Utilisation de polices TrueType avec LaTeX

Dernière mise à jour de la page : 20.11.13

I. Les polices sous LaTeX

Cette petite introduction est inspirée du très bon ouvrage sur LaTeX de Vincent LOZANO que vous trouverez là (ouvrage sous Licence Art Libre). J'ajouterai qu'il est disponible dans la collection Framabook et qu'une version papier est en vente chez InLibroVeritas-Edition (je n'ai d'ailleurs pas résisté à la tentation de l'acheter).

Il faut savoir qu'une distribution LaTeX propose de base un certain nombre de polices. Pour conserver une homogénéité dans l'allure des caractères dans un document LaTeX, sont définies trois familles :

1. la famille *roman*
2. la famille *sans sérif*
3. la famille *machine à écrire*, également appelée *typewriter*

La police permettant de reconnaître un document LaTeX entre tous est la police Computer Modern et elle se décline selon ces 3 familles.

Dans toutes les distributions LaTeX qui se respectent, nous pourrons trouver le package **`times`** qui utilise :

1. la police *Times* pour la famille roman,
2. la police *Helvetica* pour la famille sans sérif,
3. et la police *Courier* pour la famille typewriter.

et vous aurez aussi le package **`newcent`** qui lui utilise :

1. la police *NewCentury* pour la famille roman,
2. la police *AvantGarde* pour la famille sans sérif,
3. et la police *Courier* pour la famille typewriter.

Il existe aussi les packages **`mathptmx`**, **`mathpazo`**...

Ces polices, ainsi qu'un grand nombre d'autres, peuvent être appelées ponctuellement sans charger de packages. Voici une petite liste de ce que l'on peut traditionnellement trouver dans une distribution LaTeX (cliquez sur l'image pour l'agrandir) :

identifiant	codage	résultat
cmr	T1	Computer Modern roman
cmss	T1	Computer Modern sans sérif
cmtt	T1	Computer Modern typewriter
cmfib	T1	Computer Modern fibonaci
cmfr	T1	Computer Modern funny roman
cmdh	T1	Computer Modern dunhil
lmr	T1	Latin Modern roman
lmss	T1	Latin Modern sans sérif
lmssq	T1	Latin Modern sans sérif expansé
lmtt	T1	Latin Modern typewriter
lmvtt	T1	Latin Modern typewriter proportionnel
ccr	T1	Concrete font
ygoth	U	Gothique
yfrak	U	Fraktur
yswab	U	Schwabacher
ptm	T1	Times
ppl	T1	Palatino
bch	T1	Charter
pnc	T1	New Century
pbk	T1	Bookman
phv	T1	Helvetica
pag	T1	Avantgarde
pcr	T1	Courier
pzc	T1	*Zapf Chancery*

Il y a 2 méthodes pour utiliser les polices en dehors de l'utilisation de packages. La 1ère est celle qu'utilisent justement les packages de polices. Il s'agit de remplacer les polices roman et/ou sans sérif et/ou typewriter pour tout le document, mais en choisissant soi-même ce que l'on souhaite pour une ou plusieurs familles, en tapant par exemple :

- **`\renewcommand{\rmdefault}{pnc}`** pour choisir New Century comme police roman
- **`\renewcommand{\sfdefault}{pag}`** pour choisir AvantGarde comme police sans sérif
- **`\renewcommand{\ttdefault}{pcr}`** pour choisir Courier comme police typewriter

Cet exemple est ce que fait le package **`newcent`** cité plus haut. Il suffit de placer ce genre de commande dans le préambule du document avec la police de son choix, sachant que par défaut, ce sont les variantes Computer Modern qui sont logiquement chargées.

La 2e méthode consite à appliquer la police uniquement pour un passage du document. La syntaxe est la suivante :

- **`{\fontfamily{ppl}\selectfont mon texte en Palatino dans cette exemple}`**

sachant qu'on peut utiliser plus d'options que le simple changement de police dans ce cas (**`\fontencoding`** pour le codage, **`\fontfamily{xxx}`** avec comme argument `xxx` la famille, **`\fontseries`** pour préciser la graisse, **`\fontshape`** pour l'allure de la police, **`\fontsize`** pour la taille...).

Ces polices utilisent :

- un codage qui sera à quelques exceptions près le codage T1
- une série de caractères identifiant la famille : `cmr` pour « computer modern roman », `ptm` pour « PostScript times", etc.
- une série de caractères désignant la « graisse » de la police, `m` pour « médium », `b` pour « bold » (gras), `bx` pour « bold extended » (gras étendu, c'est-à-dire gras avec des caractères plus larges), etc.
- une série de caractères définissant l'allure (shape en anglais) de la police : `n` pour « normal », `it` pour « italique », `sl` pour « slanted » (penché), etc.

Les fichiers des polices se trouvent dans le sous-répertoire `/fonts` de la distribution LaTeX ou de l'arborescence propre à l'utlisateur, et les différents packages correspondant se trouvent dans le sous-répertoire `/tex/latex` de la distribution LaTeX ou de l'arborescence propre à l'utlisateur.

Les types de fichiers que l'on peut observer sont :

`afm`	métriques Adobe de la police
`enc`	fichier décrivant le codage d'une police
`gf`	police au format bitmap non compressé
`mf`	fichier source d'une police vectorielle au format Metafont
`pfa`	police PostScript (ou Adobe) Type1 au format ASCII
`pfb`	police PostScript (ou Adobe) Type1 au format binaire
`pfm`	métiques PostScript de la police
`pk`	police au format bitmap compressé
`tfm`	métriques de la police (métriques les plus utilisées)
`ttf`	police TrueType
`vf`	police virtuelle
`vpl`	"virtual property list" (version lisible de vf)

Les polices déjà intégrées à LaTeX ont des éléments dans tous ces formats. Les appels sont gérés par des fichiers `*.fd` dans des sous-répertoires de `/tex/latex`.

II. À propos de la police Computer Modern

La famille de police standard de LaTeX est la police Computer Modern et son nom normalisé est `cmr`. L'origine même de cette police, si la qualité visuelle n'est plus à prouver, fait qu'elle était sous forme *bitmap*. Comme dans un premier temps LaTeX était essentiellement dédié à la production de documents au format PostScript, cela ne posait pas de problème. Cependant, LaTeX a évolué pour suivre l'apparition du

format PDF (que ce soit par la compilation `pdflatex` ou par la compilation `latex/dvips` suivie de la transformation `ps2pdf`) et dès lors le format initial de la police Computer Modern n'était plus adapté car le rendu de polices *bitmap* s'en trouve fortement dégradé. Le format PDF favorise en effet l'utilisation de polices dites *vectorielles*, dont le format privilégié est le format Type1 (Police PostScript de Type1).

La première étape pour l'amélioration de l'intégration de la police Computer Modern dans les documents PDF fut l'utilisation du Type3 (Police PostScript de Type3) mi-*bitmap* mi-*vectoriel* qui permit un rendu bien meilleur, mais toujours pas aussi bon que le *vectoriel* à proprement parler.

Est ensuite apparue une version Type1 de la police `cmr`. Mais cette conversion fit apparaître quelques difficultés qui ont la nécessité de clarifier les encodages utilisés pour les polices. Par défaut, LaTeX utilise un codage des caractères appelé OT1, qui conduit à l'inclusion de polices de Type1 dans les fichiers PDF. Le problème posé par le codage OT1 est double :

- d'une part, dans les fichiers PDF, les caractères accentués ne sont pas codés tels quels, mais par deux caractères superposés, car les accents n'existent pas en Computer Modern,
- d'autre part, LaTeX est incapable dans ces conditions d'effectuer des césures dans les mots accentués.

Pour contourner ces deux problèmes, la solution est aujourd'hui de passer au codage T1 avec l'appel du package **`\usepackage[T1]{fontenc}`**. Dans ce cas, `pdflatex` remplace la police Computer Modern `cmr` par une version actualisée dont le nom normalisé est `cm-super`. L'utilisation de cette police est transparente pour l'utilisateur, elle sera automiquement utilisée quand on passe au codage T1 lors de la conversion `dvips` ou de la compilation directe `pdflatex`. Cela est observable dans la liste des polices intégrées du document PDF créé : les noms `cmr` sont remplacés par les noms `sfrm` qui est la base de noms des fichiers de police Type1 de `cm-super`.

Enfin, depuis quelques années, d'autres travaux ont été réalisés pour améliorer le rendu des caractères diacritiques (notamment les accents et les cédilles en français, mais les diacritiques apparaissent dans de très nombreux alphabets). Ces travaux ont abouti à la création de la police **Latin Modern** qui peut donc être considérée comme une extension de la police Computer Modern pour les alphabets à caractères diacritiques. De plus, cette police est naturellement proposée au format Type1 pour les documents PDF. Pour l'utiliser, il faut charger le package **`\usepackage{lmodern}`**. Notez toutefois que cette police ne supporte pas l'encodage OT1.

III. Les polices TrueType

On a déjà pas mal de polices à notre disposition, mais que puis-je faire si je veux la police Arial dans mon document ? Il existe un package **`uarial`** disponible sur le CTAN et qui est directement intégré à la distribution MiKTeX (version 2.5 et suivantes) pour Windows (il n'existe pas de base dans toutes les distributions). Une autre solution est de *bricoler* la police Arial disponible au format TrueType dans Windows par exemple pour pouvoir l'utiliser dans LaTeX.

Il existe quelques tutoriels sur la toile comme http://www.radamir.com/tex/ttf-tex.htm ou encore http://zoonek.free.fr/LaTeX/Fontes/fontes.html#8. Ils sont relativement bien décrits, mais le gros reproche est que nous sommes bien souvent dans l'obligation de choisir entre des compilations `dvips` ou `pdflatex`. De plus, le résultat en terme de crénage par exemple est assez difficile à obtenir de manière claire... Quels sont les problèmes que l'on peut rencontrer en utilisant ces méthodes ?

- A la compilation dvips, les problèmes de crénages sont résolus, mais quand on fait un `ps2pdf`, on se retrouve avec des polices PK (bitmap donc) ce qui enlève tout l'intérêt du PDF car elles sont pixélisées.
- la compilation `pdflatex` directe, ce sont les polices TrueType qui sont directement intégrées dans le fichier PDF (à observer dans les propriétes du fichier, ce sont les fontes « embedded »), mais la version TrueType n'autorise pas dans ce cas les caractères spéciaux et surtout ne corrige pas le problème de crénage : voir http://zoonek.free.fr/LaTeX/Fontes/fontes.html pour l'explication des ligatures et du crénage; en 2 mots, c'est la gestion élégante des caractères, comme pour les suites de caractères fi, fl, ff, ffl, ffi (ligatures) ou encore les barres du V et des A dans AVAL (crénage).

L'idéal serait de pouvoir avoir des polices jolies que ce soit via `dvips/ps2pdf` ou `pdflatex`. Quand nous observons les polices « embarquées » dans les fichiers PDF avec les polices Computer Modern, nous pouvons constater qu'elles sont au format Type1, que l'on passe par `dvips/ps2pdf` ou `pdflatex`. L'idée est donc de trouver un processus permettant de construire toute l'arborescence de fichiers autorisant ça à partir d'une police TrueType.

Je propose une technique qui marche pas mal chez moi. Je prends pour exemple la police Arial, mais cela peut être décliné à toutes les polices TrueType de Windows ou autres origines,je l'ai fait pour la Times New Roman, la Courier New, la Verdana, etc (voir mes créations actuelles). A une exception près, la distribution LaTeX doit disposer de tous les binaires nécessaires (sauf peut-être teTeX qui est assez light comme distribution).

III.1. Récupération des polices TrueType

Nous pouvons trouver les polices TrueType dans `C:\Windows\Fonts` sous WinXP, `C:\WINNT\Fonts` sous Win2k ou encore quelquechose comme `/usr/share/fonts/` sous Linux (mais il y a moins de polices TrueType sous Linux, puisqu'un grand nombre de polices sont mises dans un autre format). Pour mon exemple, je prends :

- `arial.ttf` : Arial droite
- `ariali.ttf` : Arial italique
- `arialbd.ttf` : Arial grasse
- `arialbi.ttf` : Arial grasse italique

III.2. L'identifiant et la copie des TrueType

Je choisis un nom de 3 lettres qui sera la nom d'appel et la base des noms de tous les fichiers. Attention aux noms déjà existants !!! Ici, je prends par exemple `arl`. Et je renomme mes fichiers ttf (étape pas forcément nécessaire, mais au moins on ne se perd pas) :

```
$ cp arial.ttf arlr8a.ttf
$ cp ariali.ttf arlri8a.ttf
$ cp arialbd.ttf arlb8a.ttf
```

```
$ cp arialbi.ttf arlbi8a.ttf
```

Notons bien la construction de ces noms : `***r8a.ttf`, `***ri8a.ttf`, `***b8a.ttf`, `***bi8a.ttf`, on gardera la même logique jusqu'à bout (le « a » pourra changer).

III.3. Transformation au format Type1

Je transforme les fichiers ttf en polices Type1 (pfa et pfb). Pour cela j'utilise le binaire `ttf2pt1`, le seul non disponible dans les distributions. Vous le trouverez ici. Il suffit de placer correctement le binaire et ses fichiers associés dans l'arborescence de votre distribution LaTeX (pas dans le `texmf` local!!). Ca marche très bien avec MiKTeX sous Windows, et sous openSUSE 10.2 et 10.3, un package Linux le propose directement... Ceci fait, on tape les commandes :

```
$ ttf2pt1 -a -e arlr8a.ttf arlr8a   # pour créer le fichier arlr8a.pfa
$ ttf2pt1 -a -b arlr8a.ttf arlr8a   # pour créer le fichier arlr8a.pfb
```

Il faut faire de même avec `arlri8a.ttf`, `arlb8a.ttf` et `arlbi8a.ttf` en respectant les noms, soit :

```
$ ttf2pt1 -a -e arlri8a.ttf arlri8a
$ ttf2pt1 -a -b arlri8a.ttf arlri8a
$ ttf2pt1 -a -e arlb8a.ttf arlb8a
$ ttf2pt1 -a -b arlb8a.ttf arlb8a
$ ttf2pt1 -a -e arlbi8a.ttf arlbi8a
$ ttf2pt1 -a -b arlbi8a.ttf arlbi8a
```

III.4. Utilisation du package « fontinst »

On utilise ici le package **`fontinst.sty`** directement en ligne de commande. C'est cette étape qui va créer automatiquement l'essentiel des fichiers nécessaires. Tapez la commande dans une invite de commande MS-DOS sous Windows ou dans un Terminal sous Linux :

```
$ latex fontinst.sty
```

LaTeX va lancer une compilation en invite de commandes. Dès que le prompt s'affiche, cela signifie que LaTeX attent une instruction. Tapez alors dans ce prompt :

```
\latinfamily{arl}{} \bye
```

et laissez tourner. Ici j'ai pris `arl` comme argument, car c'est l'identifiant que j'ai choisi, à vous d'adapter selon votre choix, mais vous devez être cohérents, sinon plantage. Une grosse quantité de fichiers seront créés dans votre répertoire de travail, ne vous perdez pas et surtout n'effacez rien ici. Vous allez avoir des fichiers `*.pl`, `*.vpl`, `*.mtx`, `*.afm` et `*.fd`. Les fichiers `*.fd` consituent la gestion des appels que vous pourrez faire dans votre document LaTeX, mais on y reviendra plus tard. Sachez aussi que vous n'êtes pas obligés de disposer de toutes les déclinaisons de la police (droite, italique, grasse et grasse itlaique). **`fontinst.sty`** fera avec ce qu'il trouvera et c'est dans les fichiers `*.fd` que seront gérés les appels de remplacement (cf plus loin).

III.5. Métriques et polices virtuelles

Maintenant, il vous faut créer les métriques `tfm` et les polices virtuelles `vf`. Pour cela et pour chaque fichier `*.pl`, tapez :

```
$ pltotf nom_fichier.pl nom_fichier.tfm
```

en respectant les noms !! Seuls les extensions doivent changer. Pour les fichiers `*.vpl`, tapez :

```
$ vptovf nom_fichier.vpl nom_fichier.vf
```

C'est long car les fichiers sont très très nombreux. Je ne saurais trop vous conseiller d'écrire des scripts en shell (pour Linux) ou batch (pour Windows), sinon, vous allez vite en avoir marre, et vous risquez d'oublier l'un ou l'autre fichier.

III.6. Arborescence

Ici, on trie les fichiers. Tout d'abord, on efface les fichiers `*.vpl`, `*.pl` et `*.mtx`. On n'en a plus besoin. Ensuite, on va tout placer dans le `localtexmf` (en général `/home/nom_user/texmf` pour Linux). Vous pouvez aussi tout placer dans l'arborescence de la distribtion, mais je ne le conseille pas, c'est un tel bordel de fichiers que je ne m'y retouve pas avec mes propres créations. Dans mon exemple, je place (le `ms` est pour Microsoft, mais c'est un choix perso permettant de préciser l'origine des polices) :

- les fichiers `afm` dans `~/texmf/fonts/afm/ms/arial`
- les fichiers `tfm` dans `~/texmf/fonts/tfm/ms/arial`
- les fichiers `ttf` dans `~/texmf/fonts/truetype/ms/arial`
- les fichiers `pfa` et `pfb` dans `~/texmf/fonts/type1/ms/arial`
- les fichiers `vf` dans `~/texmf/fonts/vf/ms/arial`
- les fichiers `fd` dans `~/texmf/tex/latex/ms/arial`

III.7. Mapping

Il faut encore compléter le mapping des polices. Depuis quelques temps maintenant, c'est l'outil updmap qui gère ça. Il vous faut créer un fichier de mapping, par exemple `winfonts.map`. Personnellement, je le place dans `~/texmf/fonts/map/ms/`. Ce fichier contient le mapping suivant pour mon exemple :

```
arlr8r ArialMT " TeXBase1Encoding ReEncodeFont " <8r.enc <arlr8a.pfb
```

```
arlri8r Arial-ItalicMT " TeXBase1Encoding ReEncodeFont " <8r.enc <arlri8a.pfb
arlb8r Arial-BoldMT " TeXBase1Encoding ReEncodeFont " <8r.enc <arlb8a.pfb
arlbi8r Arial-BoldItalicMT " TeXBase1Encoding ReEncodeFont " <8r.enc <arlbi8a.pfb
```

Vous constatez ici la correspondance de nom. Pour créer ces lignes, vous pouvez le faire à la main, mais sinon, il est bon de s'aider des binaires `ttf2afm` et `afm2tfm`. Ces binaires sont prévus pour faire des conversions de polices. On ne les utilise pas pour ça ici puisqu'on a utilisé `ttf2pt1` et le package **`fontinst.sty`**. Mais on peut les utiliser pour créer des instructions de mapping. Ainsi :

```
$ ttf2afm -e 8r.enc -o arlr8a.afm arlr8a.ttf
$ afm2tfm arlr8a.afm -T 8r.enc arlr8r.tfm >> winfonts.map
```

qui donne dans `winfonts.map` la ligne :

```
arlr8r ArialMT " TeXBase1Encoding ReEncodeFont " <8r.enc
```

Il suffit de compléter la ligne dans `winfonts.map` pour obtenir :

```
arlr8r ArialMT " TeXBase1Encoding ReEncodeFont " <8r.enc <arlr8a.pfb
```

Attention, le fichier d'encodage `8r.enc`, que vous trouverez dans l'arborescence de votre distribution, doit être présent dans votre répertoire de travail. Attention aussi à ne pas pourrir vos autres fichiers de travail. Une possiblité est de faire cette étape pendant l'étape 2 et de virer les fichiers `afm` et `tfm` issus de ces commandes. Ces fichiers seront recréés proprement à l'étape 4.

Pour activer le mapping :

- avec MiKTeX 2.6 : il faut d'apport rafraîchir les paths, puis taper (dans une invite de commandes DOS)
  `\> initexmf --edit-config-file updmap`
  pour ouvrir le fichier `updmap.cfg`. Dans ce fichier, ajoutez la ligne :
  `Map winfonts.map`
  puis sauvez et quitez. Ensuite, tapez la commande
  `\> initexmf --mkmaps`
  pour lancer la mise à jour du mapping.
- avec TeX Live sous Linux (et sous Windows, j'ai testé récemment), il suffit de taper :
  `$ updmap --enable Map=winfonts.map`
  puis de taper la commande **`texhash`** ou **`texconfig rehash`**. Ca met à jour les paths, mais aussi le mapping.

Attention toutefois à bien gérer les `texmf` locaux... sinon, ça ne marche pas... Vous pouvez vérifier le mapping. Pour cela, vous devez retrouver les lignes de `winfonts.map` dans les fichiers `psfonts.map` de `dvips` et `pdftex`. Si ce n'est pas le cas, le mapping a planté.

III.8. Personnalisation des fichiers `*.fd`

Vous avez 4 fichiers `*.fd` créés. Dans mon exemple on a `8rarl.df`, `ot1arl.fd`, `ts1arl.fd` et `t1arl.fd`. C'est ce dernier qui nous intéresse, puisque l'on travaille la plupart du temps en codage T1 (vérifez le préambule de vos documents, vous trouverez sans doute **`\usepackage[T1]{inputenc}`**).

Voici ce qu'on a dans `t1arl.fd` :

```
%Filename: t1arl.fd
%Created by: tex fontinst
%Created using fontinst v1.928

%THIS FILE SHOULD BE PUT IN A TEX INPUTS DIRECTORY

\ProvidesFile{t1arl.fd}
   [2007/10/22 Fontinst v1.928 font definitions for T1/arl.]

\DeclareFontFamily{T1}{arl}{}

\DeclareFontShape{T1}{arl}{m}{n}{<-> \arL@@Scale arlr8t}{}
\DeclareFontShape{T1}{arl}{m}{sc}{<-> \arL@@Scale arlrc8t}{}
\DeclareFontShape{T1}{arl}{m}{sl}{<-> \arL@@Scale arlro8t}{}
\DeclareFontShape{T1}{arl}{m}{it}{<-> \arL@@Scale arlri8t}{}

\DeclareFontShape{T1}{arl}{b}{n}{<-> \arL@@Scale arlb8t}{}
\DeclareFontShape{T1}{arl}{b}{sc}{<-> \arL@@Scale arlbc8t}{}
\DeclareFontShape{T1}{arl}{b}{sl}{<-> \arL@@Scale arlbo8t}{}
\DeclareFontShape{T1}{arl}{b}{it}{<-> \arL@@Scale arlbi8t}{}

\DeclareFontShape{T1}{arl}{bx}{n}{<->ssub * arl/b/n}{}
\DeclareFontShape{T1}{arl}{bx}{sc}{<->ssub * arl/b/sc}{}
\DeclareFontShape{T1}{arl}{bx}{sl}{<->ssub * arl/b/sl}{}
\DeclareFontShape{T1}{arl}{bx}{it}{<->ssub * arl/b/it}{}

\endinput
```

Cela indique comment seront gérés les appels de la police (droite, grasse, italique, small caps, slanted....). Cela gère aussi les cas qui n'existent pas par remplacement comme ici les cas `bx` remplacés par les cas `b`. Il vous faut savoir que vous pouvez créer des polices mêmes sans avoir toutes les déclinaisons en TrueType. Ainsi l'Arial Black n'existe que droite et italique et on a (par exemple) :

```
%Filename: t1ark.fd
%Created by: tex fontinst
%Created using fontinst v1.928
```

```
4.
5. %THIS FILE SHOULD BE PUT IN A TEX INPUTS DIRECTORY
6.
7. \ProvidesFile{t1ark.fd}
8.    [2007/10/22 Fontinst v1.928 font definitions for T1/ark.]
9.
10. \DeclareFontFamily{T1}{ark}{}
11.
12. \DeclareFontShape{T1}{ark}{m}{n}{<-> \arK@@Scale arkr8t}{}
13. \DeclareFontShape{T1}{ark}{m}{sc}{<-> \arK@@Scale arkrc8t}{}
14. \DeclareFontShape{T1}{ark}{m}{sl}{<-> \arK@@Scale arkro8t}{}
15. \DeclareFontShape{T1}{ark}{m}{it}{<-> \arK@@Scale arkri8t}{}
16.
17. \DeclareFontShape{T1}{ark}{b}{n}{<->ssub * ark/m/n}{}
18. \DeclareFontShape{T1}{ark}{b}{sc}{<->ssub * ark/m/sc}{}
19. \DeclareFontShape{T1}{ark}{b}{sl}{<->ssub * ark/m/sl}{}
20. \DeclareFontShape{T1}{ark}{b}{it}{<->ssub * ark/m/it}{}
21.
22. \DeclareFontShape{T1}{ark}{bx}{n}{<->ssub * ark/m/n}{}
23. \DeclareFontShape{T1}{ark}{bx}{sc}{<->ssub * ark/m/sc}{}
24. \DeclareFontShape{T1}{ark}{bx}{sl}{<->ssub * ark/m/sl}{}
25. \DeclareFontShape{T1}{ark}{bx}{it}{<->ssub * ark/m/it}{}
26.
27. \endinput
```

Vous constatez que la version grasse n'est donc pas explicitement créée dans ce cas. Elle est remplacée ici par la version normale (non grasse donc). Vous avez toute liberté dans ces appels. Ainsi, vous aurez sans doutes des problèmes avec les polices *slanted* (penchées), il est intéressant de les remplacer par la version *italique* de la police :

```
\DeclareFontShape{T1}{arl}{m}{sl}{<-> arlri8t}{}
\DeclareFontShape{T1}{arl}{b}{sl}{<-> arlbi8t}{}
```

Vous pouvez faire le même genre de manipulations dans les autres fichiers `*.fd`.

III.9. Utilisation

Toujours avec l'exemple Arial (il est facile d'adapter à toutes vos créations) :

- pour choisir Arial comme police roman pour tout le document : **`\renewcommand{\rmdefault}{arl}`**
- pour choisir Arial comme police sans sérif pour tout le document : **`\renewcommand{\sfdefault}{arl}`**
- pour choisir Arial comme police typewriter pour tout le document : **`\renewcommand{\ttdefault}{arl}`**
- ou pour un appel local : **`{\fontfamily{arl}\selectfont mon texte en Arial}`**

III.10. Redimensionnement (optionnel)

Vous constaterez sans doute que les nouvelles polices issues du format TrueType sont un poil plus grandes que les autres polices de LaTeX. Une technique consiste à créer un package de redimensionnement. Pour cela, je me suis inspiré du package **`uarial`** (j'ai découvert que cette technique est utilisée pour d'autres polices proposées dans les distributions LaTeX, comme c'est le cas avec le package **`helvet.sty`**). Par exemple, vous pouvez créer dans le même répertoire de vos fichiers `*.fd`, le package **`ftarial.sty`** :

```
1. % package ftarial
2. % version écrite par pulsar68
3. % permet l'utilisation de la police Arial
4. %
5. % créé le 17/10/2007 sur la base du package uarial/helvet
6. % ATTENTION : Il ne s'agit que d'un package de réglage qui agit sur l'échelle
7. %             par défaut de la police.
8. %             Cela joue directement l'affichage des appels {arl}.
9. %             * pour agir sur la police par défaut, ajouter
10. %               \renewcommand{\rmdefault}{arl}
11. %             * pour agir sur la police sans serif par défaut, ajouter
12. %               \renewcommand{\sfdefault}{arl}
13. %             * pour agir sur la police TypeWriter par défaut, ajouter
14. %               \renewcommand{\ttdefault}{arl}
15.
16. \ProvidesPackage{ftarial}[2007/10/17 Arial font scale package]
17. \RequirePackage{keyval}
18. \define@key{ArL}{scaled}[.95]{%
19.   \expandafter\def\csname arl@Scale\endcsname{#1}}
20. \def\ProcessOptionsWithKV#1{%
21.   \let\@tempc\relax
22.   \let\ArL@tempa\@empty
23.   \ifx\@classoptionslist\relax\else
24.     \@for\CurrentOption:=\@classoptionslist\do{%
25.       \@ifundefined{KV@#1@\CurrentOption}%
26.       {}%
27.       {%
28.         \edef\ArL@tempa{\ArL@tempa,\CurrentOption,}%
29.         \@expandtwoargs\@removeelement\CurrentOption
30.           \@unusedoptionlist\@unusedoptionlist
31.       }%
32.     }%
33.   \fi
34.   \edef\ArL@tempa{%
35.     \noexpand\setkeys{#1}{%
36.       \ArL@tempa\@ptionlist{\@currname.\@currext}%
37.     }%
38.   }%
```

```
  \ArL@tempa
  \let\CurrentOption\@empty
}
\ProcessOptionsWithKV{ArL}
\AtEndOfPackage{%
  \let\@unprocessedoptions\relax
}
\renewcommand{\sfdefault}{arl}
\endinput
```

Il faut encore modifier vos fichiers `*.fd` en conséquence de la manière suivante (par exemple pour `t1arl.fd`) :

```
%Filename: t1arl.fd
%Created by: tex fontinst
%Created using fontinst v1.928

%THIS FILE SHOULD BE PUT IN A TEX INPUTS DIRECTORY

\ProvidesFile{t1arl.fd}
   [2007/10/22 Fontinst v1.928 font definitions for T1/arl.]

\expandafter\ifx\csname arl@Scale\endcsname\relax
 \let\arL@@Scale\@empty
\else
 \edef\arL@@Scale{s*[\csname arl@Scale\endcsname]}%
\fi

\DeclareFontFamily{T1}{arl}{}

\DeclareFontShape{T1}{arl}{m}{n}{<-> \arL@@Scale arlr8t}{}
\DeclareFontShape{T1}{arl}{m}{sc}{<-> \arL@@Scale arlrc8t}{}
%\DeclareFontShape{T1}{arl}{m}{sl}{<-> \arL@@Scale arlro8t}{}
\DeclareFontShape{T1}{arl}{m}{sl}{<-> \arL@@Scale arlri8t}{}
\DeclareFontShape{T1}{arl}{m}{it}{<-> \arL@@Scale arlri8t}{}

\DeclareFontShape{T1}{arl}{b}{n}{<-> \arL@@Scale arlb8t}{}
\DeclareFontShape{T1}{arl}{b}{sc}{<-> \arL@@Scale arlbc8t}{}
%\DeclareFontShape{T1}{arl}{b}{sl}{<-> \arL@@Scale arlbo8t}{}
\DeclareFontShape{T1}{arl}{b}{sl}{<-> \arL@@Scale arlbi8t}{}
\DeclareFontShape{T1}{arl}{b}{it}{<-> \arL@@Scale arlbi8t}{}

\DeclareFontShape{T1}{arl}{bx}{n}{<->ssub * arl/b/n}{}
\DeclareFontShape{T1}{arl}{bx}{sc}{<->ssub * arl/b/sc}{}
\DeclareFontShape{T1}{arl}{bx}{sl}{<->ssub * arl/b/sl}{}
\DeclareFontShape{T1}{arl}{bx}{it}{<->ssub * arl/b/it}{}

\endinput
```

Pour autoriser le redimensionnement, il vous suffira d'ajouter dans le préambule de votre document avant tout appel de la police `arl` (pour cet exemple) :

```
\usepackage[scaled=0.95]{msarial}
```

pour un redimensionnement de facteur 0.95. Des facteurs de 0.9 et 0.95 semblent de bons compromis (pour la police Courier new, il semble que 0.85 soit plus adaptée). Pour ma part, je prends en général 0.95. Tous les appels `arl` du document seront impactés. L'utilisation d'un tel package n'est pas obligatoire. Et si vous décidez de le créer et donc de modifier les fichiers `*.fd`, ceux-ci seront fonctionnels même sans l'appel du package.

Le package **`msarial.sty`** que nous avons créé accepte une valeur par défaut pour la *clé* `scaled` qui est 0.95. La valeur par défaut sera utilisée avec la syntaxe suivante :

```
\usepackage[scaled]{msarial}
```

est si aucune *clé* n'est précisée, la valeur 0 sera attribuée, ce qui a pour effet dans notre cas de ne pas redimensionner la police (comme si nous avions choisi la valeur 1).

Vous pouvez intégrer ça dans d'autres packages ou classes par la commande :

```
\RequirePackage[scaled=0.95]{msarial}
```

III.11. Quelques exemples

Voici ce que j'ai pu obtenir (ici sans package de redimensionnement, cliquez sur l'image pour l'agrandir) :

identifiant	codage	résultat
arl	T1	Arial
ark	T1	**Arial Black**
crn	T1	Courier New
tnr	T1	Times New Roman
grg	T1	Georgia
vdn	T1	Verdana
amt	T1	Albany AMT
cmt	T1	Cumberland AMT
tmt	T1	Thorndale AMT
dvr	T1	DejaVu Serif
dvs	T1	DejaVu Sans Serif
dvt	T1	DejaVu Sans Serif Mono
grd	T1	Garamond
grl	T1	Garamond Light
exo	T1	EXOCET (POUR LES AMATEURS DU JEU DIABLO)

C'est pas si mal, non ?

IV. Un script shell de conversion : ttf2tex

Je propose ici un petit script en shell permettant de faire la conversion. Vous pouvez le télécharger ici. Il regroupe toutes les étapes décrites plus haut. Cependant, il nécessite :

- une distribution LaTeX pour les binaires `ttf2afm`, `afm2tfm`, `latex`, `pltotf` et `vptovf`, et le package **`fontinst.sty`**,
- le binaire `ttf2pt1` disponible sur http://ttf2pt1.sourceforge.net et dans certaines distributions Linux,
- le fichier de codage `8r.enc` (fourni avec toute distribution LaTeX) doit être présent dans le répertoire de travail.

Ce script demande à l'utilisateur de renseigner le nom des fichiers sélectionnés en tant que polices droite, italique, grasse et grasse italique. Toutes ces polices ne sont pas obligatoires. Sont également demandés le nom de 3 lettres de la police créée (comme par exemple `arl`) servant à la construction du nom des différents fichiers utilisés par LaTeX et le nom du fichier de mapping associé.

Le fichier de mapping ne demande plus de modification manuelle avec la version actuelle du script, mais les fichiers `*.fd` créés demanderont encore des modifications à faire à la main par la suite pour personnaliser les appels de la police.

Tous les fichiers créés sont triés et placés dans différents répertoires pour faciliter leur repérage et leur déplacement dans le `texmf` voulu.

Enfin, ce script est un peu grossier mais fonctionnel (j'ai écrit sa première version à mes débuts en Shell sans vraiment le restructurer ou le simplifier et les mises à jour ne font qu'ajouter de nouvelles fonctionnalités...). Il pourra être amélioré, adapté, découpé selon les besoins de chacun.

Derniers updates : j'ai fait une grosse mise à jour du script. Le package de redimensionnement est maintenant construit automatiquement, et les fichiers `*.fd` sont également mis à jour automatiquement en fonction de celui-ci. Le script est plus difficile à lire. Il est sans doute possible de simplifier les utilisations de `sed` et de `awk`...

```
#!/bin/bash

#*******************************************************************************
# script ttf2tex
# écrit par pulsar68
#-------------------------------------------------------------------------------
# - conversion de polices TrueType aux formats compatibles LaTeX
# - nécessite une distribution LaTeX pour les binaires ttf2afm, afm2tfm,
#   latex, pltotf et vptovf, et le package fontinst.sty
# - nécessite le binaire ttf2pt1 disponible sur
#   http://ttf2pt1.sourceforge.net/ et dans certaines distributions Linux
# - le fichier d'encodage "8r.enc" doit etre présent dans le répertoire
#   courant
#
# il est possible de fixer la racine du nom de police des l'appel par :
#     ttf2tex xxx (avec xxx la racine voulue)
#
#-------------------------------------------------------------------------------
# 23/10/13 : - reprise de quelques structures de test
#            - réduction de la largeur des commentaires du fichier *.sty créé
# 04/02/11 : - annulation de répertoire commun myfonts
# 10/12/10 : - annulation de l'origine de la police (lx/ms remplaces par ft
#              pour les noms de packages, répertoire commun myfonts)
# 13/01/10 : - option "-p ttf" retirée pour ttf2pt1
# 02/04/09 : - prise en compte de l'origine pour la création de
#              l'arborescence et du package de redimensionnement
# 12/02/09 : - ajout de l'invite pour un nom de police complet à afficher
#              dans les commentaires du package de redimensionnement
#              (purement cosmétique)
#            - mise a jour des différents messages
# 11/02/09 : - construction automatique du package de redimensionnement
#            - update automatique des fichiers *.fd pour compatibilite avec
#              le package de redimensionnement
#            - gestion des sources ttf manquantes pour fichiers *.fd
# 04/10/08 : - correction du texte de prompt pour le mapping
#            - les fichiers finaux sont mis dans des sous-répertoires selon
#              un choix de nom de l'utilisateur
#            - meilleure mise en forme de fichier de mapping
# 14/06/08 : - écriture d'une aide (mise a un standard personnel)
#            - ajout d'une option [-h] pour l'aide
# 09/12/07 : - création
#*******************************************************************************

ERR_FILENOTFOUND=67
ERR_AIDE=72
nomscript=$(basename $0)
nomorigine=ft

#-------------------------------------------------------------------------------
# fonction affichant le message d'aide
#-------------------------------------------------------------------------------
affiche_aide()
{
cat <<affaide

  Conversion de polices TrueType aux formats compatibles LaTeX
  - nécessite une distribution LaTeX pour les binaires ttf2afm, afm2tfm,
    latex, pltotf et vptovf, et le package fontinst.sty
  - nécessite le binaire ttf2pt1 disponible sur http://ttf2pt1.sourceforge.net/
    et dans certaines distributions Linux
  - le fichier d'encodage "8r.enc" doit etre présent dans le répertoire courant
```

```
  Appel:

        $nomscript [xxx]

  Options:

        -h  : affiche ce texte d'aide
        xxx : racine de noms en 3 lettre voulue pour les fichiers à créer

affaide
}

#-------------------------------------------------------------------------------
# main()
#-------------------------------------------------------------------------------
[ "$1" = "-h" ] && { affiche_aide; exit $ERR_AIDE; }

[ ! -f "8r.enc" ] && { echo "le fichier 8r.enc est manquant";\
                       exit $ERR_FILENOTFOUND; }

echo
echo "Listes des fichiers ttf du repertoire courant :"
echo
ls -1 *.ttf
echo

[ -z "$1" ] && { echo -n "entrez la racine de nom (= 3 lettres) : ";\
                 read racine; }\
            || racine=$1
echo "les noms de fichiers seront construits selon la racine : $racine"

r8a=r8a;r8r=r8r;
ri8a=ri8a;ri8r=ri8r;
b8a=b8a;b8r=b8r;
bi8a=bi8a;bi8r=bi8r;

echo
echo -n "nom du fichier de police droite          : "
read fonter
echo -n "nom du fichier de police italique        : "
read fonteri
echo -n "nom du fichier de police grasse          : "
read fonteb
echo -n "nom du fichier de police grasse italique : "
read fontebi
echo
echo -n "police droite choisie          : $fonter"
if [ -f "$fonter" ]
then
  cp $fonter $racine$r8a.ttf
  echo -e "\tOK"
else
  echo -e "\tLa police droite doit etre disponible !!"
  exit $ERR_FILENOTFOUND
fi
echo -n "police italique choisie        : $fonteri"
if [ -f "$fonteri" ]
then
  cp $fonteri $racine$ri8a.ttf
  echo -e "\tOK"
else
  cp $racine$r8a.ttf $racine$ri8a.ttf
  echo -e "\tpas dispo"
fi
echo -n "police grasse choisie          : $fonteb"
if [ -f "$fonteb" ]
then
  cp $fonteb $racine$b8a.ttf
  echo -e "\tOK"
  nongras=0
else
  cp $racine$r8a.ttf $racine$b8a.ttf
  echo -e "\tpas dispo"
  nongras=1
fi
echo -n "police grasse italique choisie : $fontebi"
if [ -f "$fontebi" ]
then
  cp $fontebi $racine$bi8a.ttf
  echo -e "\tOK"
else
  [ -f "$fonteb" ] && { cp $racine$b8a.ttf $racine$bi8a.ttf; } \
                   || { cp $racine$ri8a.ttf $racine$bi8a.ttf; }
  echo -e "\tpas dispo"
fi
echo
echo "------------------------------------------------------------"

echo
ficmapping=$racine.map
echo " ... creation du fichier de mapping $ficmapping ..."
if [ -f "$racine$r8a.ttf" ]
then
  ttf2afm -e 8r.enc -o $racine$r8a.afm $racine$r8a.ttf
```

```
157.   afm2tfm $racine$r8a.afm -T 8r.enc $racine$r8r.tfm >> $ficmapping
158.   echo "<$racine$r8a.pfb" >> $ficmapping
159. fi
160. if [ -f "$racine$ri8a.ttf" ]
161. then
162.   ttf2afm -e 8r.enc -o $racine$ri8a.afm $racine$ri8a.ttf
163.   afm2tfm $racine$ri8a.afm -T 8r.enc $racine$ri8r.tfm >> $ficmapping
164.   echo "<$racine$ri8a.pfb" >> $ficmapping
165. fi
166. if [ -f "$racine$b8a.ttf" -a -f "$fonteb" ]
167. then
168.   ttf2afm -e 8r.enc -o $racine$b8a.afm $racine$b8a.ttf
169.   afm2tfm $racine$b8a.afm -T 8r.enc $racine$b8r.tfm >> $ficmapping
170.   echo "<$racine$b8a.pfb" >> $ficmapping
171. fi
172. if [ -f "$racine$bi8a.ttf" -a -f "$fonteb" ]
173. then
174.   ttf2afm -e 8r.enc -o $racine$bi8a.afm $racine$bi8a.ttf
175.   afm2tfm $racine$bi8a.afm -T 8r.enc $racine$bi8r.tfm >> $ficmapping
176.   echo "<$racine$bi8a.pfb" >> $ficmapping
177. fi
178. # joint les paires de lignes ensemble cote-a-cote
179. sed '$!N;s/\n/ /' $ficmapping > $ficmapping.tmp
180. mv $ficmapping.tmp $ficmapping
181. rm *.afm *.tfm
182. echo
183. echo "------------------------------------------------------------"
184.
185. echo
186. if [ -f "$racine$r8a.ttf" ]
187. then
188.   ttf2pt1 -a -e $racine$r8a.ttf $racine$r8a
189.   ttf2pt1 -a -b $racine$r8a.ttf $racine$r8a
190. fi
191. if [ -f "$racine$ri8a.ttf" ]
192. then
193.   ttf2pt1 -a -e $racine$ri8a.ttf $racine$ri8a
194.   ttf2pt1 -a -b $racine$ri8a.ttf $racine$ri8a
195. fi
196. if [ -f "$racine$b8a.ttf" ]
197. then
198.   ttf2pt1 -a -e $racine$b8a.ttf $racine$b8a
199.   ttf2pt1 -a -b $racine$b8a.ttf $racine$b8a
200. fi
201. if [ -f "$racine$bi8a.ttf" ]
202. then
203.   ttf2pt1 -a -e $racine$bi8a.ttf $racine$bi8a
204.   ttf2pt1 -a -b $racine$bi8a.ttf $racine$bi8a
205. fi
206. echo
207. echo "------------------------------------------------------------"
208.
209. echo
210. echo " ... lancement de latex fontinst.sty ..."
211. echo "entrer une commmande de type \latinfamily{$racine}{} \bye"
212. echo
213. latex fontinst.sty
214. rm fontinst.log
215. echo
216. echo "------------------------------------------------------------"
217.
218. echo
219. echo " ... conversion de PL en TFM ..."
220. for ficpl in *.pl; do pltotf $ficpl; done
221. echo
222. echo
223. echo " ... conversion de VPL en VF ..."
224. for ficvpl in *.vpl; do vptovf $ficvpl; done
225. echo
226. echo "------------------------------------------------------------"
227.
228. echo
229. echo -n "nom generique des repertoires de la police (arial par exemple): "
230. read nomrep
231. echo -n "nom complet de la police pour commentaires (Arial par exemple): "
232. read nompolice
233. echo
234.
235. racinE=`echo $racine | sed 's#.$#\U&#'`
236. RacinE=`echo $racine | sed 's#^.#\U&#' | sed 's#.$#\U&#'`
237. Nomrep=`echo $nomrep | sed 's#^.#\U&#'`
238. ladateen=`date +%Y/%m/%d`
239. ladatefr=`date +%d/%m/%Y`
240.
241. echo " ... creation du package de redimensionnement $nomorigine$nomrep.sty ..."
242. cat > $nomorigine$nomrep.sty <<EOF
243. % package $nomorigine$nomrep
244. % version écrite par pulsar68
245. % permet l'utilisation de la police $nompolice
246. %
247. % crée le $ladatefr sur la base du package uarial/helvet
248. % ATTENTION : Il ne s'agit que d'un package de réglage qui agit sur l'échelle
249. %             par défaut de la police.
250. %             Cela joue directement l'affichage des appels {$racine}.
251. %             * pour agir sur la police par défaut, ajouter
```

```
%              \renewcommand{\rmdefault}{$racine}
%            * pour agir sur la police sans serif par défaut, ajouter
%              \renewcommand{\sfdefault}{$racine}
%            * pour agir sur la police TypeWriter par défaut, ajouter
%              \renewcommand{\ttdefault}{$racine}

\ProvidesPackage{$nomorigine$nomrep}[$ladateen $Nomrep font scale package]
\RequirePackage{keyval}
\define@key{$RacinE}{scaled}[.95]{%
  \expandafter\def\csname $racine@Scale\endcsname{#1}}
\def\ProcessOptionsWithKV#1{%
  \let\@tempc\relax
  \let\$RacinE@tempa\@empty
  \ifx\@classoptionslist\relax\else
    \@for\CurrentOption:=\@classoptionslist\do{%
      \@ifundefined{KV@#1@\CurrentOption}%
      {}%
      {%
        \edef\$RacinE@tempa{\$RacinE@tempa,\CurrentOption,}%
        \@expandtwoargs\@removeelement\CurrentOption
          \@unusedoptionlist\@unusedoptionlist
      }%
    }%
  \fi
  \edef\$RacinE@tempa{%
    \noexpand\setkeys{#1}{%
      \$RacinE@tempa\@ptionlist{\@currname.\@currext}%
    }%
  }%
  \$RacinE@tempa
  \let\CurrentOption\@empty
}
\ProcessOptionsWithKV{$RacinE}
\AtEndOfPackage{%
  \let\@unprocessedoptions\relax
}
%\renewcommand{\rmdefault}{$racine}
\endinput
EOF

for codage in t1 ot1 ts1 8r
do
  echo " ... update du fichier $codage$racine.fd ..."
  echo "        - ajout des elements du package de redimensionnement"
  sed 's/^[ \t]*//' $codage$racine.fd \
  | sed -e :a -e '$!N;s/\n<->/<-> \'"$racinE"'@@Scale/;ta' -e 'P;D' \
  | sed -e :a -e '$!N;s/\n}{}/}{}/;ta' -e 'P;D' \
  | sed -e :a -e '/fd}$/N; s/fd}\n/fd}/; ta' > _ttf2tex_temp
  awk '/%Filename/,/ProvidesFile/ {print $0}' _ttf2tex_temp > _step1
  cat > _step2 <<EOF

\expandafter\ifx\csname $racine@Scale\endcsname\relax
 \let\$racinE@@Scale\@empty
\else
 \edef\$racinE@@Scale{s*[\csname $racine@Scale\endcsname]}%
\fi

EOF
  echo "        - remplacement de slanted (bug) par italic"
  awk '/DeclareFontFamily/,EOF {print $0}' _ttf2tex_temp \
  | sed '/^$/d' \
  | awk '{if (/{sl}{<-> /) {sub("o8","i8",$0);sub("o7","i7",$0);printf "%s\n%s\n","%"$0,$0} else {print $0}]
  | awk '{if (/^%/) {sub("i8","o8",$0);sub("i7","o7",$0);print $0} else {print $0}}' \
  | awk '{if (/{n}{<->/) {printf "\n%s\n",$0} else {print $0}}' \
  | awk '{if (/endinput/) {printf "\n%s\n",$0} else {print $0}}' > _step3
  cat _step* > $codage$racine.fd
  if [ $nongras -eq 1 ]
  then
    echo "        - gestion de la police grasse (source TrueType manquante)"
    sed '/^%\DeclareFontShape{.*}{.*}{b}{sl}/d' $codage$racine.fd \
    | awk '{if (/{b}{n}/) {sub("<->.*","<->ssub * nomtemp/m/n}{}",$0);print $0} else {print $0}}' \
    | awk '{if (/{b}{sc}/) {sub("<->.*","<->ssub * nomtemp/m/sc}{}",$0);print $0} else {print $0}}' \
    | awk '{if (/{b}{sl}/) {sub("<->.*","<->ssub * nomtemp/m/sl}{}",$0);print $0} else {print $0}}' \
    | awk '{if (/{b}{it}/) {sub("<->.*","<->ssub * nomtemp/m/it}{}",$0);print $0} else {print $0}}' \
    | sed -e 's#nomtemp#'"$racine"'#' -e 's#'"$racine"'/b#'"$racine"'/m#' > _ttf2tex_temp
    mv _ttf2tex_temp $codage$racine.fd
  fi
  rm _step* _ttf2tex_temp 2>/dev/null
done
echo

if [ $nongras -eq 1 ]
then
  rm "$racine"b*
fi

echo " ... tri des fichiers ... "
echo "        - suppression des fichiers *.vpl, *.pl et *.mtx"
rm *.vpl *.pl *.mtx
echo "        - fichiers afm dans ./fonts/afm/$nomrep/"
mkdir -p fonts/afm/$nomrep; mv *.afm fonts/afm/$nomrep/
echo "        - fichiers tfm dans ./fonts/tfm/$nomrep/"
mkdir -p fonts/tfm/$nomrep; mv *.tfm fonts/tfm/$nomrep/
echo "        - fichiers ttf dans ./fonts/truetype/$nomrep/"
mkdir -p fonts/truetype/$nomrep; mv $racine*.ttf fonts/truetype/$nomrep/
```

```
347. echo "        - fichiers pfa et pfb dans ./fonts/type1/$nomrep/"
348. mkdir -p fonts/type1/$nomrep; mv *.pf* fonts/type1/$nomrep/
349. echo "        - fichiers vf dans ./fonts/vf/$nomrep/"
350. mkdir -p fonts/vf/$nomrep; mv *.vf fonts/vf/$nomrep/
351. echo "        - fichiers fd et sty dans ./tex/latex/fonts/$nomrep/"
352. mkdir -p tex/latex/fonts/$nomrep; mv *.fd *.sty tex/latex/fonts/$nomrep/
353. echo "        - fichiers de mapping dans ./fonts/map/"
354. mkdir -p fonts/map; mv *.map fonts/map/
355. echo
356.
357. echo "done"
358. exit 0
```

contact : webmaster [at] pulsar68 [dot] org
(remplacer [at] par @ et [dot] par un point)