor de Tweede Wereldoorlog werd de lijn opgeheven. Tegenwoordig zou zo'n tramlijn best rendabel kunnen draaien.

Intermezzo 37.1 Een intermezzo.

37.2 Tekstblokken



Een ander soort blok is het tekstblok. Een tekstblok is meestal een stuk tekst dat meerdere malen in een document wordt gebruikt (maar dat u maar eenmaal wilt invoeren).

Een tekstblok wordt gedefinieerd met:

```
\definieerblok[...]
```

De naam van het tekstblok wordt tussen vierkante haken geplaatst. Het is ook mogelijk een lijst van namen op die plaats in te voeren. De namen worden gescheiden door komma's.

U kunt bijvoorbeeld het volgende blok definiëren:

```
\definieerblok[nederlands]
```

Vervolgens is na deze definitie het volgende commando–paar beschikbaar:

```
\beginvannederlands ... \eindvannederlands
```

Blokken worden gemanipuleerd met:

```
\verbergblokken[..,.1.,..][..,.2.,..]
```

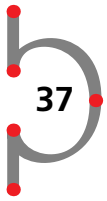
```
\gebruikblokken[..,.1.,..][..,.2.,..]
```

```
\handhaafblokken[..,.1.,..][..,.2.,..]
```

```
\selecteerblokken[..,.1.,..][..,.2.,..][..=..]
```

Hieronder wordt met een voorbeeld de werking van tekstblokken geïllustreerd. Tekstblokken worden voornamelijk gebruikt bij vragen en antwoorden in studieboeken of in meertalige documenten.

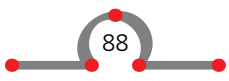
```
\definieerblok[nederlands,engels]

\verbergblokken[nederlands,engels]

\beginvanengels[dedemsvaart-e]
Since 1810 the Dedemsvaart caused some prosperity in
Hasselt. All ships went through the canals of Hasselt and the
shops on both sides of the canals prospered.
\eindvanengels

\beginvannederlands[dedemsvaart-n]
Sinds 1810 zorgde de Dedemsvaart voor enige welvaart in
Hasselt. Alle schepen voeren door de grachten en de
winkels aan weerszijden van de gracht floreerden.
\eindvannederlands

\gebruikblokken[engels][dedemsvaart-e]
```



Indien u dergelijke blokken consequent zou gebruiken kunt u een meertalige document maken. Voor dat doel is het dan ook mogelijk tekstblokken in een aparte externe file op te slaan. Dat ziet er als volgt uit:

```
\stelblokin[nederlands][file=bewaar-n]
```

De nederlandse tekstblokken worden bewaard in `bewaar-n.tex` en de tekstfragmenten kunnen met hun logische naam worden aangeroepen. Met `\stelblokin` wordt de weergave ingesteld.



37.3 Tekst bufferen (bewaren voor later gebruik)

Informatie kan tijdelijk worden opgeslagen om later in het document te worden gebruikt. Deze optie wordt het bufferen van teksten genoemd.

```
\startbuffer[...] ... \stopbuffer
```

Bijvoorbeeld:

```
\startbuffer[visite]
Als u wilt weten wat Hasselt u kan bieden, moet u dit stadje
maar eens komen bezoeken.
\stopbuffer

\haalbuffer[visite]
```

Met `\haalbuffer[visite]` wordt de tekst opgeroepen. De logische naam is optioneel. Met `\typebuffer[visite]` wordt de getypte tekst van de tekstbuffer opgeroepen en geplaatst.

Buffers worden ingesteld met:

```
\stelbufferin[..,..=..,..]
```

37.4 Tekst verbergen

Tekst worden verborgen met:


```
\startverbergen ... \stopverbergen
```

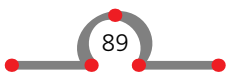
De tekst tussen dit commando–paar wordt niet verwerkt.

37.5 Lijnen

Er zijn vele commando’s om lijnen te tekenen. Om een enkele lijn te trekken, typt u:

```
\haarlijn
```

of:



```
\dunnelijn
```

Meerdere lijnen worden opgeroepen met:

```
\dunnelijnen[..=..]
```

Tekst en lijnen kunnen ook worden gecombineerd.

```
\starttekstlijn[.1.]{.2.} ... \stoptekstlijn
```

Bijvoorbeeld:

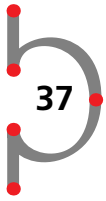
Hasselt – Amsterdam
Als u een rechte lijn trekt van Hasselt naar Amsterdam moet u een afstand overbruggen van bijna 145 km.

Als u twee rechte lijnen trekt van Hasselt naar Amsterdam dan overbrugt u een afstand van 290 km.

Amsterdam

Hasselt

Deze voorbeelden zijn als volgt ingevoerd:




```
\starttekstlijn{Hasselt -- Amsterdam}
Als u een rechte lijn trekt van Hasselt naar Amsterdam moet
u een afstand overbruggen van bijna 145 \Kilo \Meter.
\stoptekstlijn

Als u twee rechte lijnen trekt van Hasselt naar Amsterdam
dan overbrugt u een afstand van 290 \Kilo \Meter.

Amsterdam \dunnelijnen[n=3] Hasselt
```

Het tekenen van lijnen verdient altijd extra aandacht. De witruimte voor en na de lijnen wil nog weleens anders worden dan in eerste instantie mag worden verwacht.

De afstand tussen lijnen kunt u instellen met:

```
\steldunnelijnenin[..=..]
```

Er zijn enkele aanvullende commando's:

```
\stelinvullijnenin[..,..=..,..]
```

```
\stelinvulregelsin[..,..=..,..]
```

Deze commando's worden in voorbeelden geïllustreerd:

```
\stelinvullijnenin[breedte=2cm]
\stelinvulregelsin[breedte=3cm]

\invullijnen[n=1]{\bf naam}
\invullijnen[n=3]{\bf adres}

\invulregel{Kunt u het \onderstreep{aantal}
auto's aangeven dat in uw gezin wordt gebruikt?} \par

Streep ieder woord door \doorstrepen{in deze tekst}\punten[18]
```

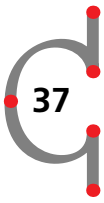
Dit wordt na verwerken:

naam _____

adres _____

_____

_____



Kunt u het aantal auto's aangeven dat in uw gezin wordt gebruikt? _____

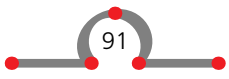
Streep ieder woord door ~~in deze tekst~~ . . . . . .

Deze commando's zijn ontwikkeld voor vragenlijsten e.d.

Opgemerkt moet worden dat TEX tekst die wordt doorgehaald met `\doorstreep` of `\doorstrepen` niet afbreekt.

37.6 Super- en subscript in tekst

Het is vrij eenvoudig om $^{\text{superscript}}$ en $_{\text{subscript}}$ in de tekst te plaatsen. Hoe dit $^{\text{superscript}}_{\text{subscript}}$ wordt genoemd, is niet bekend, maar het ziet er niet uit.



Deze tekst is gemaakt met `\laag{}`, `\hoog{}` en `\laho{}{}`. De tekst wordt tussen de accolades geplaatst.

37.7 Datum

De systeemdatum kan in uw document worden opgenomen met:

```
\huidigedatum
```

37.8 Positioneren

Voor zeer speciale toepassingen is het soms wenselijk tekst op een pagina te positioneren. Positioneren gebeurt met:

```
\positioneer(.1.,.2.){.3.}
```

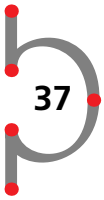
De haakjes omsluiten de x, y–coördinaten, de accolades bevatten de tekst die moet worden gepositioneerd.

Het x, y–stelsel wordt ingesteld met:

```
\stelpositionerenin[..,..=..,..]
```

Bij het instellen kan gebruik gemaakt worden van schaalfactoren en eenheden. Een voorbeeld licht het commando `\positioneer` verder toe.

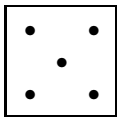
```
\def\dobbelvijf%
  {\omlijnd
     [breedte=42pt,hoogte=42pt,offset=0pt]
     {\stelpositionerenin
        [eenheid=pt,factor=12,xoffset=-11pt,yoffset=-8pt]%
      \startpositioneren
        \positioneer(1,1){$\bullet$}%
        \positioneer(1,3){$\bullet$}%
        \positioneer(2,2){$\bullet$}%
        \positioneer(3,1){$\bullet$}%
        \positioneer(3,3){$\bullet$}%
      \stoppositioneren}}

\plaatsfiguur{Dit is vijf.}{\dobbelvijf}
```



Dit toch wel lastige voorbeeld komt er als volgt uit te zien.

37.9 Roteren van tekst, figuren en tabellen

In een aantal gevallen is het noodzakelijk om teksten, figuren of tabellen te roteren. Der-



Figuur 37.1 Dit is vijf.

gelijke objecten worden geroteerd met:

```
\roteer[..,..=..,..]{...}
```

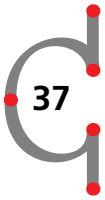
De vierkante haken zijn optioneel en worden gebruikt om de rotatie in te stellen: `rotatie=90`. De accolades bevatten de tekst of het object dat geroteerd moet worden.

```
Hasselt heeft haar stadsrechten in 1252 gekregen. Vanaf die
tijd bezat Hasselt het \roteer[rotatie=90]{recht}
om een eigen zegel op officiële documenten te plaatsen. Het
zegel toont de Heilige Stephanus die bekend staat als een van
de eerste christelijke martelaren en is hij de beschermheilige
van \roteer[rotatie=270]{Hasselt}. Na de Reformatie werd het
zegel opnieuw ontworpen. Bovendien verloor Stephanus zijn
'heiligheid'. Vanaf die dag wordt hij dan ook afgebeeld zonder
aureool.
```

Dit resulteert in een wel erg lelijke tekst:

Hasselt heeft haar stadsrechten in 1252 gekregen. Vanaf die tijd bezat Hasselt het recht om een eigen zegel op officiële documenten te plaatsen. Het zegel toont de Heilige Stephanus die bekend staat als een van de eerste christelijke martelaren en is hij de beschermheilige van Hasselt. Na de Reformatie werd het zegel opnieuw ontworpen. Bovendien verloor Stephanus zijn 'heiligheid'. Vanaf die dag wordt hij dan ook afgebeeld zonder aureool.

Ook figuren kunt u roteren met:

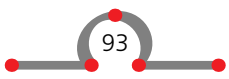



```
\plaatsfiguur
  {De Vispoort is 180 \Degrees\ geroteerd.}
  \roteer[rotatie=180]
  {\externfiguur[hass15g][type=eps]}
```

U ziet dat roteren een figuur soms erg onduidelijk kan maken.

37.10 Nieuwe regel

Een nieuwe regel kan worden afgedwongen met:



Figuur 37.2 De Vispoort is 180° geroteerd.

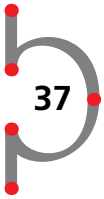
```
\crlf
```

De afkorting `\crlf` staat voor carriage return en linefeed.

Wanneer meerdere regels onder elkaar moeten worden geplaatst en moeten worden afgebroken op een door u aangegeven plaats kunt u dat als volgt doen:

```
\startregels ... \stopregels
```

```
Op een houten paneel in het stadhuis kan men lezen:

\startregels
Heimelijcken haet
eigen baet
jongen raet
Door diese drie wilt verstaen
is het Roomsche Rijck vergaen.
\stopregels

Dit rijmpje waarschuwt magistraten van Hasselt ervoor dat
persoonlijke voordelen en gevoelens de besluitvorming
niet mogen beïnvloeden.
```



Op een houten paneel in het stadhuis kan men lezen:

Heimelijcken haet
eigen baet
jongen raet
Door diese drie wilt verstaen
is het Roomsche Rijck vergaen.

Dit rijmpje waarschuwt magistraten van Hasselt ervoor dat persoonlijke voordelen en gevoelens de besluitvorming niet mogen beïnvloeden.

In enkele commando's worden regelovergangen gegenereerd met `\\`. Als u het commando `\inmarge{in de\\marge}` intypt wordt de margetekst over twee regels verdeeld.

37.11 Afbrekingen

Bij het schrijven van meertalige teksten dient u er rekening mee te houden dat afbreekmechanismen per taal kunnen verschillen.

Een afbreekmechanisme wordt geactiveerd met:

```
\taal[...]
```

Tussen de vierkante haken typt u het taalgebied in `nl`, `fr`, `en`, `de` en `sp`.

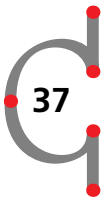
Om over te gaan van de ene taal op de andere kunt u de verkorte schrijfwijze hanteren:

```
\nl  \en  \de  \fr  \sp
```

Het voorbeeld hieronder geeft enkele overgangen van talen weer:

```
\en If you want to know more about Hasselt you could probably
best read {\nl \em Uit de geschiedenis van Hasselt} by
F.~Peereboom.
```

If you want to know more about Hasselt you could probably best read *Uit de geschiedenis van Hasselt* by F. Peereboom.



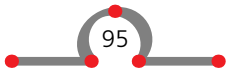
Het afbreekmechanisme van TEX en dus ook van CONTEXT is zeer goed. Indien het voorkomt dat TEX een woord verkeerd afbreekt, kunt u zelf een afbreekpatroon definiëren. Dergelijke afbreekpatronen worden in het instelgebied van de invoerfile gedefinieerd met:

```
\hyphenation{ge-schie-de-nis}
```

37.12 Invoer van andere `tex`–files

Informatie kan in meerdere TEX–files worden ondergebracht om vervolgens op de juiste plaats in de invoerfile te worden geladen. Het kan bijvoorbeeld efficiënter zijn om een document op te splitsen in meerdere files, zodat partieel verwerken mogelijk wordt.

Een andere TEX–file (met de naam `eenfile.tex`) kan in de invoerfile worden geladen



met:

```
\input eenfile.tex
```

De extensie is optioneel, dus werkt dit ook:

```
\input eenfile
```

Het commando `\input` is een TEX–commando.

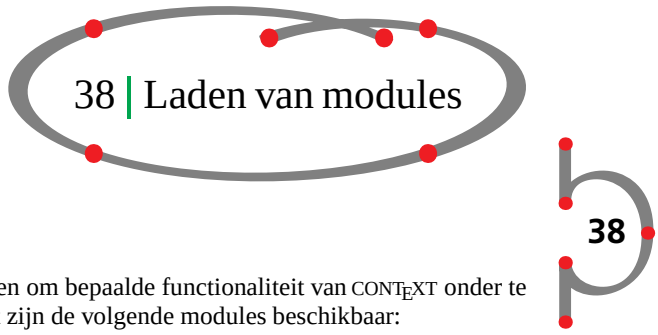
37.13 Commentaar in de invoerfile

Alle tekst tussen `\starttekst` en `\stoptekst` wordt tijdens de verwerking met CONTEXT meegenomen en gezet. Het kan echter zijn dat u tekstfragmenten wel wilt bewaren, maar niet wilt laten verwerken. Ook kan het zijn dat u uw opmaak wilt voorzien van commentaar.

Alle tekst die wordt voorafgegaan door een `%`–teken wordt door CONTEXT gezien als commentaar en wordt niet verwerkt.

```
% In grote documenten kunt u de verschillende onderdelen
% onderbrengen in meerdere files.
%
% Bijvoorbeeld:
%
% \input hass01.tex  % hoofdstuk 1 over Hasselt
% \input hass02.tex  % hoofdstuk 2 over Hasselt
% \input hass03.tex  % hoofdstuk 3 over Hasselt
```

Als u de `%` zou weghalen voor de `\input`–commando's dan worden de drie files geladen en op die plek in het document geplaatst. Het commentaar dat de inhoud van de files beschrijft wordt echter niet meegenomen.



Uit oogpunt van efficiency is besloten om bepaalde functionaliteit van CONTEXT onder te brengen in modules. Op dit moment zijn de volgende modules beschikbaar:

- `chemie` voor het zetten van chemische structuren
- `eenheid` voor het gebruik van SI–eenheden
- `pictex` voor het tekenen van plaatjes (wordt gebruikt in combinatie met module `chemie`)

Een module wordt in het instelgebied van de invoerfile geladen door middel van:

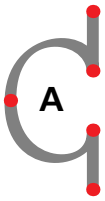
```
\gebruikmodule[..,...,..]
```

Als CONTEXT wordt aangeroepen worden enkele voorgedefinieerde instellingen geladen. Deze instellingen worden gedaan in de file `cont-sys.tex`. Gebruikers kunnen hun eigen instellingen in deze file opnemen. Het is wel de bedoeling dat CONTEXT deze file kan vinden in de door u aangemaakte directory structuur. Een voorbeeld van een instelling in deze file is:

```
\gebruikspecials[reset,ps,tr,pdf]
```

Dit commando zorgt ervoor dat de in de DVI file opgenomen specials geschikt zijn voor DVIPS.

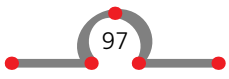
Een tweede belangrijke file is `cont-usr`. Deze file wordt gebruikt tijdens het genereren van een zogenaamde format file, bijvoorbeeld `cont-nl.fmt`. In deze file `cont-usr` kan men aangeven welke afbreekpatronen moeten worden geladen en welke taal actief is bij het opstarten. In de praktijk voldoen de standaardinstellingen.



De manier waarop CONTEXT wordt aangeroepen hangt af van de wijze waarop TEX is geïnstalleerd. In de praktijk zal het neerkomen op iets als:

```
tex &cont-nl mijnfile
```

of verpakt in een jasje:



```
context mijnfile
```

Tijdens het verwerken van de invoerfile schrijft CONTEXT informatie naar de file `mijnfile.tui`. Deze informatie wordt tijdens de volgende verwerkingsslag gebruikt door het programma TEXUTIL. Informatie over registers en lijsten worden naar de file `mijnfile.tuo` geschreven. De informatie in deze laatste file wordt door CONTEXT gefilterd en eventueel gebruikt.

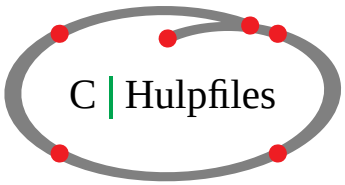
TEXUTIL is beschikbaar als een platform onafhankelijk PERL script. Dit programma wordt aangeroepen met:

```
perl texutil.pl
```

Als we geen opties meegeven krijgen we een lijst te zien. Het verwerken van een `tui` file doen we met:

```
perl texutil.pl --references mijnfile
```

Vaak zal de systeembeheerder beide verwerkingsslagen in een script of batch file opnemen, zodat de gebruiker ze als één slag kan beschouwen.



CONTEXT produceert gedurende de verschillende verwerkingsslagen een aantal hulpfiles. Indien uw invoerfile `mijnfile.tex` heet, kunnen de volgende files op uw directory worden aangemaakt.

File	Inhoud	Status
`mijnfile.tex`	tekst	niet verwijderen
`mijnfile.tui`	input informatie	kan worden verwijderd
`mijnfile.tuo`	output informatie	niet verwijderen
`mijnfile.tub`	block informatie	niet verwijderen
`mijnfile.tmp`	buffer informatie	kan worden verwijderd
`texutil.tuf`	figuur informatie	niet verwijderen
`mijnfile.dvi`	gezette tekst	kan worden verwijderd
`mijnfile.ps`	printbare tekst	kan worden verwijderd