Bijlage 2
De macros uit de nieuwe maps class

Taco Hoekwater

abstract

Dit artikel beschrijft de macros die gebruikt worden in de nieuwe MAPS class file. De nadruk ligt op de macros die proberen een stramien te ondersteunen

keywords

maps class file, macro design, stramien

1 Hoe en waarom

Wie kijkt naar de bestaande classes en stylefiles voor LaTeX, zal al snel tot de conclusie komen dat betreffende layouts eerder in elkaar geflansd zijn door een goed bedoelende amateur dan *ontworpen* door een professionele kracht. Vaak is dat vooral te zien aan slordig bij elkaar gezochte fonts, relatief te grote sectie-kopjes, en lelijke kop- en voetregels.

Ook de MAPS stylefile leed aan datzelfde probleem, met name doordat er door verschillende mensen aan geknutseld was in de loop der jaren. Bovendien was de stylefile die gebruikt werd toch al aan vervanging toe omdat het nog een oud latex2.09 bestand was, en als nieuwe hoofdredakteur wil je je natuurlijk ook een beetje afzetten tegen je voorgangers.

Na overleg binnen de redaktie hebben we Siep Kroonenberg gevraagd te komen met een nieuw ontwerp, met als belangrijkste eis dat die er 'professioneler' uit zou zien. Siep vertelt in een andere bijlage van deze MAPS over de design-beslissingen die kwamen kijken ontwikkeling van dat nieuwe class file; dit artikel gaat het hebben over de TeX-macros die nodig waren.

Met het voorstel voor een nieuwe layout, waar wij allemaal erg tevreden mee waren, begonnen de problemen van de implementatie. Het is één ding om te zeggen hoe je het wilt hebben, maar er voor zorgen dat dat ook min of meer automatisch gaat is weer heel wat anders...

- Het nieuwe class file heeft een onderliggend stramien van basislijnen (53 in totaal), waar alle `\baseline`s mee moeten lijnen.
- Er wordt op een nogal geavanceerde manier gebruik gemaakt van het NFSS. Met dank aan Siep kunnen we bijvoorbeeld 'in het jaar 1997' invoeren als `'in het jaar 1997'`, waarbij de hangende cijfers vanzelf gaan, en komt er in de hele MAPS geen enkele **Times Bold** voor, terwijl we toch gewoon `\textbf` kunnen gebruiken.
- De lijst-omgevingen wijken sterk af van de standaard latex lijsten zoals 'itemize' en 'enumerate', om het nog maar niet over het begin van het artikel te hebben
- Zoals Hans Hagen al aangaf in zijn artikel in MAPS 16, lenen zijn macros voor verbatims zich ook voor gebruik van verschillende fonts. Daarom heeft het nieuwe class file de mogelijkheid om

  a) Hans' macros te gebruiken in plaats van het standaard 'verbatim' package
  en
  b) Hebben we wat code toegevoegd zodat de omgeving 'prettyverbatim' gebruik maakt van verschillende fonts.

  Bijkomend voordeel: de macros van Hans zijn zowel stabieler als flexibeler als het 'verbatim' package.
- Grote blokken code voor tabellen en figuren zijn geleend van de class files van Kluwer Academic.

Dit artikel zal zich vooral concentreren op het eerste punt. Niet dat de andere punten zoveel moeilijker of makkelijker waren, maar omdat ik denk dat deze macros het interessantst zijn.

2 Het gebruik van een stramien

Een eerste probleem bij het hebben van een stramien is dat je het zou moeten kunnen zien in de uitvoer. Op die manier is er tenminste controle mogelijk of het allemaal wel netjes werkt. Het wordt dan immers een stuk makkelijker om aan te wijzen welke macro de schuldige is als de kolommen weer eens niet uitlijnen.

Daarom gaan de eerste macros over het zetten van het stramien zelf. Het effect van deze macros is te zien op de volgende pagina, terwijl de macros hieronder gepresenteerd worden.[1]

Het eerste probleem is natuurlijk ervoor zorgen dat het stramien zelf op een vaste plek op de pagina staat. Er komt feitelijk maar één plaats in aanmerking, namelijk de linkerhoek van de kopregel. Helaas heeft LaTeX geen

1. De meeste macros die hier worden afgedrukt zijn ietwat aangepast. Enerzijds om te zorgen dat ze passen binnen de nauwe kolommen van de MAPS, anderzijds om de gebruikte logica zo duidelijk mogelijk te maken

nette manier om daar iets aan op te hangen, dus het moet op de volgende manier, binnen de definitie van elke \pagestyle:[2]

```
\def\ps@empty{%
  \let\@mkboth\@gobbletwo
  \def\@oddhead{\@stramien\hss}%
  \let\@evenhead\@oddhead
  \let\@oddfoot\@empty
  \let\@evenfoot\@empty}
```

Het gaat hier natuurlijk om de toevoeging van \@stramien. Dit is de eenvoudigste stijl die mogelijk is, 'headings' bijvoorbeeld ziet er als volgt uit:

```
\def\ps@headings{%
\def\@oddfoot{\sffamily\inf@@tr\hfil\thepage}%
\def\@evenfoot{\sffamily\thepage\hfil\inf@@tl}%
\def\@oddhead{\@stramien\sffamily\rightmark
    \hfil Bijlage \the@bijlagenummer}%
\def\@evenhead{\@stramien\sffamily
    Bijlage \the@bijlagenummer\hfil\leftmark}%
\let\@mkboth\@gobbletwo
\let\sectionmark\@gobble
\let\subsectionmark\@gobble}
```

Hieraan valt te zien dat er een paar eisen aan de geëxpandeerde macro \@stramien zitten:

- De macro moet een natuurlijke breedte van 0pt hebben, anders zouden de andere gegevens naar rechts opschuiven.
- De macro moet ook een diepte van nul hebben, want anders eindigt de rest van de pagina beneden het papier.

Nu definiëren we de macros \stramien (die op zijn beurt \@stramien zijn uiteindelijke betekenis geeft) en \nostramien (die ervoor zorgt dat de lijnen weer verdwijnen).

```
\def\@stramien{}
\def\stramien{\global
              \let\@stramien\@istramien}
\def\nostramien{\gdef\@stramien{}}
\newcount\strcount
```

We gaan de counter straks gebruiken om regels te tellen.

Nu de verticale plaatsing. De broodtekst moet op de lijnen vallen, en de afstand tussen de kopregel en de eerste baseline is meestal niet precies een geheel aantal \baselineskips, dus we moeten het wat netter uitrekeken:

```
\def\@istramien{%
  \global\strcount = 1
  \hbox to 0pt{%
    \vbox to 0pt{%
      \kern \topskip
      \kern \headsep
      \@@stramien\vss
      }%
    \hss
  }}
```

Volgens de LaTeX Companion (pagina 85) is de afstand tussen de baseline van de kopregel en de eerste baseline van de tekst gelijk aan \headsep, maar de redenatie daar is niet al te duidelijk, en klaarblijkelijk moest ook \topskip nog toegevoegd worden. Het begin van deze pagina ziet er daarmee uit zoals aan gegeven in figuur 1.

Vervolgens wordt het tijd voor \@@stramien zelf:

```
\def\@@stramien{\vbox{\hsize=\textwidth
                      \@strrules}}
\def\@strrules{%
    \@tempcntb\@stramienlines
    \advance\@tempcntb1
    \loop \ifnum \strcount < \@tempcntb
        \@@strrule
        \global\advance \strcount 1
     \repeat }
\def\@stramienlines{53}%
```

Kortom: 53 regels met een lijn erop. Wat overblijft is de definitie van van \@@strrule. Even een paar handigheidjes:

```
\def\thestrcount{{\scriptsize
                  \number\strcount}}
\def\@strnumber{\smash{%
    \raise \normalbaselineskip
    \hbox to \hsize{%
      \hss
      \rlap{~\ \thestrcount}%
      }%
    }}%
\def\@internalrule#1{%
    \hrule \@width #1\hsize
           \@height.1pt
           \@depth .1pt
    \kern -.2pt
    \kern \normalbaselineskip }
```

De \kern -.2pt is om het effect dat de hoogte van de \hrule zelf heeft weer ongedaan te maken.

2. Binnen latex definieren de \ps@xxxxx commando's de verschillende stijlen voor de kop- en voetregels die worden aangeroepen met het commando \pagestyle{xxxxx}. Op hun beurt definieren deze \ps@xxxxx macros de macros die uiteindelijk door de output-routine gebruikt worden om de kop- en voetregels te zetten.

```
..\vbox(11.0+0.0)x455.24408, glue....          % dit is de hele kopregel
...\glue 0.0 plus 1.0fil
...\hbox(6.73709+1.94931)x455.24408
....\hbox(6.73709+1.94931)x455.24408, glue....  % dit de tekst in de kop
.....\hbox(0.0+0.0)x0.0, glue....               % \hbox to 0pt
......\vbox(0.0+0.0)x466.62239, glue....        % \vbox to 0pt
.......\kern 10.0                               % \kern \topskip
.......\kern 25.0                               % \kern \headsep
.......\vbox(583.00081+0.0)x466.62239           % \@@stramien ...
........\rule(0.1+0.1)x455.24408
..............
......\glue 0.0 plus 1.0fil minus 1.0fil
.....\OT1/pfrl/m/n/9.5 B                        % Bijlage ...
..............
..\glue 25.0                                    % \headsep
..\vbox(583.0+0.0)x455.24408
..............
......\glue(\topskip) 3.51161                   % en dit is \topskip:
......\hbox(6.48839+2.05667)x212.62204, glue....% 3.51161 + 6.48839 = 10pt
```

Figuur 1.

`\@@strrule` roept de twee macro's hierboven aan, en dan nog wat grapjes om regelnummers in de kantlijn te zetten. De `\divide` / `\multiply` truc is de manier om in TEX modulo-berekeningen te doen.

```
\def\@@strrule{%
\@tempcnta\strcount
\divide \@tempcnta 10
\multiply \@tempcnta 10
\ifnum \@tempcnta = \strcount
        \@internalrule{1.040}%
        \@strnumber
\else
    \@tempcnta\strcount
    \divide \@tempcnta 5
    \multiply \@tempcnta 5
    \ifnum \@tempcnta = \strcount
        \@internalrule{1.020}%
        \@strnumber
    \else
        \@internalrule1
    \fi
\fi
}
```

Aan deze macro's valt beslist nog het een en ander te verbeteren (zoals de toevoeging van een gebruikers-interface en wat meer opties), maar het voldoet op dit moment heel aardig. Het zal wel nooit een apart package worden, want het toevoegen van `\@stramien` in de verschillende `\ps@xxxxx` commando's kan niet gedaan worden zonder hulp van degene die de class ontworpen heeft. Het alternatief – zelf de outputroutine aanpassen – is al helemaal uit den boze.

3 De problemen die TEX heeft met vaste basislijnen

TEX is nooit ontworpen om gebruikt te worden met een stramien zoals dit, hetgeen blijkt uit alle afwijkingen die standaard voorkomen op een pagina. En dan gaat het niet eens alleen over verticale `\skip`s, maar ook:

1. Regels die hoger zijn dan een standaard regel zorgen ervoor dat `\lineskip` wordt toegevoegd. Hierdoor wordt de regelafstand ter plekke groter, en dit komt nooit meer goed. Grote boosdoener: in-lijn formules en voetnoot-tekens.
2. Verschillen in fontgrootte veranderen de baseline uiteraard, maar het is erg moeilijk om weer 'terug te vallen'.
3. Een heleboel 'standaard' glue's moeten aangepast worden zodat ze een discrete rij mogelijkheden hebben, en geen rek of krimp componenten.
4. Display formules hebben hoogte+diepte afgeleid van de natuurlijke maten met toevoeging van de verschillende `\displayskip`s, maar de natuurlijke maten zijn niet makkelijk op te vragen
5. Hetzelfde geldt voor alle `\insert`s en `\align`s (onder andere dus latex's 'tabular'.)
6. De witruimte rond `\section`-achtige commando's is slecht te voorspellen, en dan met name in de buurt van een pagina-breuk.

Deze problemen zijn nog redelijk op te lossen, gesteld dat er gebruik wordt gemaakt van een relatief simpel format, maar bij zoiets groots als latex wordt het al snel bijna onmogelijk om het goed te doen.

Voor punt 1. is bijvoorbeeld een redelijk elegante oplossing mogelijk in plain TEX:

```
\baselineskip=0pt
\lineskiplimit=-\maxdimen
\lineskip=12pt
```

Simpel en doeltreffend, maar binnen een complex format als latex is het beslist niet veilig om zomaar even `\baselineskip` op nul te zetten!

4 Een aanzet tot een gedeeltelijke oplossing

4.1 Regelafstand

Eigenlijk valt er binnen latex maar erg weinig te doen. Het enige wat ik kon bedenken was:

```
\def\imath{\smash\bgroup$}
\def\endimath{$\egroup }
\let\(\imath
\let\)\endimath
```

Helaas gebruikt niemand(!) de combinatie `\( \)` voor inlijn wiskunde.

4.2 Font groottes

Hier is heel weinig aan te doen, en er zit niet veel anders op dan overal waar de baselines afwijken van de 'standaard' met de hand in te grijpen.

De enige situatie die we kunnen automatiseren is dat de `\baselineskip` van `\large` meestal wel gelijk gemaakt kan worden met die van `\normalsize`. Voor `\small` zou het nog mogelijk zijn een oogje dicht te knijpen, maar de andere groottes zien er onzinnig uit.

4.3 Glues en parameters

In ieder geval:

```
\parskip 0pt
```

Want `\parskip` heeft normaal gesproken een rek- component. En uiteraard moeten ook alle `\baselineskip`s niet kunnen rekken of krimpen.

Ook een hele verzameling commando's die de witruimte rond lijsten regelen dienen geherdefinieerd te worden, en niet te vergeten ook deze drie (we hebben `\le@ding` gedefinieerd als `\normalbaselineskip`):

```
\bigskipamount = \le@ding
\medskipamount = .5\le@ding
\smallskipamount = .25\le@ding
```

4.4 Displays en insertions

Hier is het optimaal bereikbare ervoor te zorgen dat de display of insert een exact aantal regels hoog is, zodat verderop een en ander weer goed gaat lopen.

We hebben twee soorten macro's nodig, namelijk één die in staat is om een `\box` af te ronden naar de juiste hoogte + diepte, en één om de bedoelde tekst in zo'n `\box` te vangen.

Het aanpassen van de hoogte gebeurt met de macro `\fixbox`. Eerst de definitie, en dan kunnen we nog even stilstaan bij de problemen die bij de implementatie te pas kwamen.

Wat we willen kunnen doen is:

```
\def\cs#1{\setbox0\vbox{#1}\fixbox0}
```

en ook

```
\def\cs#1{\setbox0\hbox{#1}\fixbox0}
```

omdat bijvoorbeeld figuren en tabulars de neiging hebben om `\hbox`-en te zijn. Het verschil blijkt weinig uit te maken. De stuctuur voor `\fixbox` is:

1. Meet de hoogte en diepte van de `\box`
2. Zoek uit hoeveel regels dat geweest zijn
3. Neem nu 1 regel meer als zijnde het gewenste aantal
4. gewenst aantal regels $\times$ hoogte van 1 regel = gewenste hoogte
5. Nodige toevoeging is gewenste hoogte $-$ echte hoogte
6. toevoegen aan de hoogte (niet de diepte!), maar wel centreren
7. zetten van de `\box`

De 'echte' `\fixbox` schrijft daarnaast nog het een en ander aan debugging informatie naar het log. Dat slaan we voor het gemak over.

```
\newdimen\addedheight
\def\fixbox#1{%
    \@tempdima=\ht#1
    \advance\@tempdima\dp#1
    \@tempdimb=\dp#1
```

De totale grootte staat nu dus in `\@tempdimb`

```
    \divide\@tempdima 11
    \@tempcnta=\@tempdima
    \divide\@tempcnta 65536
```

`\@tempcnta` is het aantal gevonden regels, afgerond naar beneden. Jammer van die maffe `\divide`, maar een assignment naar een counter vanuit een dimen register geeft een getal terug in scaled points.

```
    \@tempdima=11pt
    \advance\@tempcnta1
    \@tempdima=\@tempcnta\@tempdima
```

De gewenste hoogte staat nu in \@tempdima. Dat kon, want we hebben de originele hoogte niet meer nodig in het scratch register. In plaats daarvan komen die gegevens uit het origineel:

```
\addedheight\@tempdima
\advance\addedheight-\dp#1
\advance\addedheight-\ht#1
```

Ziezo. \addedheight bevat nu de nodige toevoeging, en na de volgende regel

```
\advance\@tempdima-\dp#1
```

Is \@tempdima de gewenste uiteindelijke hoogte. Nu moet de \box nog overnieuw verpakt worden, en de diepte hersteld, en dan zijn we klaar.

```
\setbox6=\vbox to \@tempdima{%
  \vfil\kern \addedheight\box#1\vfil}%
\dp6\@tempdimb
\box6
}
```

Niet echt ingewikkeld dus, achteraf. Maar dat neemt niet weg dat dit al minstens de vijfde versie van deze macro is, en dat er nog steeds allerlei grensgevallen zijn waarin deze oplossing niet voldoet. Vooral het laatste stukje (het overnieuw inpakken van de gebruikte box) bleek nog flink lastig.

- De allereerste versie van de macro vergat bijvoorbeeld de diepte te herstellen, met als gevolg dat de hoogte van de box een paar punten teveel werd. Dat klinkt onschuldig, maar daardoor werd het verschil tussen de box en de vorige regel (die immers zelf ook een diepte had) te klein, met als effect dat alle tekst onder een \fixbox een punt of twee te laag uitkwam.
- Een soortgelijk probleem deed zich voor toen ik probeerde de toegevoegde ruimte eerlijk te verdelen tussen hoogte en diepte, maar nu was de toegenomen diepte de boosdoener.
- De twee \vfils zijn dan weer het gevolg van het feit dat veel figuren in een box staan met diepte nul, en daardoor slecht gecentreerd uit de vorige versie van \fixbox kwamen.

  Enzovoort.

Environments Nu dient natuurlijk het class file er ook voor te zorgen dat de standaard omgevingen die last hebben van de verschuivende baselines aangepast worden.

We zullen u verder niet vervelen met alle verschillende definities voor alle environments, maar een simpel voorbeeld is het abstract. Omdat de regelafstand van het abstract kleiner is dan voor de broodtekst, is het een ideale kandidaat voor aanpassing. Het zag er eerst zo uit:

```
\newenvironment{abstract}
  {\RaggedLeft
  \small\sffamily
  \textbf{Abstract}\par\noindent
  {%
  \par\vspace*{10pt}
  }
```

En dat ziet er nu zo uit:

```
\newenvironment{abstract}
  {\setbox\@tempboxa=\vbox
   \bgroup
     \RaggedLeft
     \small\sffamily
     \textbf{Abstract}\par\noindent
  {%
     \par\vspace*{10pt}
  \egroup
  \fixbox\@tempboxa
  }
```

5 Diversen

Eén smerige truc is nog wel het vermelden waard. Het class file gebruikt ook een 'emulatie-mode' (wederom met dank aan Siep), waarin weliswaar niet exact de juiste fonts gebruikt worden, maar waarin Helvetica en consorten zo worden aangepast dat de meeste regelafbrekingen redelijk kloppen (let op: dit zegt niets over de typografische kwaliteit van de uitvoer).

Onder het NFSS betekent dat normaal gesproken dat er een stuk of drie-vier extra font definitie bestanden meegestuurd moeten worden. Dat is onhandig, maar gelukkig is er een truc om daaraan te ontsnappen:

```
\begingroup
\nfss@catcodes
\Declare.....   % kortom, het hele .fd bestand
                % minus \endinput
\endgroup
```

6 Wat er nog te doen valt

Aan de macros: eigenlijk zou het hele \section mechanisme aangepast moeten worden.

Het class file geeft anderhalve witregel boven de kop en een halve regel onder de kop, en in de gewone tekst functioneert dat naar behoren. Maar als er een paginabreuk komt net voor het kopje (en dat is iets wat we eigenlijk graag willen) blijft alleen de onderste \glue over, ter grootte van precies een *halve* regel.

Voorlopig passen we dit aan door met de hand wat aan de witruimte te rommelen, maar het moet wel automatisch kunnen. Ik heb wel een paar ideeën, maar implementatie is verre van triviaal, dus de definitieve afloop daarvan zal op de volgende MAPS moeten wachten. In ieder geval zijn er in grote lijnen (minstens) twee oplossingen mogelijk:

- Eerst de bovenste witruimte (een glue) toevoegen, en vervolgens de rest (tekst plus onderste witruimte) in een `\fixbox` plaatsen. Deze oplossing is redelijk eenvoudig, maar heeft als nadeel dat de baseline van de tekst van de kop niet meer op een stramien-regel staat.
- Maar het is ook mogelijk om eerst de glue te doen, dan een penalty, en dan kijken of de pagina afgebroken is (dat kan door te kijken of het register `\pagetotal` gelijk aan nul is). Zo ja, dan is vervolgens vrij eenvoudig de witruimte onder het kopje aan te passen vanuit de macro zelf.
  Dit lijkt een betere oplossing, maar bestaat uit veel meer werk, omdat de door LaTeX 2ε aangeboden low-level macros (`\@startsection` en dergelijke) hierbij absoluut nutteloos zijn.

PDF zaken: de ondersteuning voor `pdftex` kan nog heel wat beter, en is daarnaast feitelijk nog nauwelijks getest. En voor de versie die uiteindelijk op het web gezet gaat worden is het eigenlijk wel prettig als we nog een alternatieve pagina-layout bedenken die zich beter leent voor lezen online.

De documentatie: ontbreekt tot op heden vrijwel volledig. Dat kan natuurlijk ook weer niet als we willen dat auteurs het class file gaan gebruiken.

7 Afsluiting

Wat dankwoorden zijn hier op zijn plaats. Allereerst Siep Kroonenberg natuurlijk, voor het professionele ontwerp, het aanmaken van de benodigde NFSS en TFM bestanden, en de eerste implementatie van de macros. En dan uiteraard de redaktie zelf (en niet te vergeten Frans Goddijn), wegens de geboden feedback tijdens het testen en ontwikkelen. Hans Hagen voor de verbatim macros en Kluwer Academic Publishers voor de float macros. Als laatste iedereen die in het verleden aan de vorige versie gewerkt heeft, want zonder de oude was er beslist ook geen nieuwe geweest.

Het bestand maps.cls staat natuurlijk op de NTG web pagina (redaktie-pagina) en ook op de nieuwe 4allTeX CD-ROM. Zodra de documentatie klaar is zullen het class file en de bestanden voor de font-familie Frutiger ook op CTAN gezet worden.