Cahiers GUTenberg

COMMENT PEUT-ON PERSONNALISER L'EXTENSION FRENCH DE LaTeX ?

Bernard Gaulle

Cahiers GUTenberg, nº 28-29 (1998), p. 143-157.

<http://cahiers.gutenberg.eu.org/fitem?id=CG_1998___28-29_143_0>

Comment peut-on personnaliser l'extension *french* de LaTeX ?

Bernard Gaulle

GUTenberg, vice-président

Résumé. L'extension *french* pour LaTeX comporte une grande variété d'options de base qui permettent de la personnaliser selon l'usage désiré. Cela peut être effectué à divers moments dans le document, définitivement ou temporairement. Certains paramètres agissent finement au niveau macro (comme la mise en page) ou micro-typographique (comme l'espacement autour de la ponctuation). L'ajout de nouvelles fonctionnalités, le mélange de styles et même la définition de nouvelles langues ou patois sont des choses possibles.

Cet article décrit ainsi les nombreux moyens de personnaliser l'extension *french* dans un contexte personnel ou dans celui d'un travail en groupe.

Abstract. *The French package for LaTeX present users with a large number of basic options which they can customise to suit their exact requirements. This customisation can be performed at various points in the document, and can be temporary or permanent. Some parameters affect the macrotypography of the document (such as page layout), whilst others are relevant to the microtypography (such as spacing around punctuation). Possible actions are, for example, to add new functionality, to mix styles and even to define new languages or dialects.*

This article describes the numerous ways of customising the French package, either for personal use or as part of a workgroup.

1. La francisation de LaTeX

Lorsque l'on souhaite s'attaquer à la francisation de LaTeX[5, 6, 7] il faut avant toute chose se rappeler que cela peut être effectué à plusieurs niveaux et que plus on désire obtenir une francisation poussée plus il est nécessaire d'intervenir au préalable sur l'installation.

1.1. Plusieurs choix préalables

Avant LaTeX il y a TeX sans lequel LaTeX ne serait rien. Il est donc souhaitable de pouvoir choisir son moteur TeX, de savoir s'il est capable de profiter au maximum des caractères 8-bits et de décider si l'on utilise ou pas l'option multilingue MlTeX.

Pour une bonne francisation du *format* LaTeX tous les moteurs TeX peuvent convenir dès l'instant où ils disposent des deux dispositifs que l'on vient de citer :

1° L'extension MlTeX qui permet d'utiliser librement les caractères accentués 8-bits avec les fontes *computer modern* (*cm*) tout en conservant la capacité de couper correctement les mots.

2° La possibilité d'écrire sur un fichier (ou sur la console) avec des caractères 8-bits, plutôt que de les traduire, comme cela est fait par TeX de façon standard, en leur équivalent hexadécimal `^^xx`, tout particulièrement illisible.

Il est toujours possible d'utiliser un moteur TeX de base, sans ces deux dispositifs, mais dans ce cas il est impératif de disposer de fontes *ec* disposant des caractères accentués français. MlTeX et l'extension 8-bits restent néanmoins des dispositifs complémentaires dont il ne faut pas se priver si l'on souhaite disposer d'une francisation maximale.

1.2. Choix à la création du *format*

Lorsque l'on crée le *format* LaTeX trois composants interviennent :

- le multilinguisme (chargement du fichier `hyconfig.tex`),
- les motifs de césure (chargement du fichier `frhyph.tex`),
- l'extension multi-claviers (chargement du fichier `kbconfig.tex`).

Ces choix sont à préciser dans le fichier de configuration de LaTeX (`hyphen.cfg`) où l'on trouve par exemple :

```
%\fontencoding{T1}\selectfont% hors option MlTeX,
\input kbconfig % pour les claviers,
\input hyconfig % pour le multilinguisme et la césure.
```

Pour utiliser les motifs de césure du français courant (`frhyph.tex`) il faut avoir paramétré correctement `hyconfig` par le biais du fichier de configuration `language.dat` dont voici un exemple :

```
%==============================================================================%
%| Name     | patterns file | exceptions file % comments                     |%
%==============================================================================%
  usenglish  ushyphen.tex   % this is the original default TeX language...
                            % Don. Knuth's hyphen.tex file can be used instead
   =english    enhyph.tex     enhyphex.tex    % default language is "english"
     french    frhyph.tex     frhyphex.tex    % ici on parle francais
    =patois %                                 % et ses patois...
   dumylang  dumyhyph.tex   % for testing a new language.
nohyphenation zerohyph.tex  % a language with no patterns at all.
%==============================================================================%
```

On voit dans cet exemple que le fichier des motifs français de césure est chargé pour le langage *french*. On y trouve aussi un fichier d'exceptions qui sera utilisé ultérieurement.

Il est possible d'utiliser la partie multilingue de Babel à la place de `hyconfig` et dans ce cas, le même fichier `language.dat` peut être utilisé et donc les mêmes motifs de césure. Mais ce n'est pas le choix préféré de l'auteur de cet article... car Babel nest pas le support préféré de *french* et ce depuis sa conception. Ainsi, malgré de nombreux efforts d'intégration, certains dispositifs de *french* ne peuvent fonctionner avec Babel. Nous verrons plus loin une extension (*mlp*) conçue (notamment) pour intégrer *french* qui est un meilleur garant de bon fonctionnement de la francisation.

En utilisant un clavier on génère des codes qui doivent être compris et éventuellement interprètés correctement par TeX. Le fichier de configuration à modifier pour indiquer le codage utilisé en entrée s'appelle `keyboard.dat` ; voici son contenu par défaut :

```
% Give below the encoding you want use for your input:
\kbencoding{decmulti}% "decmulti" is covering a wide range of keyboards.
%\kbencoding{ascii}% the most reduced input character set.
%%%%%%%%%%%%%%%%%%%%% (choose among xxx.kbc files)
```

2. Choix stratégiques pour la composition

L'extension *french* peut être utilisée dans des configurations variées.

Le *format* n'est pas francisé

Au cas, fort malencontreux, où l'usager LaTeX est contraint d'utiliser un *format* LaTeX non francisé [1] il n'est plus possible d'introduire les motifs français de

1. Nos amis étrangers sont rarement motivés pour refaire leur format LaTeX afin de rajouter un langage nouveau ; cela se comprend aisément, d'ailleurs n'agissons-nous pas de la sorte ?

césure et l'on est donc dans l'impossibilité d'avoir des coupures automatiques correctes pour les mots français. Il ne reste plus que la solution manuelle en intervenant sur chaque mot coupé de façon suspecte. Autre souci majeur de l'usager : l'extension *french* refuse délibérément de fonctionner car la coupure de mots est considérés comme un *minimum vital* indispensable à tout document composé en français. Il reste pourtant à l'usager la solution « du pauvre ». En effet, en faisant appel à l'extension *pmfrench* (`\usepackage{pmfrench}`) il est possible d'activer quelques dispositifs de francisation. Toutefois les fonctions suivantes restent inutilisables :

- la césure française déjà citée (et les exceptions),
- les abréviations françaises,
- les lettrines,
- les guillemets de deuxième niveau (« ... « *xxx* » ... »).

Il faut ajouter à cela que l'extension *pmfrench* n'est pas, pour l'instant, accessible en tant qu'option de Babel.

Babel est utilisé

Lorsque Babel est utilisé pour ses fonctions multilingues ou pour les langages supportés, il est possible de faire appel à l'extension *french*, sous réserve d'avoir installé préalablement tout le nécessaire dont notamment le fichier `french.ldf`. En codant : `\usepackage[french]{babel}` l'extension *french* sera chargée en tant qu'option de Babel directement exécutable. Cela signifie que le « code » de *french* dans son intégralité est activé à ce niveau de préambule. De ce fait deux commandes de l'extension *french* sont inutilisables, car elles ne peuvent être activées (pour l'instant) qu'au `\begin{document}` :

- `\frhyphex` pour le chargement des exceptions concernant la coupure des mots,
- `\usersfrenchoptions` pour la personnalisation, dans le document, de l'extension *french*, ce qui est fort gênant, comme nous le verrons par la suite.

Bien entendu, dans ce cas, le mécanisme multilingue utilisé est celui de Babel. D'autres dispositifs globaux, imposés par Babel, peuvent intervenir par ailleurs.

L'intégrale *french*

Le cas le plus favorable à l'usager, désireux de disposer de toute la personnalisation de l'extension *french* est donc d'utiliser un *format* francisé (et d'éviter Babel). Nous verrons un peu plus loin que l'on peut utiliser, par contre, une autre extension multilingue ne réduisant pas les fonctions de *french*. Ainsi toute la panoplie d'outils de personnalisation est à la disposition de l'usager ; c'est *l'intégrale french* dont nous allons parler maintenant.

3. Utiliser l'extension *keyboard*

L'extension *keyboard* fait partie intégrante de la distribution *french* pour des raisons de mise en commun de codes TeX. Cette extension peut toutefois être utilisée de façon totalement séparée de *french*. Pour faire appel à *keyboard* on code : `\usepackage[`*codage d'entrée*`]{keyboard}`

Il existe actuellement les codages d'entrée suivants : `ascii` (sans intérêt majeur), `cp850` pour DOS, `ansinew` pour Windows, `applemac` pour MacOS, `next` pour système Next, `latin1` pour l'essentiel des Unix et `decmulti` pour les systèmes Digital. C'est ce dernier codage qui est pris par défaut car il apporte en général plus que l'iso-latin1 pour les systèmes à base d'Unix, et notamment les caractères « œ » et « Œ ».

3.1. Intérêt de *keyboard* ?

Quel est l'intérêt d'utiliser *keyboard* plutôt que *inputenc* ? Plusieurs raisons :

- seuls les caractères accentués utiles au français sont définis ; mais à l'inverse des caractères peuvent manquer pour d'autres langues[2] ;
- avec MlTeX les caractères 8-bits (donc les lettres accentuées entre autre) ne sont pas des caractères actifs mais de simples caractères ;
- les caractères accentués peuvent être employés en mode mathématique ;
- la conversion minuscule-majuscule par les ordres TeX de base (`\lowercase` et `\uppercase`) est assurée ;
- avec MlTeX la césure de mots contenant des caractères accentués est garantie ;
- dans un avenir (que l'auteur espère proche) les messages accentués sortiront à la console de façon lisible, sous réserve que le moteur TeX sache sortir des caractères 8-bits plutôt que les fameux `^^xx`.

2. L'extension *keyboard* n'a pas, pour l'instant, de support multilingue spécifique.

3.2. Les niveaux de personnalisation de *keyboard*

Le niveau le plus bas reste celui du *format* par l'appel à `hyconfig.tex` déjà cité. Dans ce cas il n'est plus nécessaire de charger l'extension par un `\usepackage`. L'installateur de site LaTeX peut choisir de modifier le fichier de configuration `keyboard.dat` pour imposer son jeu de caractères par défaut pour le site.

Dans le document lui-même on peut modifier le codage par défaut en donnant un autre nom de codage en option de `\usepackage`. Tout au long du document, ce codage peut être modifié *à la volée* en codant `\kbencoding{`*codage*`}`.

Si on le désire, il est même possible de définir un nouveau codage personnel ; on peut alors s'inspirer des fichiers `.kbc` pour créer `moncodage.kbc`. On peut même attribuer des actions (La)TeX spécifiques à des touches du clavier (choisies minutieusement), exemple :

```
%      \_____________________________________________________/
%      |___1__|___2__|___3__|___4__|___5__|___6___|___7__|___8__|%%%%%comments
    \ac[  \¡  |  \'  |  \"  |   \^ | {\c} |   {}  |   {} | {\r} ]%<==acc.7 bits
  \noms|  gr  |  ac  |  um  |  hat |   c  |   .   |   .  |   an |%<==acc. names
%-------------------------------------------------------------------------------
                        \MACROtrue
%-------------------------------------------------------------------------------
[{\fguillemets\typeout{Penser à fermer les guillemets}} %
       |   .  |   .  |   .  |   .  |  .   |   <<  |   .  |   .  |% guillemets
   \hex|   .  |   .  |   .  |   .  |  .   |  ^^S  |   .  |   .  ]% francais
[{\endfguillemets\typeout{Ok, j'ai fermé mes guillemets.}} %
       |   .  |   .  |   .  |   .  |  .   |   >>  |   .  |   .  |%
   \hex|   .  |   .  |   .  |   .  |  .   |  ^^T  |   .  |   .  ]%
%-------------------------------------------------------------------------------
```

On a utilisé ici l'envoi d'un message à chaque utilisation des touches guillemets du clavier mais tout autre code fonctionnel pourrait être mis en œuvre.

4. 150 commandes de base !

L'extension *french* apporte effectivement 150 commandes à l'utilisateur mais nous n'allons pas pour autant en faire l'inventaire. Donnons pour l'instant les grandes lignes de la personnalisation du noyau *french* qui se compose de 6 parties distinctes :

1. les traductions françaises,
2. des macros-outils,
3. la typographie fine,
4. la mise en page,
5. la césure des mots,
6. le mécanisme multilingue.

Chaque partie est commutable, c'est-à-dire peut être activée ou désactivée à tout moment au sein du document, exemple :
`\frenchmacros` ... `\nofrenchmacros`

Lorsque l'on souhaite personnaliser tout un document pour, par exemple, désactiver les `\frenchmacros`, on codera :

```
\usepackage{french}
\usersfrenchoptions{\nofrenchmacros}
```

N'importe quel code peut être mis dans cette commande même des commandes contenant le fameux caractère `@` ; celles-ci sont exécutées à chaque fois que l'on revient au langage *french*. L'ordre `\usersfrenchoptions` est la commande la plus importante à connaître pour personnaliser *french*.

Bien entendu, à tout moment l'on peut désactiver *french* dans son intégralité en revenant à l'état (presque) initial par l'ordre `\english`. On notera que : `\begin{nonfrench}` ... `\end{nonfrench}` est plutôt réservé au dépanage et plus particulièrement pour désactiver les caractères actifs qui n'interviennet que dans la partie typographie fine.

L'ordre `\english` fournit la personnalisation maximum ! (en fait la plus dure, puisqu'il n'y a plus aucune personnalisation française dans ce cas) mais il existe d'autres commandes ayant des effets moins radicaux, nous en verrons quelques-unes par la suite.

L'installateur du site peut choisir de configurer *french* d'une certaine manière et dans ce cas il donnera ses commandes dans un fichier `french.cfg`, on se reportera à [3] pour le détail de ce mécanisme.

L'ordre de chargement des différentes personnalisation de *french* est important :

1° après le chargement de `french.sty` intervient :

2° `frpatch.sty` (les corrections temporaires de l'auteur),

3° `french.cfg` pour les modifications du site,

4° les modifications apportées par la classe (de documents),

5° `\usersfrenchoptions` insère les modifications personnelles et viennent enfin

6° les modifications dans le texte du document.

5. Les traductions

Les différents libellés utilisés dans LaTeX sont, par défaut, en anglais, la partie `\frenchtranslation` permet de les traduire (Résumé, Annexe, Glossaire,

Table des matières, etc.). La date est aussi francisée. Toutes les extensions devraient se décharger sur les extensions linguistiques comme *french* pour la traduction des libellés utilisés, mais cela est encore rarement le cas. En attendant l'on peut, tout au plus, espérer pouvoir passer l'option *french* aux autres extensions. Ce sera par exemple le cas de :

```
\documentclass{article}
\usepackage{french}
\usepackage{varioref}
```

L'ordre de chargement des extensions est essentiel ici pour que l'option *french* soit passée à *varioref*. On préfèrera plutôt passer l'option *french* de façon globale, de la manière suivante :

```
\documentclass[french]{article}
\usepackage{varioref}
\usepackage{mlp}
```

On notera ici l'utilisation de l'extension généralisée *mlp* (Multi-Lingual Package) pour charger *french*. Cette extension fait aussi partie (pour l'instant[3]) de la distribution *french*.

Les utilisateurs souhaitent parfois modifier les libellés imposés aux tableaux et figures (FIG. et TAB.), dans ce cas on codera par exemple :

```
\usersfrenchoptions{\renewcommand{\figurename}{Figure}}
```

pour changer « FIG. » en « Figure ».

Si l'on veut rajouter une traduction spécifique à une classe ou à une extension, on le précisera de la manière suivante :

```
\usepackage{mlp}
\begin{document}
\addcaptionnames{\def\LIBELLE{Mon libellé}}
```

On notera bien que cette possibilité passe par l'utilisation de l'extension *mlp* et que l'on utilise la commande `\def` de TEX pour imposer ce nom, même s'il est déjà défini. La commande `\addcaptionnames` affecte uniquement le langage courant, mais porte toutefois sur tout le document, quels que soient les changements de langages successifs.

6. Les macro-instructions

Les ordres apportés par `\frenchmacros` fonctionnent uniquement avec le langage *french* ; cela fait partie des objectifs de conception de *french* visant à

3. L'auteur envisage en effet, lorsque d'autres langues auront pu être *connectées* à cette extension et que son utilisation aura été éprouvée, de faire une distribution spécifique indépendante de *french*.

remettre LaTeX dans son état (presque) initial, une fois que l'on n'utilise plus *french*.

Si l'on souhaite rajouter une macro-instruction au langage *french* l'on n'utilisera pas `\usersfrenchoptions` mais plutôt une définition globale au document comme par exemple :

`\newcommand{\mamacro}{\ifFMA` *le code de la macro* `\fi}`

En utilisant le test `\ifFMA` on n'exécutera le code que lorsque l'option `\frenchmacros` aura été activée.

Par contre, pour remplacer une commande existante par une autre on suivra le procédé standard :

```
\usersfrenchoptions{\renewcommand{\fup}[1]%
                     {\ifFMA$^{\mbox{\underline{#1}}}$\fi}%
                   }
```

Ainsi :

`{\Large B}\fup d` donnera B$^{\underline{\mathrm{d}}}$

Il y a dans l'extension *french* un mécanisme d'abréviation qui n'est pas en service par défaut. Pour l'activer il faut préciser l'ordre `\abbreviations` qui peut tout à fait être indiqué dans le fichier de configuration du site (`french.cfg`).

7. La mise en page

La commande `\frenchlayout` met en œuvre la mise en page française (retrait des débuts de paragraphe, numérotation des notes et des sections, lettrines, etc.)

Concernant les hauts et bas de page, l'extension *french* laisse faire LaTeX [4] ou les extensions comme `fancyhdr`. Par contre, pour la classe `letter`, il est possible d'indiquer ceux-ci en redéfinissant les commandes `\formhead` et `\formfoot`.

Il est possible de ne pas effectuer de retrait du tout en début de paragraphe, de section ou de chapitre en codant : `\usersfrenchoptions{\parindent=0pt}`. Cet effet est, bien entendu, supprimé en sortant de *french*, donc en passant à une autre langue (mais pas à l'environnement *nonfrench* qui n'agit pas sur la mise en page).

4. Ce n'est pas entièrement vrai dans le cas de la classe `book` pour laquelle le style de page a été homogénéisé.

Une remise à zéro des compteurs de sectionnement intervient dans *french* au début de chaque partie ; ce dispositif peut être annulé par la commande `\noresetatpart`.

`\noautomaticlettrine` a été créé pour des raisons de compatibilité ascendante et ainsi, par défaut, les lettrines sont réalisées à une taille déterminée éventuellement choisie par l'utilisateur (par défaut `\Huge`). Il est bien préférable de laisser l'extension *french* calculer la taille adaptée à la situation (c'est-à-dire selon la police utilisée). On précisera alors `\automaticlettrine`. Exemple :

```
\automaticlettrine
\parbox{4cm}{
\lettrine{Qu'elle est belle}
la lettrine r^^e9alis^^e9e auto%
matiquement par l'extension
\emph{french}.\par}
```

Qu'elle est belle la lettrine réalisée automatiquement par l'extension *french*.

alors que par défaut l'on obtient :

```
\noautomaticlettrine
\parbox{4cm}{
\lettrine{Qu'elle est belle }
la lettrine r^^e9alis^^e9e auto%
matiquement par l'extension
\emph{french}.\par}
```

Qu'elle est belle la lettrine réalisée automatiquement par l'extension *french*.

Dans le cas de la classe *letter* il est possible de produire un document avec des lignes plus large ; il suffit pour cela de préciser `\wideletter`.

8. La typographie fine

Cette partie typographique intervenant au niveau de la ligne est activée par la commande `\frenchtypography` ; elle applique les règles décrites en [8]. Elle gère l'espacement de la ponctuation, mais aussi différents autres dispositifs.

Voyons d'abord le cas des macros-instruction d'accentuation qui restent utilisables dans l'environnement `tabbing`, quand l'option `\tabbingaccents` est choisie :

```
\tabbingaccents
Avec \verb|\tabbingaccents|
je peux accentuer mes
\parbox{\hsize}{\center%
\begin{tabbing}
xxxxxxxxxxx \= yyyyyyy \kill
caract\¡eres \> utilis\'es \\
\¡e \¡E \¡u  \> \'e \'E \¡u
\end{tabbing}}
```

Avec \tabbingaccents je peux accentuer mes

caractères	utilisés
è È ù	é É ù

Pour les figures et tableaux, par défaut

`\captions{Titre de la figure}` donne : FIGURE 1.1 *Titre de la figure*

Toute la typographie peut être modifiée ici. On avait déjà vu la possibilité de changer FIG. en ce que l'on veut, on peut changer aussi le `\captionseparator` qui est initialisé dans *french* à - et si on le désire l'on peut rétablir une écriture droite :

```
\renewcommand{\captionseparator}{/}
```

`\captions{\emph{Titre de la figure}}` FIGURE 1.1 / Titre de la figure

La numérotation peut même être retirée sur les figures et/ou sur les tableaux (pour tout le document) : `\unnumberedcaptions{figure/table}`, mais alors il ne sera plus possible d'obtenir une `\listoffigures` ou une `\listoftables`.

Par défaut, les guillemets sont toujours en romain et construits à partir des glyphes des polices *lasy* car ces guillemets n'existent pas dans les fontes *cm* et aussi parce que les guillemets se prêtent mal à des modifications de graisse et de style. Cela reste donc la valeur par défaut dans *french*. Si l'on dispose de polices comme les *ec* ayant des guillemets français, on peut toutefois coder : `\guillemetsinallfonts`.

Pour améliorer les textes mal saisis on pourra coder par ailleurs : `\noenglishdoublequotes`, `\untypedspaces` et `\idotless`, cela ajoutera les espaces nécessaires devant la double ponctuation « ; ! : ? », transformera les guillemets anglais (“ et ”) en guillemets français et génèrera un « î » même si l'on code `\^i` (au lieu de `\^\i`).

9. Les coupures de mots

Cette partie (dont les fonctions ont éte décrites en [1, 2]) est activée par la commande `\frenchhyphenation`. On peut toutefois souhaiter retirer ponctuellement sur un paragraphe, par exemple, les coupures de mots en codant

`\nofrenchhyphenation` mais cela ne fera rien tant que l'on n'aura pas refait appel à l'ordre `\french` pour réinitialiser la francisation. Le résultat risque cependant de ne pas être celui que l'on espère car on obtient alors des coupures de mots correspondant au langage précédemment utilisé (par défaut l'anglais). Si on veut réellement annuler toutes les coupures on peut faire appel temporairement à un nouveau langage ne disposant pas de motifs de césure. Il en existe un qui a normalement été défini dans `language.dat` à la création du format ; il suffit de coder `\nohyphenation` pour annuler toute coupure à partir de ce moment précis et ce jusqu'à un nouvel appel à `\french`.

À la demande d'un comité *ad hoc* il avait été décidé que, par défaut, la coupure des mots contenant une majuscule est autorisée par *french*. L'auteur préfère, quant à lui, le choix inverse, plus rigoureux typographiquement, plus juste sémantiquement mais aussi plus contraignant. Un bon conseil : faire le test avec `\disallowuchyph` et voir si cela est vraiment trop contraignant.

Avec la fonte télétype (*tt*) il n'y a aucune coupure automatique de mots, ce qui est très souvent bien gênant ; cela peut être modifié, sous certaines conditions, avec l'ordre `\tthyphenation`.

Comment personnaliser les coupures de mots ou plutôt introduire des mots avec des coupures non prévues par l'algorithme et les motifs français ? Il est possible d'avoir son propre dictionnaire de coupure de mots adapté au domaine scientifique et technique de travail. Ce dictionnaire doit être dans un fichier `frhyphex.tex` et contenir un ou plusieurs ordres `\hyphenation`. Ce dictionnaire sera chargé automatiquement au `\begin{document}` si l'on a précisé au préalable l'option `\frhyphex`.

Le problème des mots composés n'est pas simple à résoudre avec TEX car il ne sait couper en standard que sur le trait d'union. Cela tient aussi au fait que nos claviers ne distinguent pas en général le trait d'union du tiret, ni même du signe moins. Si l'on désire que le mot puisse être coupé avant le trait d'union, on codera `\allowhyphens` juste avant le trait d'union. Si l'on souhaite que ce soit (aussi) sur le mot d'après on codera alors (aussi) `\allowhyphens` juste après le trait d'union. Il n'y a malheureusement pas de technique pour que cela s'applique sur tout le document.

10. Le multilinguisme

Aucune extension linguistique capable de personnaliser LATEX n'est maintenant indépendante des autres langues. Cela a toujours été le cas de *french* qui, dès le départ (Version 3.0), comportait son système de basculement multilingue. Cela est aussi le cas de l'extension *german* et maintenant d'autres extensions

comme le *romanian*. À l'inverse d'autres extensions, comme Babel, sont venues apporter tout un système pour le développement de styles linguistiques. L'extension *mlp* proposée avec *french* offre un cadre de fonctionnement multilingue pour toutes les extensions déjà prévues à cet effet mais en les fédérant de façon à tirer le meilleur parti de toutes les langues utilisées et sans intervenir sur les développements effectués par les groupes d'utilisateurs.

L'extension *mlp* est donc destinée à permettre le raccordement de styles linguistiques existants dont la solidité a été éprouvée *en solo* pendant quelques années plutôt que pour le développement de nouvelles extensions linguistiques. Autant les styles évolueront en fonctionnalités, autant *mlp* sera enrichi de dispositifs fonctionnels.

Pour raccorder une extension linguistique à *mlp* il faudra tout d'abord déclarer le nouveau langage dans la déclaration internationale, en anglais, dans le fichier `internat.mlp` de façon à ce que ce nom de langage puisse être mis en option d'appel de *mlp*. Ce même fichier sera ensuite traduit dans la langue voulue pour produire un fichier `<`*language*`>.mlp` donnant ainsi la possibilité d'appeler *mlp* avec les noms de langages utilisés localement. Pour terminer l'opération on s'inspirera du fichier `mlp-49.sty` qui raccorde l'extension `german` et l'on créera un fichier `mlp-`*nn*`.sty` où *nn* correspond au code téléphonique international. À partir de ce moment le nouveau langage est utilisable avec *mlp*, sous réserve d'avoir au préalable introduit les motifs de césure *ad hoc* et pour le moins créé une entrée dans `language.dat`.

Dans le cas où le problème à résoudre est d'abord de créer une extension linguistique, l'extension *french* peut y aider. Les nouvelles fonctionnalités seront définies à partir d'un ordre `\<`*langage*`>TeXmods` et une commande `\end<`*langage*`>` permettra de revenir au langage précédent *comme si de rien était*. Une fois ces deux opérations réalisées l'on définit ce langage pour *french* en codant `\NouveauLanguage[`*n*`]{`*langage*`}`. Cette nouvelle extension est alors utilisable par appel à `\<`*langage*`>`. L'extension *fenglish* donne un prototype fonctionnel qui peut servir d'exemple concret. Ensuite, après une longue période de tests, l'on pourra la raccrocher à l'extension *mlp*.

11. Conclusion

Il doit rester fort peu de recoins où l'extension *french* n'est pas ouverte à une quelconque personnalisation. L'installateur peut choisir ses options par défaut pour le site. Un groupe de travail peut mettre au point une extension contenant la personnalisation adaptée à son domaine de travail. L'utilisateur final dispose quant à lui des mêmes dispositifs pour ses propres documents.

D'autres exemples concernant la francisation peuvent être obtenus en [4].

Avec l'extension *mlp* il est possible de surcroît d'aller bien au delà de la francisation en adoptant un style linguistique spécifique, voire personnel.

Principales URLs pour télécharger *french*

`ftp://www.univ-rennes1.fr/GUTenberg/french/` * *original* *

`ftp://ftp.loria.fr/pub/ctan/language/french/`
`ftp://ftp.tex.ac.uk/ctan/tex-archive/language/french/`

Bibliographie

[1] J. DÉSARMÉNIEN, La division par ordinateur des mots français : application à TEX, *TSI* vol. 5 N° 4, 1986.

[2] D. FLIPO, B. GAULLE et K. VANCAUWENBERGHE, Motifs de césure français, *Cahiers GUTenberg* N° 18, 1994.

[3] B. GAULLE, *Notice d'utilisation du style french multilingue pour LaTeX*, document électronique (distribution logicielle du style *french*), 10e édition, 1998.

[4] B. GAULLE, *French style torture test*, document de travail (distribution logicielle du style *french*), 1998.

[5] M. GOOSSENS, F. MITTELBACH et A. SAMARIN, *The LaTeX companion*, Addison-Wesley, 1993.

[6] L. LAMPORT, *LaTeX, A document preparation system*, Addison-Wesley, 1994.

[7] L. LAMPORT, *An Index Processor For LaTeX*, 1987.

[8] *Lexique des règles typographiques en usage à l'Imprimerie nationale*, 3e édition, 1990.

Annexe : où mettre ces commandes ?

commandes *french*	préambule (après le `\usepackage`)	`\users-french-options`	après le `\begin{document}`
`\addcaptionsnames`			X
`\allowhyphens`			X
`\(no)automaticlettrine`		X	X
`\captionseparator`		X	
`\disallowuchyph`		X	
`\figurename` redéfini		X	
`\formhead/foot`			X
`\french/\english`			X
`\(no)frenchhyphenation`		X	X
`\(no)frenchmacros`		X	X
`\(no)frenchlayout`		X	X
`\(no)frenchtranslation`		X	X
`\frhyphex`	X		
`\fup` redéfini		X	
`\guillemetsinallfonts`		X	
`\idotless`		X	
`\kbencoding`	X	(X)	(X)
`\noenglishdoublequotes`		X	(X)
`\nohyphenation`			X
envir. `nonfrench`			X
`\noresetatpart`		X	(X)
`\NouveauLanguage`			X
`\parindent` redéfini		X	
`\tabbingaccents`		X	X
`\tthyphenation`		X	X
`\unnumberedcaptions`		X	
`\untypedspaces`		X	(X)
`\usersfrenchoptions`	X		
`\wideletter`			X

Nous indiquons ici l'emplacement normal de ces commandes, sachant qu'il peut exister certains cas où ces commandes peuvent être appelées d'un autre endroit.