METAFONT

Zabawy z METAFONT-em

Tomasz Przechlewski

Tekst ten ma stanowić zachętę do zajęcia się METAFONT-em. Oczywiste jest, że trudne będzie generowanie własnych fontów — nie wystarczy tutaj znajomość samego programu. Wystarczy ona jednak z pewnością do projektowanie prostych symboli geometrycznych czy modyfikacji istniejących czcionek a od czasu do czasu tego typu rzeczy są wielu z nas potrzebne. Z własnego doświadczenia wiem, jak trudne może być rozpoczęcie pracy z METAFONT-em stąd starałem się zamieścić w tekście dużo informacji, które — mam nadzieję — pozwolą bezproblemowo zainstalować program.

Zaczniemy od czegoś prostego, ale równocześnie choć trochę użytecznego, mianowicie zaprojektujemy znak koperty (patrz rysunek 1). Po pierwsze utwórzmy plik `koperta.mf` i wpiszmy za pomocą naszego ulubionego edytora tekstowego taki oto dziwny kod:

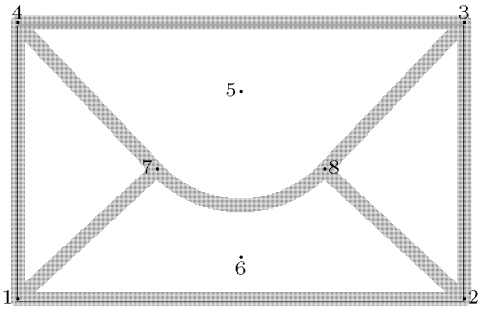
```
1. golden_ratio:=.5(sqrt(5)-1);
2. if unknown size#: size#=36pt#; fi
3. if unknown p#: 20p#= fi
4. golden_ratio*ew#=eh#=size#;
5. if unknown a.top: a.top=.75; fi
6. if unknown a.bot: a.bot=.15; fi
7. mode_setup; p:=round(p#*hppp);
8. beginchar("0",ew#,eh#,0);
9.  z1=(0,0);
10.  z2=(w,0); z3=(w,h); z4=(0,h);
11.  z5=(.5w,a.top*h); z6=(.5w,a.bot*h);
12.  z7=whatever[z1,z5]=whatever[z4,z6];
13.  z8=whatever[z2,z5]=whatever[z3,z6];
14.  pickup pencircle scaled p;
15.  draw z1--z2--z3--z4--cycle; % kontur
16.  % klapka
17.  draw z4--z7{z7-z4}..{z3-z8}z8--z3;
18.  draw z1--z7; % lewa dolna przekątna
19.  draw z2--z8; % prawa przekątna
20.  labels(1,2,3,4,5,6,7,8);
21. endchar;
22.
23. end.
```

Linie 1–6 zawierają *parametry*: `golden_ratio`, `p#`, `size#`, `a.top`, `a.bot`, których konkretne wartości określają kształt koperty. Koperta będzie doskonała jeżeli długości jej boków pozostaną w złotej proporcji. Krótszy z boków prostokąta ma mieć długość `size#`, długość drugiego jest określona jako `size/golden_ratio`. Warto przyjrzeć się linii 4, w której w zwięzły sposób (typowy dla METAFONT-a) określono ten związek, i odpowiednie wartości przyporządkowano nowym zmiennym `eh#` i `ew#` (wysokość i długość koperty). Inną zasługującą na uwagę formułą jest czterokrotnie powtórzona `if unknown` *coś*`: ... fi`. W programie sztuczka ta jest użyta do sprawdzania czy odpowiednie zmienne zostały zdefiniowane i jeżeli nie zostały zdefiniowane to przyjmują one wartości standardowe. Tak więc jeżeli ktoś kiedyś będzie potrzebował np. koperty o wysokości 5 cm, wystarczy że utworzy pliczek `koperta5.mf` zawierający tylko dwie instrukcje:

```
size#:=5cm#;
input koperta;
```

Konstrukcja `if` *coś*`: ... fi` może być składnikiem instrukcji, o czym świadczy linia 3 (instrukcje są oddzielane średnikiem a w linii 3 go nie ma!). Jeżeli `p#` jest rzeczywiście nieznane to METAFONT przeczyta linie 3 i 4 jako: `20p# = golden_ratio * ew# = eh# = size#`. Parametr `p#` standardowo ma więc przyjąć wielkość równą jednej dwudziestej `size#`.

METAFONT output 1993.01.01.0000 Page 1 Character 48



Rysunek 1.

Skromne słówko `mode_setup` z linii 7 jest makrodefinicją, i to makrodefinicją tak ważną, że występującą w każdym programie metafontowym. Tam ustalana jest m.in. wartość zmiennej hppp (horizontal pixels per point). Idea jest następująca: parametry programu metafontowego powinny być określone w jednostkach „rzeczywistych” takich jak: punkty, milimetry, centymetry itd. W celu oznaczenia, że chodzi o wielkości rzeczywiste (Knuth nazywa je „sharp units”) piszemy po nich znak #. Gdzieś podczas wykonywania programu powinno nastąpić przekształcenie: coś:=coś# * hppp;. Następna instrukcja w tej samej linii jest przykładem tego działania. Nadaje ona wartość parametrowi p# „odpowiedzialnemu” za szerokość piórka. Wartość „pixelowa” powstała przez pomnożenie rzeczywistej przez hppp zostaje następnie zaokrąglona. Innymi słowy p jest wielkością piórka mierzoną w pikselach urządzenia na które generujemy kopertę. Jeśli np. urządzeniem tym jest drukarka o rozdzielczości 300 DPI to p=7 (p·hppp = 7,47198), dla fotonaświetlarki 1270 DPI, p=32 (31,63142) a dla drukarki o rozdzielczości 120 DPI p=3 (2,98882). Podane wartości odnoszą się do piórka o standardowej szerokości 1/20 size# = 20/36pt#.

Makrodefinicja beginchar w linii 8 definiuje cztery rzeczy — kod znaku (tutaj ma on numer 48 — pozycja zera w kodzie ASCII), szerokość (width), wysokość (height) oraz głębokość (depth). Wszystkie te wielkości powinny być podane w jednostkach rzeczywistych ponieważ makro beginchar tworzy zmienne w, h, d i nadaje im odpowiednie wartości „pixelowe”. W naszym przypadku 242, 149 i 0 pixeli.

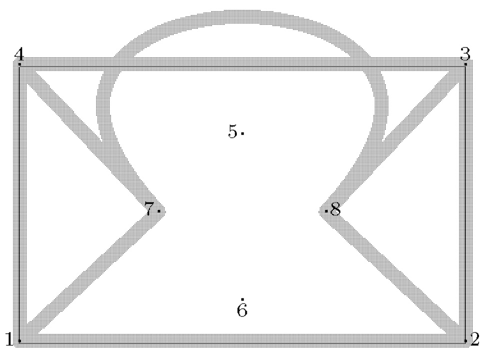
Zawartość linii 9 i 10 nie jest żadną tajemnicą. Po prostu punkty z1, z2, z3, z4 są zdefiniowane jako leżące w czterech wierzchołkach prostokąta o długości boków w i h. W programie zastosowano najbardziej zwięzłą notację „z-ową”, ale można je określać na kilka innych sposobów. Przykładowo punkt z1 z linii 9 możemy zdefiniować wykorzystując dowolny z poniższych zapisów:

```
(x1,y1)=(0,0)
x1=0; y1=0;
x1=y1=0;
```

Śledząc zapis w linii 11 dowiemy się po co były nam potrzebne parametry a.top i a.bot. Punkty 5 i 6 mają jednakże tylko charakter pomocniczy — posłużą do wyznaczenia dwóch brakujących dla wykreślenia koperty punktów.

Zapis z7=whatever[z1,z5] oznacza, że punkt z7 leży gdzieś na prostej określonej przez punkty z1 i z5. Ponieważ jednocześnie postulujemy, żeby tenże punkt z7 leżał na prostej z1--z6, jednoznacznie definiujemy go jako leżący na przecięciu tych prostych. Podobnie został określony punkt z8.

METAFONT output 1993.01.01.0001 Page 1 Character 48



Rysunek 2.

Zaczynamy rysować kopertę. Najpierw w linii 14 za pomocą makra, pickup pen *jakiś* określamy kształt piórka. Nasze ma kształt kolisty, ale nie zawsze tak musi być. Inne możliwości to na przykład pensquare (kwadrat, jak sama nazwa wskazuje) lub penrazor (piórko o szerokości jednego pixela i zerowej wysokości). Słówko scaled to jedna z możliwych transformacji jakimi możemy poddać piórko. pencircle scaled p oznacza użycie pióra o okrągłym końcu o średnicy p#. Skalowanie można także przeprowadzić „w poziomie” instrukcją xscaled lub „w pionie” używając yscaled. Gdyby potrzebne nam było pióro o eliptycznym końcu, moglibyśmy je wybrać przez:

```
pickup pencircle xscaled px yscaled py
```

W liniach 15–19 rysujemy odpowiednie fragmenty koperty (znak procentu podobnie jak w składni TEX-owej rozpoczyna komentarz). Słowo cycle wskazuje, iż narysowana krzywa będzie się kończyć w punkcie z którego wyruszyła i gwarantuje, że w punkcie tym krzywa przebiegać będzie gładko. Zapis z7{z7-z4}..{z3-z8}z8 oznacza, że życzymy sobie aby METAFONT rysował linię z punktu z7 w kierunku określonym przez wektor z7-z4, a z punktu z8 w kierunku określonym przez wektor z3-z8 (zapis z7{z4-z7}..{z8-z3}z8 to nie to samo — patrz rysunek 2). Łącząc punkty w instrukcji draw znakiem -- powodujemy wykreślanie odcinków, .. spowoduje wykreślanie krzywych. W programie koperta taka linia użyta jest tylko raz — do wykreślenia klapki. Zwróćmy uwagę, że oprócz dwóch punktów startowych i podania dwóch kierunków nie ingerowaliśmy w sposób kreślenia tej linii. METAFONT zrobił to za nas i zrobił to tak, że poprawienie go nie wydaje się łatwym zadaniem.

Słowo endchar kończy definiowanie znaku rozpoczęte makrem beginchar. Słowo end. kończy pracę METAFONT-a, jeżeli go nie napiszemy METAFONT nie zakończy pracy i zgłosi gotowość dalszego działania wyświetlając gwiazdkę (dokładnie tak samo zachowuje się TEX). Znaczenie makrodefinicji label wyjaśni się za chwilę.

Teraz potrzebujemy programu METAFONT.EXE oraz paru dodatkowych plików. Wszystko co będzie nam potrzebne znaleźć możemy np. w pakiecie emTEX. Utwóżmy zatem katalog np. \EMTEX\EMMF i skopiujmy do niego następujące pliki: MF286.EXE lub MF.EXE (wersja dla komputerów z procesorem 8086), GFTOPK.EXE, GFTODVI.EXE. Ponadto do nowo utworzonego katalogu np. \EMTEX\MFBASES skopiujmy PLAIN.BAS.

Podobnie jak TEX, METAFONT po uruchomieniu musi odnaleźć i wczytać plik zawierający podstawowe makroinstrukcje (w TEX-u taki plik nazywa się formatem a w METAFON-cie bazą — np. PLAIN.BAS, lub CM.BAS, używana do generowania czcionek Computer Modern).

W opisie kodu generującego kopertę często używałem określeń: makro, makrodefinicja. Chodziło o konstrukcje znajdujące się właśnie w bazie. Baza, podobnie jak format TEX-owy, jest skompilowana, w celu przyśpieszenia jej czytania. W postaci źródłowej znajduje się ona w pliku PLAIN.MF. Tam można zobaczyć co znaczą: beginchar, mode_setup itd. Nie jest to jednak z pewnością lektura dla początkujących METAFONT-owców.

Używając pakietu emTEX powinniśmy określić zmienną środowiskową MFBAS. Podanie na przykład: SET MFBAS=C:\EMTEX\MFBASES, wskazuje, że bazy powinien METAFONT szukać w katalogu
C:\EMTEX\MFBASES. Ponadto uruchamiając METAFONT-a powinniśmy określić na jakie urządzenie ma zostać wygenerowany font poprzez określenie odpowiedniej wartości zmiennej mode. Jeżeli generujemy kopertę na drukarkę laserową to możemy to zrobić tak:

```
mf286 \mode=hplaser; input koperta.mf
```

a dla drukarki mozaikowej epson-fx:

```
mf286 \mode=epsonfx; input koperta.mf
```

Otrzymamy po krótkiej chwili trzy pliki:
koperta.tfm, koperta.300 (lub koperta.240 dla drukarki epson-fx), koperta.log. Co zawierają pliki z rozszerzeniami .tfm i .log łatwo się domyśleć, plik koperta.300 (koperta.240) zawiera obraz naszej koperty, ale w postaci niedostępnej dla sterowników np. pakietu emTEX. Potrzebny jest jeszcze jeden program — GFTOPK.EXE, przekształcający pliki typu .GF do postaci .PK. Piszemy po prostu:

```
gftopk koperta.300
```

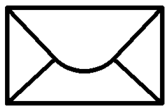
lub

```
gftopk koperta.240
```

I jeżeli program odpowie taką mniej więcej formułą:

```
This is GFtoPK, Version 2.2 [1g]
'METAFONT output 1993.01.01:0002'
516 bytes packed to 288 bytes.
```

oznacza to, że otrzymaliśmy plik koperta.pk, zawierający jeden znak, który jeżeli wszystko poszło dobrze powinien wyglądać mniej więcej tak:



Grupa Użytkowników Systemu TEX
Instytut Badań Systemowych
ulica Newelska 6
Warszawa

Jeżeli koperta ma być drukowana na innej drukarce niż HP Laser lub Epson-FX musimy zadeklarować odpowiednią wartość zmiennej mode. W pakiecie E. Mattesa dostępne są następujące możliwości (są one określone w pliku LOCAL.MF): hplaser (HP LaserJet+), epsonfx (Epson FX-80), epsonmx (Epson MX-80), lqhires (NEC-P6 rozdzielczość 360×360), lqmedres (LQ-1500, NEC-P6 360×180), lqmedresl (LQ-1500, NEC-P6 180×360), lqlores (LQ-1500, NEC-P6 180×180), itoh (C.ITOH 8510A rozdzielczość 160×144). kyocera (Kyocera F-1010),

Wreszcie powinniśmy mieć możliwość obejrzenia znaków na etapie ich projektowania. Służy do tego program GFTODVI.EXE, generujący powiększony znak wraz z zaznaczonymi punktami kontrolnymi. Program ten tworzy obraz znaku posługując się specjalnymi czcionkami: grayfont i slantfont. Zestaw tych czcionek jest dostarczony w pakiecie EMTEX, ale w postaci plików źródłowych: gray.mf, slant.mf, slantlj.mf, graylj.mf, grayfx.mf oraz slantfx.mf. Dla drukarki mozaikowej generowanie tych fontów wygląda następująco (dla laserowej używamy plików graylj i slantlj oraz nadajemy odpowiednią wartość zmiennej mode):

```
mf286 \mode=epsonfx; input slantfx.mf
```

a potem

```
gftopk slantfx.240
```

i już mamy font slantfx. Pliki slantfx.tfm oraz slantfx.pk kopiujemy do odpowiednich katalogów, z których korzysta nasz TEX. W analogiczny sposób generujemy font grayfx, po czym uruchomiamy METAFONT-a po raz trzeci, tym razem znowu z plikiem koperta.mf, ale bez nadawania jakiejkolwiek wartości zmiennej mode (co nie oznacza, że zmienna ta jest niezdefiniowana — if unknown mode ...):

```
mf286 \input koperta.mf
```

lub*

```
mf286 \nodisplays; \input koperta.mf
```

Otrzymamy plik koperta.260, z którego generujemy plik koperta.dvi pisząc**:

```
gftodvi koperta.260/ grayfont graylj
  slantfont slantlj labelfont plr8
  titlefont plr10 /
```

Uwaga: aby uniknąć kłopotów i mało sensownych komunikatów o błędach powinniśmy dokładnie odtworzyć powyższą linię. W szczególności po nazwie pliku podanej koniecznie z rozszerzeniem nie ma odstępu, a po znaku / i przed kończącym / jest! W wyniku działania programu otrzymamy plik koperta.dvi zawierający jedną stronę, a na niej znajomy rysunek. Oglądamy go na ekranie lub drukujemy tak jak każdy inny plik DVI. Szary kontur koperty został złożony czcionkami grayfont i slantfont. Słowo METAFONT za pomocą czcionki logo8, oznaczenia punktów zadeklarowanych uprzednio instrukcją label czcionką labelfont (plr8) a data i godzina titlefont (plr10). Wygenerowany plik jest podstawą do oceny wyglądu znaku i usuwania ewentualnych błędów, na przykład takich jak na rysunku 2 (Co tam się mogło stać?).

W ten sposób instalcja METAFONT-a została zakończona. Pozostaje tylko usprawnienie wywoływania poszczególnych programów, aby zabawa była rzeczywiście zabawą. Ja wykorzystuję prosty plik M.BAT, kolejno wywołujący MF.EXE, GFTODVI.EXE, GFTOPK.EXE, i edytor. Każdy z nas może wykonać taki plik samodzielnie, kierując się własnymi przyzwyczajeniami.

Na koniec pytanie. Jaki znak powstanie w wyniku uruchomienia następującego programu? (Podpowiedź: także symbol z kolekcji „pocztowo--telekomunikacyjnej").

```
% Co to jest?

ht#:=18pt#;
mode_setup; define_pixels(ht);
u:=1/50ht; % jednostka

beginchar(0,18pt#,12pt#,0);
x0=.5w;y0=0;
y1'=y1=0; x1+16.5u=x1'-16.5u=x0;
y2=y2'=5u;x2+17u=x2'-17u=x0;
y3=y3'=18u;x3+12u=x3'-12u=x0;
y4=y4'=22u;x4+9u=x4'-9u=x0;
y5=y5'=23u;x5+6u=x5'-6u=x0;
y6=y6'=25u;x6+9u=x6'-9u=x0;
y7=y7'=25.5u;x7+7u=x7'-7u=x0;
x8=x9+1u;x8'-x0=x0-x8;y8=y8'=y9+2u;
x9=x0-13u;x9'-x0=x0-x9;
y9=y9'=y10=y10'=y4;
x10=x9-12u;x10'-x0=x0-x10;
x11'-x0=x0-x11;x11=x10+2.5u;
y11=y11'=y10+7u;
x99=x0;y99=13u;% środek tarczy
x88=x0;y88=h;
x20=x10;x19=x9;y20=y19=y10-1u;
x21=.5[x19,x20];y21=y20-2.5u;
x20'=x10';x19'=x9';y20'=y19'=y10'-1u;
x21'=.5[x19',x20'];y21'=y20'-2.5u;

filldraw z1--z1'--z2'--z3'--z4'--
  z6'--z7'--z5'--z5--z7--
  z6--z4--z3--z2--cycle;
filldraw z6..z8..z9--z10{up}..
  z11..tension1.44..z88..
  tension1.44..z11'..{down}
  z10'--z9'..z8'..z6'..tension1.55
  ..cycle; % rączka
filldraw z20{down}..{right}
  z21{right}..{up}z19--cycle;
filldraw z20'{down}..{left}
  z21'{left}..{up}z19'--cycle;

erase filldraw fullcircle
  scaled 14u shifted z99;
labels(range 0 thru 99);
endchar;
bye
```

*: Posiadacze karty EGA lub lepszej mogą wypróbować obie postacie instrukcji.

**: Na monitorze bez problemu całość zmieści się w jednej linii.