TEX op Mac OS X

Gerben Wierda

8 november 2001

Samenvatting

Een uitgebreide introductie van TEX en TEX op Mac OS X ten behoeve van de NLUUG najaarsconferentie 2001. De bijbehorende presentatie zal noodzakelijkerwijs slechts aandacht kunnen besteden aan een deel van de inhoud van dit artikel.

1 Waarom TEX?

TEX (de X spreek je uit als een Nederlandse 'g') is een tekstzetprogramma dat ooit bedacht en geschreven is door de beroemde informaticus Donald Knuth. Het werkt met een normale ascii invoer en produceert — hoe dat zien we later – zo ongeveer de best uitziende resultaten op de planeet. Bovendien is TEX ook wel zo ongeveer het krachtigste tekstzetprogramma dat er bestaat. Een voorbeeld: In de volgende paragraaf heeft niet de auteur, maar TEX zelf bedacht waar de regelafbreking plaats moet vinden:

> The flame of conception seems to flare and go out, leaving man shaken, and at once happy and afraid. There's plenty of precedent of course. Everyone knows about Newton's apple. Charles Darwin said his *Origin of Species* flashed complete in one second, and he spent the rest of his life backing it up; and the theory of relativity occurred to Einstein in the time it takes to clap your hands. This is the greatest mystery of the human mind — the inductive leap. Everything falls into place, irrelevancies relate, dissonance becomes harmony, and nonsense wears a crown of meaning. But the clarifying leap springs from the rich soil of confusion, and the leaper is not unfamiliar with pain. *John Steinbeck* – Sweet Thursday
>
> ♡

TEX zelf kun je het beste beschouwen als een soort programmeertaal voor tekstzetten. Er zit een relatief klein aantal basisbewerkingen in TEX, en voor de rest is het een groot programma. Een stukje TEX code ter illustratie[1]:

```
\def\dolist{\afterassignment\dodolist\let\next= }
\def\dodolist{\ifx\next\kdbendlist \let\next\relax
  \else \\\let\next\dolist \fi
  \next}
\def\kdbendlist{\kdbendlist}
\def\hidehrule#1#2{\kern-#1%
  \hrule height#1 depth#2 \kern-#2 }
\def\hidevrule#1#2{\kern-#1{\dimen0=#1
    \advance\dimen0 by#2\vrule width\dimen0}\kern-#2}
\def\makeblankbox#1#2{\hbox{\lower\dp0\vbox{\hidehrule{#1}{#2}%
    \kern-#1 % overlap the rules at the corners
    \hbox to \wd0{\hidevrule{#1}{#2}%
      \raise\ht0\vbox to #1{}% set the vrule height
      \lower\dp0\vtop to #1{}% set the vrule depth
      \hfil\hidevrule{#2}{#1}}%
    \kern-#1\hidehrule{#2}{#1}}}}
\def\maketypebox{\makeblankbox{0pt}{1pt}}
\def\makelightbox{\makeblankbox{.2pt}{.2pt}}
\def\\{\expandafter\if\space\next\
 \else \setbox0=\hbox{\next}\maketypebox\fi}
\def\demobox#1{\setbox0=\hbox{\dolist#1\endlist}%
  \copy0\kern-\wd0\makelightbox}
```

Dat lijkt nogal complex, maar gelukkig zijn er allerlei pakketten in omlooop, een groot aantal zitten bij elke TEX-installatie, de belangrijkste daarvan is LATEX 2_ε. Het logo TEX is de volgende opdracht:

```
T\kern-.1667em\lower.5ex\hbox{E}\kern-.125emX
```

Gelukkig hoef je dat zelf niet te typen, maar typ je in een LATEX-file gewoon `\TeX`.

En dat is meteen de introductie van een tweede reden waarom werken met TEX prettig is: je schrijft eerder *conceptueel* dan *visueel*. Als je in TEX werkt word je niet snel afgeleid door visuele aspecten. Bijvoorbeeld, wanneer moet je eigenlijk aan een nieuwe alinea beginnen? Wel, veel mensen die in WYSIWYG (What you See Is What You Get) systemen schrijven willen nogal eens een nieuwe alinea beginnen als ze het gevoel hebben dat de bestaande alinea te lang wordt. Of ze willen nogal eens meerdere onderscheiden stukken aan elkaar plakken omdat de alinea anders zo mager is. Dat soort intuïties zijn haast niet te vermijden, we zijn tenslotte als mensen visueel ingesteld. Maar het levert meestal geen goede documenten op. Het WYSIWYG wordt door TEX-gebruikers dan ook wel eens smalend omschreven als What You See Is *All* You Get.

In LATEX maak je gebruik van conceptuele commando's:

1. Stukje code uit het TEX-boek

- `\section{Introductie}` als je een hoofdstuk 'Introductie' wil, bijvoorbeeld.
- `\selectlanguage{english}` als je een stukje Engels in een Nederlandstalig document zet.

En zo voorts.

Het gebruik van een macrotaal om documenten te maken heeft nog een voordeel: wijzigen van het uiterlijk van grote en complexe documenten wordt eerder makkelijker dan moeilijker. In TEXheb je veel mogelijkheden om je formattering te abstraheren. Je zet zelden zelf een fontstijl, normaal maak je gebruik van conceptuele instrumenten. Je zegt niet `\bold{let op}` maar `\emph{let op}`.

Een vierde voordeel is het mogelijk gebruik van je favoriete editor, zeg maar dezelfde editor die je ook gebruikt om je code te schrijven.

Een vijfde voordeel is dat TEX zeer geschikt is voor wiskundige formules die er ook erg mooi uitzien. Neem bijvoorbeeld[2]:

$$\begin{aligned} |I_l| &= |\textstyle\int_\Omega gRu\,\Omega| \\ &\le C_3 \left[\textstyle\int_\Omega \left(\int_a^x g(xi,t)\,d\xi\right)^2 d\Omega\right]^{1/2} \\ &\quad \times \left[\textstyle\int_\Omega \left\{u_x^2 + \frac{1}{k}\left(\int_a^x cu_t\,d\xi\right)^2\right\} c\Omega\right]^{1/2} \\ &\le C_4 \left\| f \left| \widetilde{S}_{a,-}^{-1,0} W_2(\Omega,\Gamma_1)\right\| \right\| |u|\circ \to W_2^{\widetilde{A}}\Omega;\Gamma_r,T)\right\|. \end{aligned}$$

En hoewel TEX ooit is ontstaan vanwege die behoefte aan fraai gezette wiskundige teksten is het vandaag de dag door vele vrijwilligers geschikt gemaakt voor allerlei toepasisngen, van literaire teksten tot muziek. Een professioneel uitziende brief schrijven is een fluitje van een cent met TEX.

O ja, en had ik al gezegd dat TEX gratis is?

TEX heeft wel een nadeel: het is erg tekst-geörienteerd. Voor grafische truuks in TEX zijn allerlei oplossingen in omloop die vaak afhangen van de beschikbaarheid van PostScript ergens verderop in het systeem.[3]

1.1 Hoe TEX werkt

Voor de geïnteresseerden, de ingebouwde algorithmes van TEX voor het zetten van tekst, inclusief het afbreekalgorithme, zijn onovertroffen. TEX weet van elk teken dat het moet zetten dimensies, weet van diverse tekens hoe ze combineren en heeft al die informatie ook nog eens in huis voor alle verschillende groottes (als het moet).

Om van een ascii file met TEX invoer te komen tot die mooie uitvoer zijn de volgende stappen normaal gesproken noodzakelijk:

---

2. Gejat uit de handleiding van de pdfslide macro's
3. Dit is een probleem bij het gebruik van pdfTEX omdat er in het geval van pdfTEX geen PostScript interpreter aan te pas komt.

1. TEX wordt aangeroepen om de ascii tekst (de '.tex' file) om te zetten in een '*device independent* .dvi' file. In deze file staat tusseninformatie, elk teken wordt vertegenwoordigd door een index die verwijst naar files waar bitmaps van tekens worden opgeslagen en een hokje dat de ruimte van het teken definieert. De zin: "Logica ligt altijd voor de hand" ziet er in de DVI-file dus uit als

    [4].

   Als TEX wordt aangeroepen laadt hij eerst een set met voorgecompileerde macro's. Als TEX wordt aangeroepen als `tex` dan laadt hij `tex.fmt`[5]. Wordt TEX aangeroepen met een andere naam (dus via een normale 'link', dan laadt TEX de `.fmt` file van die naam. Het commando:

   `latex foo.tex` laadt de voorgecompileerde macroset `latex.fmt` voor hij aan `foo.tex` begint.

   `pdftex foo.tex` laadt de voorgecompileerde macroset `pdftex.fmt` voor hij aan `foo.tex` begint.

   enzovoorts.

2. Vervolgens bestaan er programma's om de '.dvi' file om te zetten in scherm of printeruitvoer, de bekendste daarvan is `dvips`, het programma dat DVI informatie omzet in PostScript. Dit prgramma vult de hokjes die TEXheeft gemaakt dus in:

    [6]

3. Op de meeste systemen is er dan nog een PostScript omgeving nodig om van PostScript naar ruwe scherm- of printeruitvoer te komen. In de Unix-wereld wordt daarbij vooral gebruik gemaakt van `ghostscript`.

Vaak wordt tegenwoordig PDF als een standaardformaat gebruikt voor documentuitwisseling. Na de reeds genoemde stappen kan men dan met een apart programma de PostScript omzetten in PDF. Maar PDF is een formaat dat behalve uitvoer nog wat grappen in huis heeft, zoals navigatie. Om betere PDF uit TEX te kunnen genereren is er tegenwoordig ook een speciale versie van TEX die direct PDF genereert: pdfTEX. Samen met wat speciale macro's kan er dan vanuit TEXgebruik gemaakt worden van speciale mogelijkheden van PDF.

1.2 TEX en fonts

De kwaliteit en bruikbaarheid van je uitvoer hangt uiteindelijk sterk af van je fonts. En hoe je het ook wendt of keert, aan het eind van het hele zettraject moeten er pixels worden gezet, of dat nou voor het scherm of voor de printer is.

TEX bereikt zijn hoge kwaliteit niet alleen omdat het de bovengenoemde 'hokjes' zo goed over de beschikbare ruimte kan verdelen, maar ook omdat er goed na-

---

4. Deze weergave is gemaakt met de TEX-macro demobox die gedefinieerd is in dat stukje code dat eerder in dit artikel staat. Het gebruik van 'plain' TEX binnen een LATEX document is niet geheel zonder risico's
5. Een `.fmt` file is een geheugendump van TEX. Deze kan later weer geladen worden. Op deze wijze hoeft TEX niet alle macro's elke keer weer opnieuw te parsen en te vertalen. Vroeger heette `tex.fmt` `plain.fmt` en heette `latex.fmt` `lplain.fmt`.
6. Met dank aan Hans Hagen, een Nederlandse TEXwizard

gedacht is over fonts. Daarvoor is een krachtig fontbeschrijvingsalgorithme en bijbehorende programmatuur beschikbaar: METAFONT. Het hele oorspronkelijke mechanisme van TEX is gericht op het genereren van zo professioneel mogelijke uitvoer. Elke zetter kan je vertellen dat mooie uitvoer sterk afhankelijk is van diverse aspecten die nauw op elkaar aansluiten. En dat betekent bijvoorbeeld dat fonts van vorm veranderen als de grootte verandert, als de resolutie verandert, etc. Neem de volgende twee regels, de eerste is gezet met Computer Modern, de 5-punts versie, uitvergroot tot 17-punts. De tweede is de echte 17-punts versie:

Hello, World!
Hello, World!

je kunt goed zien dat de ontwerper rekening gehouden heeft met de leesbaarheid van de 5-punts versie. Als het zo klein is, dan moet je een beetje een vettere letter hebben en wat meer afstand en wellicht moeten onderlinge relaties tussen letters ook wel worden aangepast. Dat dat nodig is kun je hierna zien:

Hello, World!

Hello, World!

het is wel duidelijk wel font ook echt ontworpen is voor 5-punts.[7]

Feitelijk heb je voor de *beste* uitvoer dus uiteindelijk een op de resolutie en feitelijke grootte aangepaste bitmap nodig. Dat betekent dat in principe de uitvoer geoptimaliseerd is voor printer (met een hoge resolutie) of scherm (met een lage resolutie) en de een ziet er hopeloos uit op de ander. Zogenaamde schaalbare fonts (zoals de PostScript fonts) lossen dat probleem op, maar hebben uiteindelijk een suboptimaal resultaat. Voor huis-, tuin- en keukengebruik zijn schaalbare fonts meestal wel goed genoeg.

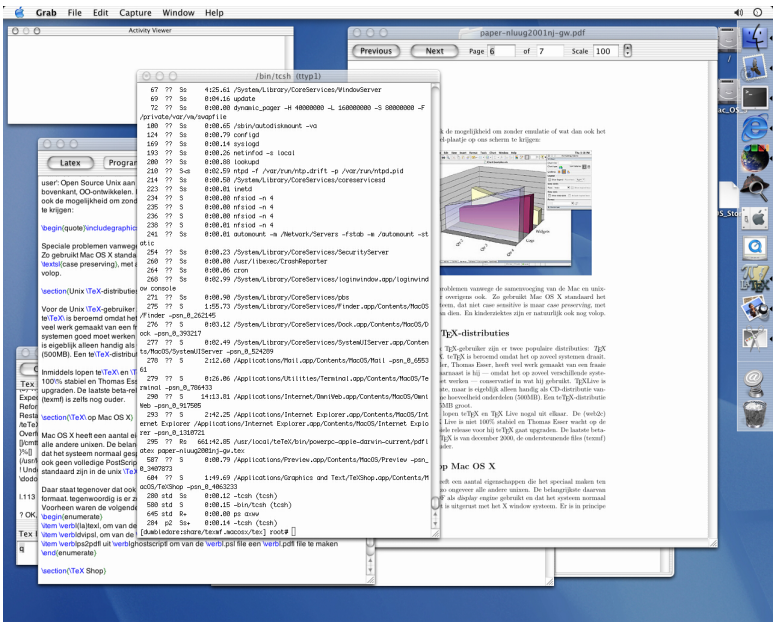
De behoefte aan het gebruik van PostScript fonts in TEX heeft eerst geleid tot een oplossing waarbij TEX kennis krijgt van de 'hokjes' van de PostScript fonts en een mechanisme waarmee de uitvoergenerator (b.v. `dvips` de PostScript fonts toevoegt.

Later zijn er zelfs PostScript-beschrijvingen gemaakt van TEX's eigen fonts (zoals Computer Modern Roman) zodat ze zich schaalbaar gaan gedragen. Dat is overigens geen sinecure want beide vormen van fontbeschrijving zijn fundamenteel niet compatibel, zodat voor elke letter uiteindelijk menselijke interventie nodig is om het er goed uit te laten zien.[8]

---

7. De bovenste is een 17-punts font op 5-punts grootte en de onderste het echte 5-punts font
8. Er is inmiddels een programma `textrace` dat METAFONT fonts omzet in `.pfb` files door ze met grote resolutie te genereren en ze vervolgens te *tracen*. De resultaten zijn redelijk.

2 Waarom Mac OS X?



Omdat Mac OS X (de X spreek je uit als het Engelse ‘ten’) uiteindelijk gewoon NEXTSTEP versie 6 is, maar dan *met* Microsoft Office. Om even terug te gaan in de historie:

In *Triumph of the Nerds*, een driedelige documentaire over de begindagen van de personal computer, legt Steve Jobs uit wat er mis was met de Mac. Bij Xerox hadden ze volgens Jobs *drie* revoluties gezien:

- De grafische gebruikersinterface,
- Het netwerk van computers,
- Objectgeorienteerde software,

en ze waren zo verblind door die eerste, dat ze de andere twee niet hadden opgemerkt. Het lag dus voor de hand dat toen Jobs vertrok bij Apple dat hij het nog een keer wilde gaan doen, maar nu goed. Het resultaat was NeXT, een computer die op een aantal punten zijn tijd ver vooruit was, zowel qua hardware maar vooral ook qua software.

NeXT had echter een probleem. Inmiddels was Windows de standaard geworden en zonder compatibiliteit met de belangrijkste Windows-software (Office suites met name) was de markt voor NeXT beperkt.

In 1994 is NeXT gestopt met het maken van hardware. In de loop van de jaren er na is NEXTSTEP beschikbaar gekomen voor 4 zeer verschillende architecturen: NeXT, Intel PC’s, Sun werkstations en HP werkstations, uiteindelijk onder de naam OPENSTEP. Op die machines (met verschillende ‘endianness’!) draaide exact dezelfde software. Ook de OO-ontwikkelomgeving was overal beschikbaar en produceerde desgevraagd binaries voor alle systemen.

NeXT is doorgegaan met het ontwikkelen van OO-omgevingen en heeft zich vooral gericht op interfaces boven databases. Uiteindelijk is daar WebObjects

uit voortgekomen, een systeem voor de ontwikkeling van Web-applicaties en — voor zover ik weet – de meest succesvolle.

Apple was intussen dik in de problemen gekomen omdat de ene na de andere poging om het besturingssysteem te moderniseren was mislukt. Uiteindelijk heeft Apple alle eigen pogingen geschrapt en heeft NeXT gekocht. De top van NeXT (inclusief de nu teruggekeerde hardware guru van weleer) vormt nu de top van Apple, en gezien de grote ommekeer heeft het Apple geen kwaad gedaan. De hardware mag er weer wezen, en de software...

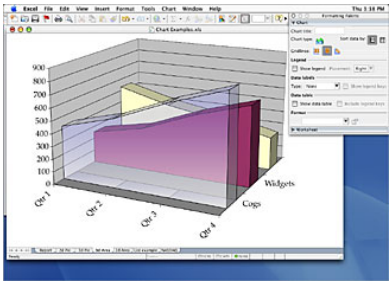
Apple heeft als basis OPENSTEP genomen, de onderkant van het broodnodige achterstallige onderhoud voorzien en aan de bovenkant fors gesleuteld. Belangrijkste stappen:

- Toevoegen van diverse technieken zoals OpenGL, Java.
- Display PostScript als weergavesysteem (dezelfde 'engine' voor scherm en printer: gegarandeerde WYSIWYG voor scherm versus printer) is vervangen door Portable Document Format (PDF) als weergavestandaard. De weergave kent een erg goede 'antialiasing': kortom, het ziet er prachtig uit.[9]
- De 'look' is een mix geworden van het klassieke Mac OS en NEXTSTEP en vervolgens opnieuw vormgegeven (Aqua).
- De 'feel' is meer en meer opgeschoven in de richting van NEXTSTEP.
- De gebruikervriendelijkheid van Mac OS op het gebied van instellingen en onderhoud is meegekomen naar Mac OS X

Al met al een zeer geslaagde mix en met voldoende mogelijkheden voor de toekomst.

Apple heeft nog een — uit het oog van de unix-gemeenschap — belangrijke stap gezet: het heeft de onderkant van zijn nieuwe besturingssysteem 'open source' gemaakt. Onder de naam *Darwin* zijn de sources van het hele unix-fundament beschikbaar. Zelfs een aantal andere onderdelen zijn gepubliceerd.

Dat alles met elkaar maakt Mac OS X in theorie een ideale combinatie voor de absolute 'power user': Open Source Unix aan de onderkant, compatibiliteit met de kantoorwereld aan de bovenkant en OO-ontwikkelen. In beeld: we hebben een 'lekkere' unix op onze desktop maar toch ook de mogelijkheid om zonder emulatie of wat dan ook het volgende Excel-plaatje op ons scherm te krijgen:



9. Als je Internet Explorer en Omniweb als internet browsers naast elkaar ziet kun je het verschil goed zien: de weergave van Internet Explorer (oude Mac OS weergave met QD) ziet er een stuk minder uit dan de 'native' weergave m.b.v. PDF

Speciale problemen vanwege de samenvoeging van de Mac en unix-wereld zijn er overigens ook. Zo gebruikt Mac OS X standaard het HFS+ filesysteem, dat niet *case sensitive* is maar *case preserving*, met alle ellende van dien. En kinderziektes zijn er natuurlijk ook nog volop.

3 Unix TEX-distributies

Voor de Unix TEX-gebruiker zijn er twee populaire distributies: *TEX Live* en *teTEX*. teTEX is beroemd omdat het op zoveel systemen draait. De samensteller, Thomas Esser, heeft veel werk gemaakt van een fraaie inrichting. Daarnaast is hij — omdat het op zoveel verschillende systemen goed moet werken — conservatief in wat hij gebruikt. TEXLive is meer up-to-date, maar is eigeblijk alleen handig als CD-distributie vanwege de enorme hoeveelheid onderdelen (500MB). Een teTEX-distributie is ongeveer 45MB groot.

Inmiddels lopen teTEX en TEX Live nogal uit elkaar. De (web2c) kern van TEX Live is niet 100% stabiel en Thomas Esser wacht op de volgende stabiele release voor hij teTEX gaat upgraden. De laatste beta-release van teTEX is van december 2000, die van de ondersteunende files (texmf) is zelfs nog ouder.

4 TEX op Mac OS X

Voor de TEX-distributie voor Mac OS X heb ik gekozen voor teTEX. Daar zijn grofweg de volgende wijzigingen op gemaakt:

- Toevoegen van de laatste pdfTEX. Dit is overigens wel een complicerende factor omdat normaal gesproken alle binaries van TEX alle voorgecompileerde formaten (zoals LATEX) kunnen lezen. Een zekere nauwkeurigheid is nodig om te zorgen dat de verschillende versies elkaar niet in de weg zitten.
- (In samenwerking met Thomas Esser): opwaarderen van `texconfig`, het command line configuratiesysteem zodat pdfTEX beter ondersteund wordt
- Toevoegen en opwaarderen van enkele pakketten in de standaard macroset (texmf), zoals `hyperref` (voor links in PDF uitvoer) en ConTEXt.
- Toevoegen van pfb-ondersteuning van enkele fonts.

Mac OS X heeft een aantal eigenschappen die het speciaal maken ten opzichte van zo ongeveer alle andere unixen. De belangrijkste daarvan is dat het PDF als *display engine* gebruikt en dat het systeem normaal gesproken niet is uitgerust met het X window systeem. Er is in principe ook geen volledige PostScript interpreter aanwezig. Daarmee ontbreken twee onderdelen die standaard zijn in de unix TEX-wereld.

Daar staat, zoals eerder genoemd, tegenover dat ook de TEX-wereld tegenwoordig beter ingespeeld raakt op het PDF-formaat. Tegenwoordig is er zoiets als pdfTEX, een TEX-programma dat direct PDF produceert. In herhaling: voorheen waren de volgende stappen noodzakelijk:

1. `(la)tex`, om van de `.tex` file een `.dvi` file te maken
2. `dvips`, om van de `.dvi` file een `.ps` file te maken.
3. `ps2pdf` uit `ghostscript` om van de `.ps` file een `.pdf` file te maken

Daarnaast was er dan nog een DVI-viewer nodig om de uitvoer ook op het scherm te bekijken, zeg maar het equivalent van `dvips`, maar dan voor het genereren van scherm-uitvoer. Maar met de beschikbaarheid van pdfTEX is er een kortere route:

1. `pdf(la)tex`, om van de `.tex` file een `.pdf` file te maken

pdfTEX heeft een nadeel: het kan wat betreft het invoegen van plaatjes diverse grafische formaten aan (en meer dan gewone TEX), maar echte PostScript (`.eps` bestanden) zit daar niet bij. Een PostScript-interpreter ontbreekt immers. En aangezien er nogal wat TEX-materiaal in omloop is waarbij gebruik is gemaakt van `dvips` om `.eps`-plaatjes in te voegen is er behoefte aan de alternatieve route voor het maken van PDF.

Hiertoe is er bij mijn TEX distributie (die ook bij TEXShop wordt verspreid) een script toegevoegd: `altpdftex`[10]. Dit script voert de hele `.tex` file → `tex` → `dvips` → `ps2pdf` → `.pdf` uit. Hiervoor is het uiteraard wel noodzakelijk dat `ghostscript`[11] op het systeem is geïnstalleerd, hetgeen meteen de reden is waarom `ghostscript` toch in mijn installer is opgenomen.

Het gebruik van `altpdftex` heeft voor- en nadelen:

- De uitvoer van `altpdftex` is mogelijk wat compatibeler met de meeste readers, pdfTEX is tenslotte nog steeds experimenteel en het wil wel eens voorkomen dat een document gemaakt met pdfTEX niet gelezen kan worden door de ontvanger, terwijl hetzelfde document gemaakt met `altpdftex` wel gelezen kan worden.
- Het is mogelijk met `altpdftex` om `.eps` plaatjes in te voegen. Er is overigens wel een alternatief: men kan de `.eps` files ook eerst omzetten in `.pdf` met behulp van `ghostscript` of een andere 'distiller' en dan vervolgens gewoon pdfTEX gebruiken. Maar voor grote projecten is dat een heidens karwei.
- Als men pdfTEX gebruikt kan men gebruik maken van speciale faciliteiten van pdfTEX, zoals het aanbrengen van links met behulp van het `hyperref` macropakket.
- `altpdftex` biedt een fijne controle over het gebruik van fonts. Zo kan men vai de command line `altpdftex` eenvoudig vertellen om zo veel mogelijk fonts als bitmaps op te nemen (voor optimaal resultaat op een bepaalde resolutie) of andersom zo veel mogelijk `pfb` fonts voor optimale schaalbaarheid.

Voor echte `.pfb` fonts wordt op Mac OS X de goede antialiasing ondersteund, hetgeen betekent dat de uitvoer er op het scherm stukken beter uitziet.

---

10. `altpdftex` is een script gebaseerd op een idee van Sean Luke
11. Versie 6 aangezien en nog een bug in Mac OS X zit bij de weergave van PDF met zowel bitmap als pfb fonts van files die gecreëerd zijn met versie 7

4.1 Intermezzo: een fijne ramp

De texmf directory van teTEX bevat ook nog het een en ander aan handleidingen. Deze zijn vrijwel zonder uitzondering in DVI en PostScript formaat. Voor de Mac OS X gebruiker zijn beide formaten niet zomaar leesbaar. Dus kwam ik op het idee om de hele texmf directory af te lopen met een scriptje dat alle `.dvi` files zou vertalen in `.pdf`. Aangezien het om `.dvi` files ging, moest dat met behulp van `dvips`.

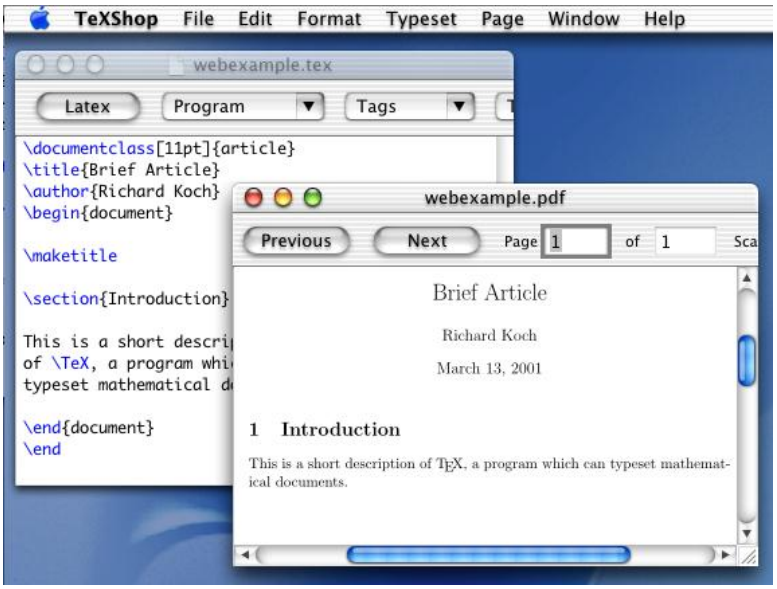
Affijn, het scriptje ging van start (dit vond plaats voor `altpdftex` geschreven was), en het duurde allemaal zo lang dat ik eerst maar eens koffie ging zetten. Na enige uren begon ik te vermoeden dat er iets mis was. Ik probeerde een shell window te openen en het systeem was enorm traag geworden. Uiteindelijk kon ik met behulp van het `ps` command zien wat er gaande was, en het bleek dat `dvips` bezig was op de file `pspicture.dvi` te bewerken. Mijn vermoeden werd later bevestigd: `pspicture.dvi` bevat opdrachten voor `dvips` om `pspicture.ps` in te voegen. En daar mijn `dvips` opdracht `pspicture.dvi` als tussenstap aan het omzetten was naar `pspicture.ps` was er een eindeloze herhaling ontstaan van `dvips` die de file `pspicture.ps` aan het invoegen was in de file `pspicture.ps`.

In de directory bleek dat de uitvoerfile `pspicture.ps` inmiddels vele gigabytes groot was en de disk was inmiddels vol. Snel gooide ik de file weg en beëindigde ik dvips met het 'kill' commando. En kwam vervolgens tot de ontdekking dat er 9GB zoek was op mijn HFS+ filesysteem. Rebooten, file system checks, de ruimte was en bleef zoek. `df` gaf keurig aan dat de ruimte bezet was, maar een `du` vond alles behalve de ontbrekende 9GB.

Uiteindelijk bleek alleen het programma `DiskWarrior` in staat om de ellende op te lossen. Mijn uiteindelijke conclusie was dat, door de file eerst weg te gooien en pas daarna het process om zeep te helpen, de file als open file nog was blijven bestaan en dat het doden van het proces (dat alleen met `SIGKILL` kon worden bereikt de filehandle niet goed had terugggeven aan het systeem. Ik vermoed dat de HFS+ filesysteem code nog niet echt was ingericht op een echt *timesharing* systeem met fatsoenlijke atomaire acties. Dat er vervolgens zo'n bizarre situatie kon ontstaan is ook merkwaardig en belooft niet veel goeds wat betreft het HFS+ systeem.

Uiteindelijk ben ik er maar op overgegaan om Mac OS X op een Unix File Systeem (UFS) te draaien. Om Mac OS X goed te kunnen draaien is — zeker voor sommige echte Mac-applicaties — wel een HFS+ file systeem echter noodzakelijk. Voor unix-mensen is een ideale opzet mijns inziens dan ook een UFS file systeem met Mac OS X op een aparte partitie en een HFS+ file systeem met Mac OS 9 op een andere partitie. Dat is nog niet ideaal, maar voor de meeste 'glitches' (zoals Word die geen files op UFS kan openen) is een workaround beschikbaar.

5 TEXShop



TEXShop is een uiterst elegant en simpel programma. Er is een editor window (met *syntax colouring*), vanuit dat window (of via het toetsenbord) kan een compile worden opgezet, aan het eind van de compile wordt de bijbehorende `.pdf` uitvoerfile geladen in een display window. Indien men dat wil kan het zetproces gevolgd worden in een log window. De auteur van TEXShop (Richard Koch van de Universiteit van Oregon) hoefde feitelijk niet zoveel te doen. PDF-ondersteuning — inclusief anti-aliasing — is ingebouwd in Mac OS X, pdfTEX bestaat ook al, het afhandelen van DVI, PostScript, etc, het is allemaal niet nodig.

Enkele eigenschappen van TEXShop op een rijtje:

- TEXShop kent de mogelijkheid om via voorkeuren de commando's te zetten die worden uitgevoerd bij het `(la)tex` commando.
- TEXShop kan per file de *project root* zetten. Dat wil zeggen dat bij grotere projecten je de file `hoofdstuk1.tex` hebt gewijzigd maar dat de compile vervolgens de file `boek.tex` gaat zetten en de file `boek.pdf` zal weergeven.
- TEXShop kan op elk moment 'templates' in de tekst invoegen, bijvoorbeeld een template voor het invoegen van een plaatje.

6 Toekomstplannen

6.1 CCS: Comment Command Settings

TEXShop's (versie 1.12) manier voor het zetten van de *project root* is niet erg elegant. Er wordt een `.texshop` gecreëerd met daarin de full path name van de file. Dat maakt het verhuizen van je files niet alleen problematischer, het is ook gewoon lelijk. Ook de mogelijkheden tot het beïnvloeden van de manier waarop een file behandeld wordt (`pdftex`? `altpdfelatex`? `make`?) is niet erg elegant.

Hiervoor hebben we een oplossing bedacht, een oplossing die ook geschikt is als standaard voor eventueel andere frontends.

Het basisidee is afkomstig van TEX op de NeXT, waar ooit een mechanisme is ingevoerd dat inmiddels tot de stadaard TEX-wereld behoort. Als de eerste regel van een `.tex` file er als volgt uitziet (% is het TEX commentaar teken):

```
%& latex
```

dan zal, ongeacht met welke TEX-variant de file wordt gezet (`latex`, `pdftex`, `etex`, etc.) `latex.fmt` worden geladen.[12]

Hiervoor zal TEXShop het volgende mechanisme krijgen:

1. Alle eerste commentaarregels worden gelezen. Een regel die aan het volgende formaat voldoet is een *comment command setting* (ccs):
   (a) Eerste *non whitespace* is een `%`
   (b) Eerste volgende *non whitespace* is een `!`
   (c) Eerste volgende alfanumerieke woord is `texshop`
   (d) *whitespace*
   (e) Een alfanumeriek woord. Dit is de naam van een variabele (ofwel de *key*)
   (f) Eerste volgende *non whitespace* is een `=`
   (g) De rest van de regel, vanaf de eerste *non whitespace*. Dit is de waarde van de variabele (ofwel de *value*). Binnen de waarde wordt de string `${naam}` vervangen door de waarde van de variabele `naam`. Binnen de waarde wordt de string `${naam:r}`vervangen door de waarde van de variabele `naam` zonder een eventuele extensie (een punt gevolgd door een string).

   Als de file moet worden gecompileerd, dan worden eerst de eerste commentaarregels gelezen en volgens het voorgaande systeem vertaald. Wanneer de ccs zijn gelezen wordt
2. `${command}` uitgevoerd, en vervolgens wordt (als er geen fouten opgetreden zijn)
3. `${output}` geopend, tenzij `${output}` een lege variabele is.

TEXShop zal voorzien zijn van de volgende standaard instellingen:

---

12. In mijn distributie kan dit tot problemen leiden, omdat ik verschillende versies van TEX door elkaar gebruik (de oude stabiele voor gewone TEX en de nieuwe voor pdfTEX) en de verschillende versies elkaars geheugendump niet kunnen lezen. Een file `foo.tex` die begint met `%& latex` en die wordt vertaald met het commando `pdflatex foo.tex` leidt dus tot een foutmelding.

`${filename}`	de filenaam van de file waarin de ccs staan, zonder directory-gedeelte
`${directory}`	de directory waar ${filename} zich bevindt. TEXShop maakt dit de *current directory*. voor dat `${command}` wordt uitgevoerd.
`${argument}`	`${filename}`
`${program}`	Het programma dat TEXShop aanroept om te compileren, afhankelijk van de instellingen van TEXShop
`${command}`	`${program} ${argument}`
`${output}`	`${argument:r}.pdf`

De kennis over hoe een document aangepakt moet worden staat in het document zelf (en kan dus niet zoekraken), het interfereert niet met TEX, en zonder wordt standaard de juiste activiteit uitgevoerd. Voorbeelden:

- TEXShop's *project root* kan worden uitgevoerd door
  ```
  % !texshop argument=bar.tex
  ```
  in de ccs te zetten.
- Voor een `.bib` file kan men bijvoorbeeld
  ```
  % !texshop program=bibtex
  % !texshop output=
  ```
  in de ccs zetten.
- En voor echt complexe projecten zegt de *power user* natuurlijk gewoon
  ```
  % !texshop program=make
  ```

7 Verwijzingen

Voor TEX ga je naar `http://www.tug.org/` of `http://www.ntg.nl/`.

TEXShop kan gevonden worden op de web site van Richard Koch:

`http://www.uoregon.edu/\~koch/texshop/texshop.html`

Op deze pagina staat ook een verwijzing naar mijn TEX-distrbutie voor Mac OS X.

Alle sources en instructies voor het genereren van mijn distributie van de sources kan gevonden worden op de onvolprezen NLUUG ftp site:

`ftp://ftp.nluug.nl/pub/comp/macosx/tex-gs/`

Voor Mac OS X zijn er vele sites, de belangrijkste is natuurlijk

`http://www.apple.com/macosx`

Voor *Darwin*, het *public source* software fundament van Mac OS X ga je naar

`http://publicsource.apple.com/`

En voor je plezier ga je naar

`http://www.userfriendly.org/`