
MetaFont neo doc Documentation

Release 0.1.2

Bertrand Bordage

Sep 27, 2017

Contents

| | | |
|----------|--------------------------------------|----------|
| 1 | Qu'est-ce que MetaFont ? | 3 |
| 1.1 | Historique | 3 |
| 1.2 | Principe de fonctionnement | 3 |
| 1.3 | Forces et faiblesses | 4 |
| 1.4 | Projets utilisant MetaFont | 4 |
| 1.5 | Exemples de fontes | 4 |
| 1.6 | Conversions | 5 |
| 2 | Tutoriel | 7 |
| 2.1 | Installation | 7 |
| 2.2 | Un premier caractère | 7 |

Contenu :

Qu'est-ce que MetaFont ?

Todo

Finir cette introduction.

Historique

MetaFont est un langage de programmation permettant de générer des [fontes de caractères](#). Il fut créé par le mathématicien et informaticien [Donald Knuth](#) en 1977 en même temps que [TeX](#), un logiciel de composition de documents depuis utilisé par une majeure partie de la communauté scientifique internationale.

See also:

[Code source du programme MetaFont](#)

Principe de fonctionnement

Jusqu'au XX^e siècle, les poinçons des caractères étaient réalisés dans des moules. Les centaines de moules d'une fonte n'étaient toutefois créés qu'après de longues étapes de calculs et dessins.

Qu'elle soit donc réalisée avec ou sans MetaFont, une fonte de caractères doit une grande partie de sa lisibilité à l'*uniformisation* entre ses caractères. En effet, une fonte dont le « e » n'aurait pas la même taille que le « a » pourrait vite devenir illisible.

Voici quelques uniformisations indispensables :

- forme
- hauteur
- épaisseur

- inclinaison (pour l’italique)
- diamètre des **empattements**

MetaFont propose une solution simple pour effectuer proprement ces uniformisations. L’idée est de considérer chacune des contraintes citées précédemment comme une variable. Ces variables peuvent être liées entre elles par des équations plus ou moins complexes. Cela permet de générer des fontes de différentes tailles (12, 14, 16, 18 **points**, etc), différents poids (léger, normal, semi-gras, gras, etc), différentes densités (condensé, normal, etc)...

Exemple simple : Pour plus de lisibilité, on souhaite que les tailles les plus petites de la fonte aient des caractères proportionnellement plus épais. Il faut donc créer une équation qui liera la taille à l’épaisseur, du type : $epaisseur = 0.5pt + \frac{hauteur}{60}$

C’est là la grande force de MetaFont. On ne conserve pas uniquement le rendu final ou même le résultat des nombreux calculs de fonte. On ne conserve que les équations ayant permis de mettre au point la fonte. MetaFont se charge de trouver les solutions des équations puis d’en faire une fonte utilisable.

À tout moment, on dispose donc du **code source** de la fonte. Ainsi, si on désire réduire l’inclinaison de l’italique, il suffit de changer un seul paramètre pour que le changement s’effectue sur tous les caractères de la fonte.

Forces et faiblesses

Forces

- Écrire un minimum d’équations pour un maximum de résultats
- Aucun bug connu (327,68 \$ à celui qui trouvera un bug)
- **Conservation de la procédure et non du résultat**
 - rigueur scientifique
 - modifier une étape modifiera toute la fonte en conséquence
 - pérennité

Faiblesses

- Documentation difficilement abordable
- Monochrome uniquement
- Requiert les talents de designer, mathématicien et informaticien

Projets utilisant MetaFont

- **MetaPost**
- **TeX** et ses variantes (**LaTeX**, **ConTeXt**...)
- **LilyPond**

Exemples de fontes

- **Computer Modern** de TeX
- **Feta** de LilyPond

Conversions

MetaFont → images matricielles

La **compilation** d'un code source de syntaxe MetaFont à l'aide du programme MetaFont (dont la commande est `mf`) permet d'obtenir une fonte **TeX**. Hélas, MetaFont et TeX datent d'une époque où les imprimantes vectorielles n'existaient pas. Le format de sortie est donc **matriciel** et non **vectoriel**.

La fonte **TeX** obtenue (extension `.2602gf`) n'est pas directement lisible. Pour l'ouvrir, avec *evince* par exemple, on la convertit avec `gftodvi`.

MetaFont → OpenType

La création d'une fonte **OpenType** à l'aide de MetaFont n'est hélas pas simple. Elle comporte de nombreuses étapes intermédiaires. Heureusement, quelques programmes permettent de simplifier ce travail :

- **METATYPE1** ([documentation](#)), script **Perl** par l'équipe polonaise JNS
- **mf2pt1**, script **Perl** par Scott Pakin
- **mftrace**, script **Python** par le créateur de LilyPond, Han-Wen Nienhuys

Le point commun de ces scripts est l'utilisation de **MetaPost**. MetaPost permet en effet de créer des dessins vectoriels à partir d'un code presque identique à MetaFont.

Le plus simple à utiliser est **mf2pt1**, c'est pourquoi je l'utiliserai tout au long de cette documentation. De plus, le projet LilyPond, pour lequel fut créé **mftrace**, utilise désormais ce script. LilyPond étant à ma connaissance le dernier projet utilisant MetaFont, il me paraît d'autant plus utile d'utiliser *mf2pt1*.

mf2pt1 génère une fonte au format **PFB**. Elle est directement utilisable sous Linux, mais pour des raisons de compatibilité et de possibilités, une conversion vers OpenType est indispensable. Elle peut être facilement effectuée avec **FontForge**.

Warning: *mf2pt1* ne gère pas toutes les possibilités de MetaFont. Hélas il gère mal les `penstroke` et comporte par ailleurs quelques bugs.

Todo

Finir de rédiger le tutoriel !

Installation

Système d'exploitation GNU/Linux recommandé

Sous Ubuntu : `sudo apt-get install texlive-binaries`

Un premier caractère

Je vous propose de passer directement à la pratique : voir le rendu d'un fichier de code MetaFont, puis comprendre point par point son contenu.

1. Enregistrez les lignes suivantes dans un fichier texte nommé `T_basic.mf` :

```
beginchar("T", 10pt#, 12pt#, 0);  
  
  z1 = (w/2, 0);  
  z2 = 1/2 [z3, z4];  
  z3 = (0, h);  
  z4 = (w, h);  
  
  pickup pencircle scaled 2pt;  
  
  draw z1  
    -- z2;  
  draw z3
```

```
-- z4;

labels(1, 2, 3, 4);

endchar;

end;
```

2. Compilez le fichier avec le programme MetaFont :

```
mf T_basic.mf
```

Cela permet de générer un fichier `T_basic.2602gf`. Il s'agit d'un fichier de fonte TeX, utilisable uniquement par TeX et ses variantes.

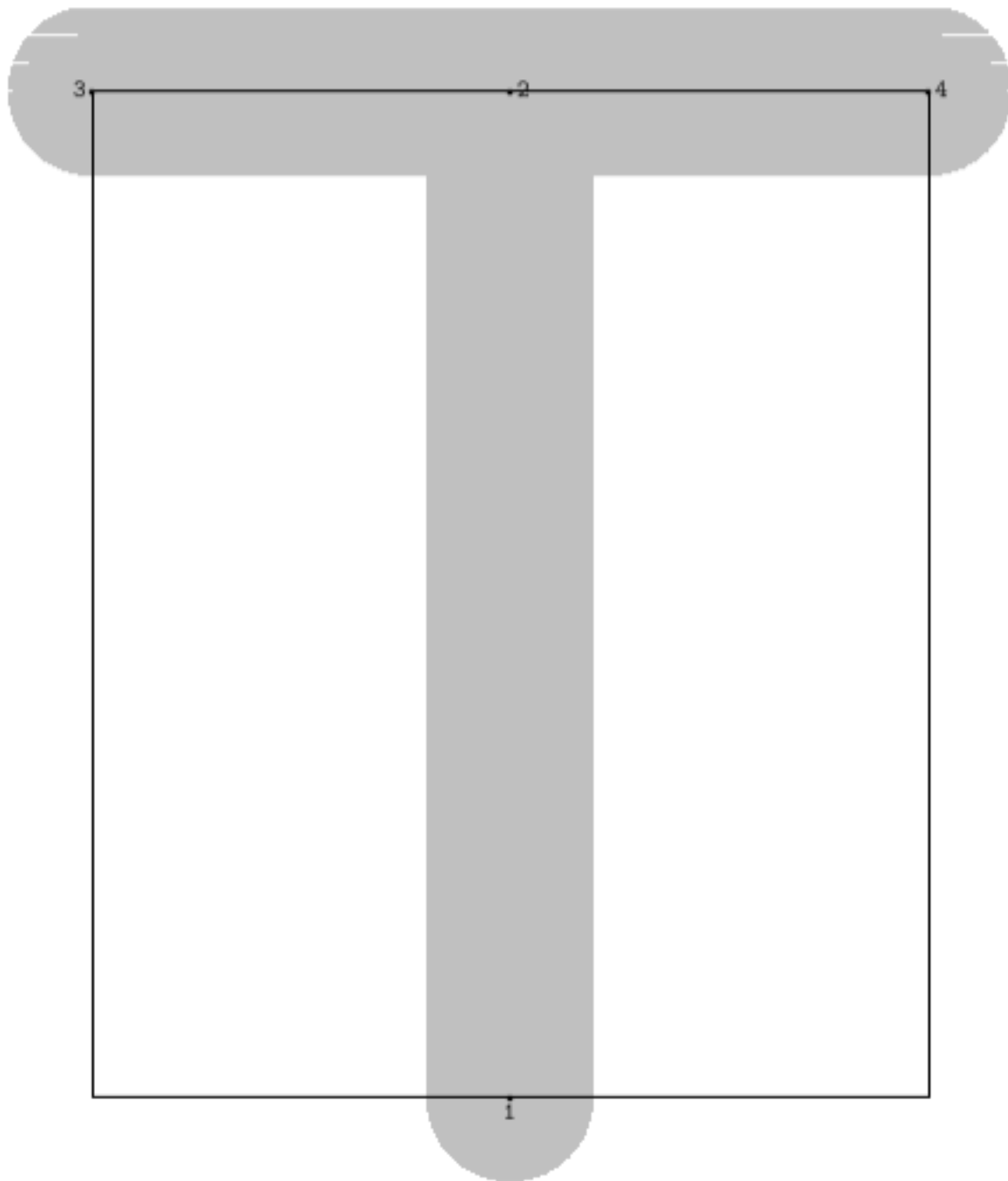
3. Convertissez le fichier `T_basic.2602gf` en DVI :

```
gftodvi T_basic.2602gf
```

Le fichier ainsi généré peut être ouvert avec *evince*.

Le résultat devrait ressembler à ceci :

METAFONT output 2012.04.29:1847 Page 1 Character 84



Les petits défauts d’affichage sont sans importance. Comme je l’expliquais dans le paragraphe *Conversions*, il s’agit d’un caractère matriciel. L’approximation qu’en fait *evince* est donc imparfaite. Ces défauts disparaîtront avec la conversion en fonte.