Computer Typesetting

Constucties van Elastiekjes

Dit artikel beschrijft de concepten die ten grondslag liggen aan zetwerk met de computer. Als voorbeeld hiervan wordt het programma TEX gebruikt, dat al jaren oud is, maar zo goed is doordacht dat de principes nog steeds werkbaar zijn, en inderdaad ook terug te zien zijn in modernere tekstverwerkers en layout-pakketten. De grap is, dat die pakketten nog op vele punten het nakijken hebben ten aanzien van TEX!

Dit artikel is een voorbeeld bij het artikel 'Modelleren' dat ook in deze I/O-Vivat verschijnt. Verder komt er nog een artikel 'Computer Kalligrafie' over de werking van het programma MetaFont, waarmee de letters voor TEX worden getekend.

Geschiedenis

Met Schoenen van Lood

Zoals iedereen weet, is de boekdrukkunst uitgevonden door een Nederlander, wat de Amerikanen ook mogen beweren. En die Nederlander bedacht dat het handig zou zijn om een letterbak vol met lettertjes te hebben, die gegoten waren uit lood, en die op een plank met gleuven konden worden geschoven om zo te dienen als negatief voor het drukproces. Dezelfde lettertjes konden na het drukken weer opnieuw worden gebruikt voor een volgende te drukken tekst.

Dit proces vergde vakmanschap. Men diende de lettertjes een beetje fraai over de gegleufde plank (de 'pagina' dus) te verdelen, om te voorkomen dat er grote stukken wit ontstonden die storend zouden kunnen werken bij het lezen. Zetwerk heeft in hoge mate te maken met esthetisch gevoel.

Tegenwoordig doen steeds meer mensen thuis hun eigen zetwerk, onder de vlag van 'desktop publishing'. De tekstverwerker heeft voor die mensen het zetwerk op zich genomen.

Iemand, en wel Donald Knuth, heeft in 1984 eens goed nagedacht over hoe je dit proces van zetten in een computer zou moeten inbouwen. En, zo mag wel worden gesteld, met groot succes. Ook al zijn inmiddels 13 jaren verstreken, ook al hebben we WYSIWYG tekstverwerkers en laserprinters binnen ieder handbereik, ik heb nog geen stuk software gezien dat zo goed is in dit specialisme als TEX. Het lijkt me dan ook de moeite waard om eens uit te leggen wat de interne werkingsprincipes van TEX zijn.

Letters

De Vierkante Oogjes van TEX

TEX richt zich op het zetten van 2-dimensionale informatie. Die informatie bestaat uit letters, plaatjes en dergelijke. Veel van de inhoudelijke details van die dingen heeft Knuth wijselijk uit TEX weggelaten, en overgelaten aan andere programma's.

Zo zal het voor TEX worst wezen hoe een letter er uit ziet: het enige wat TEX interessant vindt zijn de afmetingen van die letter, plus wat gegevens waarmee hij in zijn output kan aangeven welke letter hij op een door hem uitgekozen positie wenst. Idem voor plaatjes: dit zijn gewoon rechthoeken. Eigenlijk is alles in TEX een rechthoek.

TEX's output is een file met de extensie `.dvi`, wat staat voor 'device independent'. Dit is zo genoemd omdat de file alleen informatie bevat over posities van rechthoekjes op pagina's, en wat er in die rechthoekjes moet komen, maar de file bevat geen bitmapped of anderszins systeemafhankelijke zaken. Er bestaan voor vele grafische uitvoerdevices omzettingen van dit formaat naar het grafische formaat van dat uitvoerdevice, bijvoorbeeld voor postscript, voor X11 displays, en voor VGA schermen.

Met de aldus verkregen, inhoudsloze rechthoekjes komen we bij de kern van wat TEX doet: schuiven met rechthoeken. En TEX's levenstaak is het vinden van de optimale verdeling van rechthoeken over een aantal pagina's. Daarvoor heeft hij allerlei rekenregels, die straf- en bonuspunten uitdelen voor allerlei vormen van positionering.

Welbekend is de weduwe/wezenregel: van een alinea wil je liever niet dat er een losse regel op de vorige of volgende pagina achterblijft

door de wijze van opbreken van een alinea over twee pagina's. Soortgelijk zijn ook afbreekregels te definiëren. TEX kent stapels van deze strafpunten-regels, en samen maken ze het mogelijk dat TEX, door alle mogelijkheden uit te proberen, uiteindelijk de optimale verdeling van een tekst vindt.

Het is maar goed dat de ouderwetse zetter met zijn loodletters niet alle mogelijkheden uit hoefde te proberen, maar in plaats daarvan op zijn ervaring af kon gaan!

Soorten Rechthoeken

De Stank van Rubber

TEX werkt dus met rechthoeken. Welke soorten rechthoeken bestaan er eigenlijk allemaal binnen TEX?

De eerste vorm van rechthoek is de informatie die de gebruiker wil typesetten. En dat zijn letters, plaatjes, enzovoort. Tegenwoordig zeggen we: objecten. Toch leuk, dat Knuth er al aan dacht de verantwoordelijkheden van de inhoud van deze rechthoeken ver buiten TEX te plaatsen en zich alleen op het zetproces te richten.

In een apart artikel zal ik ingaan op de manier waarop de letters voor TEX worden gemaakt. Hoe plaatjes tot rechthoeken worden is iedereen bekend: dit zijn bijvoorbeeld `.eps` files met een plaatje in 'encapsulated postscript' formaat, waarvan het zetpakket doorrekent met alleen de 'outline'.

Er zijn verder nog wat bijzondere rechthoeken, die door de gebruiker worden gemaakt. Eén vorm, die speciale aandacht krijgt binnen TEX, is die van wiskundige formules. TEX is zelfs nu, na al die jaren, nog steeds ongeëvenaard als het gaat om wiskundig zetwerk en gebruiksgemak! Een andere bijzondere vorm is die van tabellen. Die zijn minder prettig te maken in TEX: WYSIWYG is voor zulk werk wel weer heel prettig.

Naast de bovengenoemde vormen van rechthoeken zijn er nog vormen die TEX intern gebruikt en die minder vaak door de gebruiker worden ingezet. Dit zijn de horizontale en vertikale rechthoeken, waarin materiaal naast, respectievelijk boven, elkaar wordt verzameld. Deze rechthoeken worden met de termen 'hbox' respectievelijk 'vbox' aangeduid.

Verder kent TEX nog een speciaal soort rechthoek, en dat staat bekend als 'glue' of ook wel 'rubber'. Deze rechthoeken zijn (beperkt) rekbaar, en worden bijvoorbeeld tussen woorden geplaatst als spatie. Zo kan een regel een beetje worden uitgerekt (of ingedrukt) totdat hij in de gewenste regelbreedte past. Vanzelfsprekend is er ook rubber dat vertikaal werkt.

Het Zet-Algoritme

Metselen voor Gevorderden

Nu de ingrediënten van TEX bekend zijn, kunnen we eens gaan kijken hoe TEX nu precies werkt.

Allereerst nemen we eens een losse letter onder de loep. Die wordt door drie variabelen beschreven: een breedte, een hoogte en een diepte. TEX zal letters aansluitend aan elkaar rijgen met gebruikmaking van de breedte. Dit doet hij zo, dat al die aaneengeregen letters dezelfde basislijn volgen. De hoogte is de hoogte van de letter boven die basislijn, en de diepte is de diepte van de letter onder die basislijn. Zo komen alle letters dus netjes op de regel te staan, ook al steken ze met kop en schouders boven andere letters uit.

Nu zal de gehele regel een hbox worden. Dit is dus een rechthoek waarin horizontaal materiaal wordt verzameld. TEX voegt net zo lang letters toe aan de regel totdat die vol is. Wanneer een regel eenmaal vol is, zal TEX onderzoeken hoe erg het is om de tekst op dat punt af te breken, en dat gegeven verwerken in de berekening van de optimaliteit van de gehele alinea. Op basis van die berekening wordt de uiteindelijke layout van de alinea bepaald.

Goed, TEX heeft dus een regel vol, of deels vol, en nu moet hij die netjes af maken. Het eerste wat TEX met de regel doet, is het toevoegen van eventuele rubberen tekens. Spaties zijn al een beetje rekbaar (gespecificeerd in het font, bijvoorbeeld typemachine-letters hebben geen rekbare spaties), en de ruimte na een punt van een zin is nog beter rekbaar. Maar als de tekst bijvoorbeeld rechts rafelig mag zijn, dan voegt TEX gewoon een rubberen rechthoek toe aan het einde van de regel, en dat is een rechthoek die héél erg bereid is om te rekken — veel meer dan de rek tussen woorden of tussen zinnen — en die dus de ruimte aan het einde van de regel zal opvullen. Nu gaat TEX kijken of de tekst in de regel goed past. Dit doet hij door te bepalen hoeveel rek (of indrukking) *nodig* is, en hoeveel in totaal *mogelijk* is door alle rechthoekjes op de regel samen. Rek of comprimeer je het rubber te erg dan knapt het, TEX

geeft dan een waarschuwing ('Underful/Overful hbox') met een aanduiding van de tekst waar hij niet mee uit de voeten kon.
Nu heb je dus een regel die qua breedte mooi klopt, en die uitgelijnd is. Dan kijkt TEX nog of er niet meer dan de normale ruimte voor een regel aan de boven- of onderkant van de basisregel nodig is. Zo ja, dan neemt TEX $\mathrm{m^{ee^{r}}\,^{ru}i_{m}t_{e}}$ voor zulke regels, en anders past het mooi in de standaard regelhoogte.
Het resultaat is dus een hbox. En zo gaat TEX er ook mee om: als een doodgewone rechthoek. Alleen nu zit TEX in vertikale modus. Dat houdt in, dat TEX deze rechthoeken, die achtereenvolgende regels voorstellen, onder elkaar zal plaatsen in een vbox. En dan herhaalt zich een spel dat lijkt op hetgeen al voor regels gebeurde: TEX verzamelt net zo lang regels totdat hij een pagina vol heeft. Dan zal hij weer gaan afbreken, rubber invoegen, rekken (bijvoorbeeld tussen alinea's) en besluiten dat de pagina af is. Vervolgens pakt TEX de vbox, die de tekst van de hele pagina bevat, plakt er een kopregel boven en voetregel onder, en voert het zo ontstane geheel uit naar de `.dvi` file. Paginanummer ophogen, en op naar de volgende pagina!
Een leuk, maar ook moeilijk aspect van TEX is, dat vele van deze acties, zo ze al niet in TEX's taal zijn gedefinieerd (en dus herschreven kunnen worden), dan zijn ze in ieder geval sterk te beïnvloeden. Dat houdt in, dat de werking van TEX zeer goed aan te passen is aan individuele wensen. De primitieven waarmee men werkt zijn dezelfde als die welke de drukker met de loodletters hanteerde: rechthoekjes.

Software

Goed, Bugfree en Gratis

TEX is dus ontzettend flexibel. Het hart van TEX manipuleert rechthoekjes, en om dingen voor dat hart uit te kunnen drukken, is het mogelijk om in TEX te programmeren. Dit kan door definitie van macro's, die net zo als `#define` in C werken: ze definiëren regels voor klakkeloze tekst-herschrijving.
Om te voorkomen dat iedereen dat zelf zou moeten doen, zijn er standaard macropakketten voor TEX, met als bekendste Plain TEX en LATEX. LATEX (heel lang in versie 2.09, maar tegenwoordig in een versie die bekend staat als LATEX 2_ε) is een macro-pakket dat het mogelijk maakt om *logisch* een layout te maken, in plaats van *fysiek* zoals een WYSIWYG pakket nog wel eens uitlokt. Logische layout houdt in, dat je bijvoorbeeld specificeert dat een hoofdstuk begint, maar niet welk font, welke grootte etc. nodig zijn. Vergelijk het maar met goed doordachte stijlen in modernere tekstverwerkers.
Verder is er nog het al eerder genoemde MetaFont programma in de TEX familie. Met dit pakket kunnen fonts worden getekend. En tenslotte zijn er nog viewers en omzetters voor `.dvi` files voor allerhande grafische devices.
Al deze software is commandline driven, goed gestandaardiseerd, veel is na al die jaren vrij van bugs, en het is ook nog eens public domain. Wat ikzelf verder nog een geruststelling vind, is dat ons aller Billy-Boy er niet met zijn tengeltjes in heeft geroerd.
Voor een simpel briefje af en toe is TEX te moeilijk, maar als je interesse hebt in wat serieuzer zetwerk, dan moet je je toch eens afvragen of TEX niet honderdmaal genuanceerder en handiger werkt dan hetgeen je nu gebruikt voor je teksten.

Referenties

Voor Leesgierige Aagjes

- Donald Knuth, *The TEX book*, alles wat je ooit wilde weten over TEX, en vermoedelijk nog veel meer.
- Norbert Schwarz, *Inleiding TEX*, goede inleiding in TEX, met een duidelijk overzicht van alle commando's.
- Leslie Lamport, *LATEX, a document preparation system*, een gebruikershandleiding voor LATEX.
- Rick van Rein, *Computer Kalligrafie — Wiskundige Ganzeveren*, over de principes achter MetaFont.

Rick van Rein