*Cahiers GUTenberg*

# COMMENTAIRES SUR LA PORTABILITÉ DES DOCUMENTS (LA)TEX

Bernard GAULLE

*Cahiers GUTenberg*, nº 18 (1994), p. 61-86.

<http://cahiers.gutenberg.eu.org/fitem?id=CG_1994___18_61_0>

# Commentaires sur la portabilité des documents (LA)TEX *

Bernard GAULLE

*Institut du Développement et des Ressources en Informatique Scientifique, Informatique distribuée (BP 167, F–91403 ORSAY Cedex)*
gaulle@idris.fr

**Résumé**. Dans le *Cahier GUTenberg* N° 15 Daniel TAUPIN faisait état de sa réflexion et de ses expériences sur la portabilité [des documents [1]] TEX. Cet article reprend, point par point, sous les mêmes titres (à une ou deux exceptions près) et dans le même ordre, chaque élément pour le soumettre à une critique technique. Les quelques conseils de portabilité retenus par l'auteur sont beaucoup plus simples voire plus élémentaires que ceux énoncés dans l'article en question.

**Abstract**. *In* Cahier GUTenberg *# 15, Daniel Taupin expressed his thoughts and experiences about the portability of TEX documents. This article reviews, point by point, using the same headings (with one or two exceptions) and in the same order each of its elements and discusses their technical validity. This study reveals some rules for increasing the portability of (LA)TEX documents that are both simpler and more elementary than the ones proposed in the article in question.*

## 1. Le plus grand mérite de TEX...

TEX est certes un logiciel gratuit, de qualité, mais cela ne suffit pas pour expliquer pourquoi après 15 ans on utilise toujours ce programme de *formatage de document* malgré son utilisation exclusivement *batch* donc très peu convivial. Il faut y ajouter quelques qualités fondamentales :

– c'est un langage informatique à vocation typographique pour mettre en page des objets textuels, mathématiques ou autres. Grâce à ce langage, fort compliqué et unique en son genre, des documents informatisés dans les années 80 peuvent encore être recomposés, à l'identique,

*. Cet article ayant été largement écrit avant la diffusion de la dernière version de LATEX, ne prend pas, ou peu, en compte les apports liés à LATEX2ε. Il ne sera donc pas question ici des *classes* de documents ni des *extensions*.

1. Le titre original parle de portabilité de TEX alors qu'il s'agit des documents écrits en TEX.

et être imprimés sur les machines les plus modernes. Quel logiciel de PAO peut se targuer de cette qualité ?

– comme tout compilateur de langage, TEX produit un fichier objet (le « *dvi* ») qui est totalement indépendant de l'unité de restitution, écran, fax, imprimante, composeuse, etc. Ceci est un atout énorme qu'aucun logiciel de PAO ne peut, même à ce jour, revendiquer[2].

– longtemps TEX avec ses centaines de commandes, ses notions de boîte, d'élasticité, de groupage et autres complexités, est apparu – à juste titre – comme un instrument pour initiés de quotient intellectuel hautement supérieur... Cela est toujours vrai mais à la différence que l'on utilise beaucoup moins TEX directement depuis l'apparition de langages de plus haut niveau tel LATEX, plus à la portée de tous.

– ainsi TEX est devenu ces dernières années un *moteur* de mise en page commun à différents logiciels évolués. Ce résultat fait de TEX un standard *de facto*[3].

## 2. ... c'est sa portabilité

Le moteur TEX est certes portable puisqu'on ne connaît pas d'ordinateur usuel sur lequel il n'ait pas été installé. Est-ce vraiment un si grand mérite ? Les compilateurs Fortran existent aussi sur toutes les machines... mais ils sont tous différents !

S'il existe encore quelques vieux *moteurs* TEX V2 ou à l'inverse quelques extensions comme MlTEX, dans tous les cas le fichier `.dvi` obtenu produira le même résultat imprimé pourvu que l'on dispose des polices de caractères demandées ; ce qui ne peut pas poser de problème si l'on utilise les polices TEX par défaut (CMR) dont chaque installation est normalement pourvue.

On peut poser le problème des codages différents des documents comme par exemple entre PC et Macintosh, que chacun connaît mais qui fort heureusement se simplifie beaucoup avec TEX car il est muni d'un codage de transport commun à toute l'humanité ; nous y reviendrons.

2. Les logiciels de PAO savent, entre autre choses, produire des fichiers PostScript. C'est par le biais de ce langage de description de page, maintenant très généralisé, que l'on a l'impression d'une indépendance entre ces logiciels et les unités de restitution mais cela est un leurre – disons plutôt un artifice – qui nécessite bien sûr un interpréteur PostScript. Ce n'est pas obligatoirement le cas des fichiers *dvi*.

3. Des travaux sont d'ailleurs en cours pour que TEX devienne un standard ISO.

Ce qui n'est pas commun à tous les moteurs concerne les *formats* qui diffèrent en fonction du jeu de macros-instructions et du paramétrage utilisés à l'installation; on en distingue essentiellement deux: *plain*, le jeu standard de TEX et *lplain* [4] celui de LATEX. L'un et l'autre ne sont évidemment pas intervertibles. Ainsi un document préparé pour *plain* TEX ne peut être utilisé avec LATEX et inversement. Il ne faut pas y voir un défaut de portabilité. Par contre certains installateurs peuvent s'être écartés des standards et avoir personnalisé leurs *formats*; des défauts de portabilité peuvent alors être rencontrés mais, on le voit, ils ne peuvent pas être imputés à TEX.

La diversité des standards graphiques [5] pose réellement des problèmes de portabilité mais pas plus qu'entre logiciels commerciaux différents. À l'inverse TEX est probablement le seul logiciel sérieux à accepter presque n'importe quel type de graphique mais il est vrai qu'il n'en fait rien lui-même puisqu'il se contente de le transmettre – à l'identique – au pilote d'impression ou de visualisation. On comprendra alors aisément qu'un graphique au standard HPGL, par exemple, ne puisse pas être directement compréhensible d'une imprimante purement PostScript, donc d'un pilote PostScript. Et là encore, TEX n'y est pour rien...

Comme on le voit, ce n'est pas la portabilité de TEX qui pose un problème mais la portabilité de ce qui est autour de TEX. On peut ainsi se lamenter de la diversité des codages de caractères mais serions-nous plus contents si TEX ne connaissait que l'EBCDIC? Certes non! Heureusement TEX sait s'adapter à tous les codages de caractères: ceci est une force extraordinaire et non un défaut.

## 3. La portabilité, à quel niveau?

Il n'est pas juste de dire que la portabilité de TEX se situe soit au niveau du fichier source soit au niveau du fichier `.dvi`; il faut aussi y ajouter le niveau du langage de description de page (PostScript, HPGL, ... ) car de plus en plus les documents sont transportés dans ces formats (en attendant l'émergence d'autres standards de transport comme *Acrobat* d'Adobe ou

4. Avec LATEX2$_\varepsilon$ ce format est appelé, tout simplement, `latex`.

5. Il s'agit en général de standards de fait car chaque constructeur-développeur informatique développe son propre standard graphique.

*SDIF* [6] pour SGML). En résumé la portabilité de TEX doit être examinée à trois niveaux :

1. Le fichier source (`.tex`) qui est portable s'il n'est composé que de caractères ASCII [7] (7-bits, soit des caractères de 0 à 127), ce qui veut dire que chaque lettre accentuée devra être exprimée sous sa forme TEX de base (`\`*accent*`{`*lettre*`}`).

2. Le fichier objet (`.dvi`) qui est portable en tant que fichier binaire 8-bits. Le pilote de visualisation ou d'impression est soit très lié à un type d'écran ou à un type d'imprimante, soit à un langage de description de page mais jamais à un type d'ordinateur.

   Le fichier `.dvi` a été parfois préféré au texte source lorsqu'il s'agissait de porter un document fini d'un endroit (le producteur) à un autre (l'imprimeur, professionnel ou privé). Cela reste vrai dans deux cas :

   - lorsque l'on ne veut pas ou on ne peut pas transmettre le source ;
   - lorsque les deux endroits ont exactement les mêmes types de matériels de restitution [8].

3. Le fichier que j'appellerai « piloté » qui est le résultat du traitement du fichier `.dvi` par un pilote. Il est essentiellement portable entre matériels de restitution de même type. Ainsi un fichier PostScript est portable, à condition de ne pas avoir fait d'écart à la règle de base c'est-à-dire d'avoir utilisé les polices CMR (qui seront transportées dans le document). On peut même prendre un peu plus de risques et utiliser des polices PostScript mais dans ce cas on en restera aux 35 polices dites « usuelles ». Par contre, éviter à tout prix l'échange de documents vers des unités ayant des résolutions en points par pouce (*dpi*) différentes. On veillera aussi à ce que les moteurs d'impression laser utilisent la même technique d'écriture (noir sur blanc ou blanc sur noir, voir [13]), faute de quoi les rendus d'impression seront très différents. Ces petits ennuis ne sont en fait pas gênants en regard de la destination du document qui est quand même dans 99 % des cas de pouvoir être lu rapidement plutôt que d'être imprimé avec une

6. SGML Document Interchange Format.

7. Ce que l'on nomme communément ASCII est la norme ISO 646-1973, révisée ensuite par l'organisme de standardisation américain et officiellement appelée *American Standard Code for Information Interchange*.

8. On suppose alors que les pilotes utilisés sur les deux sites sont identiques, ce qui permet l'utilisation, à coup sûr, des ordres `\special` permettant l'usage de dispositifs graphiques.

excellente qualité typographique (objectif essentiellement réservée aux documents d'importance).

# 4. Défauts de portabilité de TEX

## 4.1. Défauts possibles de portabilité du TEXte source

Le texte source doit être considéré comme non portable dans deux cas :

1° lorsque le texte ne peut être compilé normalement avec le moteur d'arrivée ;

2° lorsque le résultat de la compilation (le fichier `.dvi`) est différent de ce qu'il devrait être.

Dans le premier cas, tout peut être imaginé, à commencer par des défauts venant de l'auteur lui-même, pour terminer par des anomalies dues aux installateurs.

Lorsque l'auteur utilise un sous-ensemble non finalisé, telle la première version du *New Font Selection Scheme* (NFSS, voir [15]), il ne faut pas s'étonner de certaines anomalies qui par la suite seront considérées soit comme des bogues et donc corrigées ou éliminées, soit comme des dispositifs nouveaux.

Si l'auteur, depuis son PC, transmet un texte 8-bits adoptant un certain *codepage* rien ne permet d'avoir l'assurance que chacun disposera du même *codepage* sur sa machine. Un auteur ne devrait jamais transmettre un texte source codé sur 8-bits[9]. Il peut arriver que l'auteur suppose l'existence de mécanismes ou de polices non standard et dans ce cas le texte source a de fortes chances de ne pouvoir être compilé. Il ne faut donc pas s'écarter, encore une fois, des polices CMR que chaque installation se doit de posséder.

Dans le deuxième cas la différence des fichiers objet ne se voit en fait qu'au moment de l'impression ou de la visualisation. Une formule mathématique composée différemment est une anomalie grave qui s'est effectivement produite dans la première version de NFSS, considérée par tous comme une version d'essai.

9. Malgré les avancées techniques rapides, la transmission d'un fichier 8-bits reste, à l'heure actuelle, encore une exception qui nécessite une validation préalable de bout en bout.

### 4.2. Défauts possibles de portabilité du DVI

Les documents imprimés depuis un fichier `.dvi` doivent être rigoureusement identiques quelque soit l'endroit où il est imprimé. Si votre installation n'utilise pas les polices CMR alors on peut imaginer de nombreuses différences car une substitution de polices risque d'être faite à l'arrivée sur l'autre site. Les polices DC donnent un dessin différent des caractères à tous les niveaux et notamment à celui des accents. L'emploi d'autres fontes donnera encore d'autres résultats ; tout cela est parfaitement normal.

On ne peut donc pas conclure que le fichier `.dvi` est plus portable que le texte source, d'autant que quelques ordres `\special` peuvent avoir été utilisés, auquel cas seul le pilote employé sera l'élément décisif. Ce qui nous amène à dire à nouveau que le fichier *piloté* est bien souvent le fichier le plus transportable.

## 5. Remèdes pour une meilleure portabilité de TEX

Il s'agit ici de chercher des remèdes pour accroître la portabilité du texte source. Toute méthode contraignante, que ce soit pour l'auteur ou l'installateur de TEX doit être évitée. Mais on pourra difficilement empêcher les gens de faire des erreurs et d'entraver ainsi tous les efforts louables de normalisation !

### 5.1. Rendre les implémentations de TEX compatibles ?

Contrairement à ce que chacun peut imaginer, l'objectif des actions de portabilité n'est pas la compatibilité des installations mais leur non-incompatibilité. C'est le principe des « systèmes ouverts » bien connu maintenant des informaticiens : chacun se doit d'accueillir naturellement l'autre, sans autre forme de procès. Cela veut dire qu'il faut respecter une norme minimale, qu'elle soit *de facto* ou non, peut importe. Et cette règle existe bien entendu :

> *Ce qui est écrit dans le* TEXbook *ou dans le* LATEX user's guide *doit fonctionner à l'identique quelque soit la personnalisation faite à l'installation dans le* format.

On peut rêver aussi à la portabilité parfaite des textes sources dans le temps mais il est difficile d'imaginer un monde entièrement *figé* où tout ce

qui pouvait être fait *avant* doit pouvoir être fait *maintenant* sans aucun changement. Cela interdirait de fait toute possibilité de revenir sur une erreur fondamentale de conception. Faut-il rappeler que TEX version 2 était incompatible avec la précédente version ? Nous verrons aussi certainement un LATEX V3 très différent de ce que l'on connaît aujourd'hui. À l'inverse il faut tout faire pour que la compatibilité ascendante soit un objectif mais uniquement après la correction des « erreurs passées ».

### 5.2. La portabilité, c'est dans la façon d'écrire !

Dès l'instant où les installations TEX sont non-incompatibles, on peut espérer que les textes source ne posent plus de problèmes. Faut-il tout de même prendre quelques précautions ?

- ne pas utiliser des commandes récemment introduites, donc inconnues des précédentes versions (de même si vous avez toujours respecté le manuel LATEX vous n'avez certainement pas remarqué le passage de votre moteur TEX à la version 3) ;
- pour rendre les textes lisibles et éditables partout, il faut éviter tout caractère ne faisant pas parti du jeu ASCII standard à 128 moments (c'est-à-dire 7-bits) utilisé dans le TEXbook ;
- si l'on souhaite une impression identique quel que soit le jeu de polices utilisées pour le document, cela veut dire que l'on fait abstraction des jeux de polices des installations. Il faut donc soit se ramener au seul jeu standard des moteurs TEX, c'est-à-dire les polices CMR (les seules référencées dans le TEXbook), soit transmettre les polices du document dans le document lui-même mais cela n'est pas possible au niveau du document source ;
- il va de soi qu'un document français ne sera pas correctement composé sur les installations non-francisées sauf si vous avez imposé « *à la main* » toutes les spécificités françaises dont notamment les points de division de tous les mots du texte. Cela semble vraiment une précaution qui demanderait un travail énorme, totalement disproportionné avec les résultats escomptés. Et puis ce serait faire à la main ce que TEX sait si bien faire automatiquement !

### 5.3. Styles et jeux de macros

Chacun est en droit de supposer deux éléments essentiels de ses destinataires :

1. que le moteur TEX et les logiciels de plus haut niveau soient standard c'est-à-dire qu'ils aient été normalement installés et qu'aucune modification locale ne soit venu l'altérer ;
2. que les dernières mises à jour de ces logiciels aient été faites ; cela ne veut pas dire nécessairement *la* toute dernière version mais plutôt une version de l'année s'il y en a eu plusieurs dans l'année [10] ou sinon une version datant de moins de trois ans. Au delà de cette durée une version de logiciel devient *malheureusement* [11] suspecte : n'avez-vous réellement réalisé aucun changement, aucun ajout, dans votre *paquetage* TEX depuis trois ans ? Si tel était le cas, autant tout remettre à jour sur le champ et veiller ensuite au rafraîchissement périodique des différents composants.

Par conséquent, lorsqu'on envoie un document à un destinataire on n'a pas à lui envoyer les styles standard LATEX ni aucun style communément connu comme `a4.sty`, `psfig.tex`, `pictex`, ... [12] mais par contre il ne faut pas oublier de joindre tout ce qui est personnel : macro-instructions, options de style et fichiers de données.

Il faut éviter au maximum l'envoi de tout style public car le risque de propager ainsi un style périmé, endommagé ou simplement bogué est plus grand que l'impossibilité du destinataire à se procurer le style en question. Seuls les styles faisant partie intégrante du document peuvent faire exception à cette règle [13].

On s'interdira, bien entendu, tout envoi de style faisant l'objet d'une licence d'utilisation. Il s'agit là de styles absolument non diffusables (à ne pas confondre avec les autres notions de portabilité ou de *copyright* qui sont des contraintes totalement différentes).

10. Avec LATEX2$_\varepsilon$, un cycle de une année (une mise-à-jour majeure et une mineure) est prévu pour garantir le synchronisme des versions.

11. Peut-être un jour verrons-nous un ralentissement dans cette course effrénée à la production de logiciels ?

12. Rappelons que ces styles sont accessibles sur le réseau Internet via les serveurs `CTAN` ou par l'intermédiaire des associations d'utilisateurs.

13. On notera avec plaisir le dispositif de LATEX2$_\varepsilon$ permettant de transporter des styles avec le document et de les décharger uniquement s'ils n'existent pas déjà, même si cela peut s'avérer dangereux quand la version transportée est périmée.

# 6. Quelques recettes de portabilité

## 6.1. La commutation de langues pour la coupure automatique de mots

Nous entrons ici dans les arcanes des moteurs TEX et des styles ; notre discours va devenir beaucoup plus TEXnique et emprunt d'expérience personnelle [14].

Depuis la réalisation de MlTEX [3] en 1986 et de TEX V3 [14] en 1990 il est possible de faire travailler TEX dans plusieurs langues, à la fois ou séparément. L'installation de MlTEX définit des ordres tels que `\fhyph`, `\ehyph` et `\ghyph` pour composer [15] ce qui suit en français, en anglais ou en allemand, respectivement. Les efforts de D. KNUTH pour la version 3 de TEX se sont bornés dans ce domaine à offrir le mécanisme de base et non à apporter à l'utilisateur final un dispositif standardisé. C'est ainsi que sont apparus des styles publics où le français [16] devait être impérativement le langage numéro 3 de TEX. Or le choix des langues et des numéros internes est laissé, bien entendu, à l'installateur de TEX. Le logiciel Babel [2] est venu proposer en 1991 une solution générale à ce problème, tout en amenant une commande légèrement plus compliquée que celle de MlTEX :

```
\selectlanguage{french}
\selectlanguage{english}
\selectlanguage{german}
```

(les définitions de ces langues en termes de fichiers de « *patterns* [17] » étant faites dans un nouveau type de fichier : `language.dat`).

C'est à partir de la version 3 du style french [8] en 1992 qu'un mécanisme complet, du même genre [18], est proposé, mais avec des syntaxes de commande plus simple :

```
\french
\english
```

14. Ce qui expliquera l'emploi fréquent de la première personne.

15. Le terme *composer* n'est pas tout à fait adapté puisque ces ordres ne font qu'activer le mécanisme de division des mots dans la langue appropriée, d'où d'ailleurs leur nom.

16. Il s'agit là d'un style qui avait été conçu et réalisé en dehors des pays francophones.

17. Les *patterns* sont des motifs de césure de mots, voir [6].

18. Le codage nécessaire à la lecture du fichier `language.dat` a été emprunté de Babel et modifié pour lui apporter quelques améliorations fonctionnelles.

`\german`

Rien de très novateur en fait car ces commandes étaient déjà en place dans beaucoup d'installations, dont les installations MlTEX où `\french` était un surensemble de la commande `\fhyph` d'origine.

On comprend alors que quelque soit la syntaxe employée par le rédacteur pour passer à une langue particulière dans son document, cette commande a de fortes chances d'être inconnue de l'installation d'accueil. Comment faire alors pour informer le moteur TEX destinataire de composer le document dans la langue voulue ?

1. Essayer de deviner quelles langues sont installées ? Cela relève beaucoup du jeu « *au petit bonheur la chance* » car nous n'avons pas dans TEX la possibilité de vérifier, à la composition du document que, par exemple, toutes les règles de coupure de mots d'une langue sont bien respectées. Mais pourquoi le faire d'ailleurs ? Depuis plus de 10 ans que je reçois des documents anglais, américains ou euranglais[19] par le réseau, je n'ai jamais vu un auteur se préoccuper de savoir si son document imprimé chez moi comporterait les bonnes coupures de mot. Il faut dire à ce propos que les motifs de césure que j'ai installés sur ma machine sont américains et datent déjà de quelques temps. Qui est assez compétent en anglais pour être choqué par une division de mot anglais malheureuse ?

   Cela dit, il ne me serait pas venu à l'esprit de proposer un *style french* pour franciser un moteur TEX sans qu'il ne comporte de motifs de césure français. L'ordre `\french` est bien plus qu'une simple commutation de langue interne dans TEX (`\language`) c'est-à-dire plus que l'application des motifs de césure (le `\fhyph` de MlTEX) ; c'est par lui que toute la francisation peut se faire. On ne peut donc pas remplacer `\french` par `\fhyph` et encore moins par `\language=`*n'importe quoi*.

   Il est peut-être bon de rappeler, à ce stade, que bien d'autres paramètres influent sur la coupure des mots, dont :

   `\lefthyphenmin` qui est le nombre minimum de lettres à conserver avant de pouvoir couper un mot ;

   `\righthyphenmin` qui est le nombre minimum de lettres à rejeter à la ligne lorsqu'un mot est divisé ;

19. Excusez ce barbarisme qui correspond à cette langue parlée et comprise dans presque tous les pays.

`\uchhyph` indique si les mots contenant une majuscule doivent être coupés ;

`\lccode` indique si la lettre précisée entre ou non dans le champ d'application du mécanisme de césure.

On consultera [9] et [6] pour plus de détails à ce sujet.

Ces paramètres doivent être précisés à chaque changement de langue, c'est pourquoi il est nécessaire de faire appel à quelque option de style comme avec `babel` ou comme avec le style `french`.

2. Plutôt que d'essayer de deviner la présence des motifs de césure français sur le site destinataire, certains préfèrent « dissuader les coupures de mots ». Mais attention il faut bien savoir qu'en déclarant `\hyphenpenalty=500` vous interdisez totalement la division des mots et donc vous augmentez les contraintes de composition, appauvrissant inévitablement encore plus la typographie de la mise en page. Ce n'est pas à mon sens une image à donner des documents français.

   Quel est le comportement du style `french` face à la francisation du moteur qui l'utilise ? Rien ne permet de vérifier que la césure française est en place. Pourtant la division des mots est un élément primordial dans la mise en page. C'est d'ailleurs, pour moi, la première des contraintes après la qualité du texte et le choix des polices ; tout le reste n'est que de la typographie fine. Si on attache aucune importance aux césures, pourquoi alors en attacher, par exemple, à l'espace fine devant le point-virgule ?

   Hormis l'initialisation des variables appropriées, le style `french` n'apporte rien de plus au niveau des césures. Il rechigne toutefois à laisser croire que la typographie française pourrait être appliquée sur un texte qui n'aurait aucune coupure de mot française. Il effectue donc quelques tests pour s'en assurer. Le fichier `language.dat` n'est jamais exigé, mais s'il existe, il est lu et décodé pour confirmer l'ordre des langues et leur nom, en ajouter d'autres éventuellement et permettre de préciser des fichiers d'exception (de césures). En l'absence de quelque signe de francisation dans le *format* et dans un éventuel fichier `language.dat`, le style `french` refuse de fonctionner. Ce n'est pas le cas du « style french du pauvre », `pmfrench.sty`. C'est ce dernier qu'il faut utiliser dans tous les cas où le *format* n'est pas francisable, qu'elle qu'en soit la raison.

   Pour ou contre une césure approximative ? Chacun sait que TEX refuse toute coupure de mot contenant un accent (`\accent`). Si on écrit `\'` ou

\‘ ou \" etc., on introduit un accent et c'est bien ce que l'on doit faire puisque nous avons vu qu'il fallait transmettre les textes sous forme 7-bits. Il existe bien sûr la possibilité de contourner l'interdiction de couper [7] en autorisant la coupure *après les lettres accentuées*. Cela ne peut donner que des résultats approximatifs qu'il ne faut en aucun cas, à mon sens, présenter à l'étranger si on souhaite défendre, un temps soit peu, la culture française d'autant que les effets indésirables peuvent encore se cumuler avec l'absence de motifs français.

### 6.2. Les caractères de votre TEXte

Votre texte, une fois saisi, est devenu en fait un fichier informatique c'est-à-dire une suite de valeurs numériques que l'on a l'habitude de lire en base 8 (octal) ou en base 16 (hexadécimal). Chaque paire hexadécimale (de `00` à `ff`) ou chaque triplet octal (de `000` à `377`) représente une touche ou une combinaison de touches de votre clavier. Ce fichier est portable sur n'importe quelle autre machine (moyennant parfois quelques précautions d'empaquetage[20]) mais cela ne veut pas dire que vous pourrez le visualiser et retrouver exactement votre texte. Pour que cela soit possible il faut et il suffit que les deux machines disposent du même *vecteur de codage* c'est-à-dire de la même correspondance entre codes numériques et caractères (dont ceux de tabulation, retour chariot et autres codes de fonction spécifiques aux claviers). C'est toujours possible entre machines de même système, tels UNIX, Mac, PC DOS et même entre deux systèmes différents, à condition de savoir changer le type de codage[21] (en DOS dans le fichier `config.sys`, en UNIX par les commandes `stty`, etc.) en général très lié au système d'exploitation ou au constructeur.

La récupération d'un texte saisi sur une autre machine ou un autre système suffit-elle à en permettre sa composition, puis son impression comme cela l'aurait été sur la machine d'origine ? Non, car chaque installation TEX a configuré le programme TEX au vecteur de codage qu'il considérait le mieux adapté à ce type de machine, de système ou de clavier. Ainsi vous pouvez avoir saisi un texte avec des caractères accentués comme É, è ou ï, les visualiser correctement sur l'autre machine et ne rien imprimer du tout

20. L'empaquetage (certains utilisent l'anglicisme *encapsulation*) permet de protéger le paquet de tous les dégâts prévisibles dans le transport. Ici il s'agit de maintenir l'intégrité des codes numériques.

21. En France on utilise généralement le code ISO-Latin-1 (ou 8859-1) sous UNIX et le *codepage* 850 sous DOS, qui sont deux jeux de caractères à 8-bits.

de ces caractères, sans même un message d'erreur[22], tout simplement parce que le code numérique de vos caractères est sans signification pour le moteur TEX de votre machine cible. Il faut alors, soit transcoder c'est-à-dire faire passer le fichier de l'ancien vecteur de codage au nouveau (cela se fait *au pire* avec un tout petit peu de programmation), soit adapter le moteur TEX au nouveau vecteur de codage (ce qui peut se faire parfois aisément lorsque le moteur dispose de filtres d'entrées-sorties mais cela nécessite de toute façon une petite programmation méticuleuse). Donc si vous n'êtes pas sensible aux charmes de l'informatique vous avez de bonnes chances d'avoir un texte parfaitement inexploitable.

Ce problème est souvent aggravé par le fait que l'émetteur du document n'a pas précisé quel vecteur de codage il a utilisé (souvent il ne le sait même pas lui-même) et que donc le receveur est incapable d'agir sur le fichier reçu. Tout cela peut sembler dramatique mais l'est beaucoup moins dans la pratique si on respecte la bonne règle du standard ASCII (qui est aussi le standard utilisé par TEX en interne). Ce standard est restreint aux 128 codes (7-bits) et ne comporte pas de caractères accentués ni de caractères nationaux. Que fallait-il faire alors ? Tout simplement ne pas envoyer votre texte sous sa forme présente (8-bits ou ASCII *étendu*) mais le transformer en ASCII (7-bits) au départ, quitte au receveur d'effectuer l'opération inverse, adaptée à sa machine. On trouve ces programmes dans les distributions GUTenberg[23]. On trouvera en annexe plusieurs exemples : sous UNIX avec les langages `lex` et `sed`, sous VMS en `EDT/TPU` et pour EMACS une version `lisp`.

Ma conclusion sur cette partie est donc sans ambiguïté :

> *Votre texte source (au sens des caractères qui le composent) est toujours portable à condition de le transmettre en* ASCII *(7-bits).*

### 6.3. Sélection des polices

Si vous sélectionnez des polices par leur nom il faut que ces polices puissent être trouvées ou construites sur la machine cible. Il faut donc éviter au maximum les polices qui sortent des standards. Quels sont les standards ? Les polices CMR sont indissociables de TEX ; je ne pense pas (je ne souhaite pas) qu'elles disparaissent des distributions TEX. Les polices

22. Seul le fichier `.log` est, dans ce cas extrême, un peu plus bavard.

23. `tex7a8` et `tex8a7` de N. BROUARD ou `kb7to8` et `kb8to7` de la distribution des fichiers du style `french`.

provisoires DC sont en voie d'achèvement ; ce sera bientôt un standard, que ce soit sous forme réelle ou plus probablement sous forme virtuelle. Pour l'instant ces polices ne sont pas encore d'usage généralisé car, parmi les problèmes restant encore à régler, le codage mathématique n'est pas encore entièrement normalisé.

En ce qui concerne l'utilisation de polices PostScript, le seul changement à apporter dans le document consiste à dire que vous changez de fonte en indiquant l'option de style `times` ou `helvetica` ou ... Le destinataire a alors tout loisir de retirer cette option s'il ne dispose pas de cette facilité. Mais attention : cette option de style ne peut pas être transmise avec le document car elle utilise d'autres fichiers très dépendants de l'installation et sans lesquels elle n'a aucun sens.

Bien entendu, n'importe quel gourou saura faire plus compliqué et donc inévitablement se retrouvera confronté à des problèmes de portabilité mais pour ce qui est de l'utilisateur courant les choses sont plus simples, en résumé :

> *Votre sélection de polices, quelle soit* CMR *ou* PostScript *ne doit pas poser de problème de portabilité.*

J'ai lu qu'il ne fallait jamais mettre de lettres accentuées, sous prétexte qu'elles pouvaient ne pas figurer dans les polices du destinataire ! Ce message, malgré toute son exagération, reste juste toutefois sur le fond, puisque nous avons vu que vous deviez envoyer votre document en ASCII (7-bits) et que donc vous transmettez uniquement des couples accent-lettre mais jamais de lettres accentuées 8-bits. Votre destinataire a alors l'une des deux possibilités suivantes :

1° Transformer le texte en 8-bits et le composer. Aucun problème ne devrait alors être rencontré si les fichiers de césure française sont identiques.

2° Dans le cas ou il ne dispose pas de polices 8-bits, il peut (ou il doit s'il n'a pas MlTEX) laisser le texte en 7-bits et le composer ainsi. Le seul inconvénient pouvant survenir est une mise en page ne faisant pas la césure de certains mots (les macros d'accentuation 7-bits de TEX inhibent la césure du mot). Mais votre correspondant a-t-il installé les fichiers de césure française sur sa machine ? Si c'est le cas, il y a fort à parier que son installation supporte le 8-bits et dans

ce cas il peut se ramener au cas le plus favorable (1°). S'il n'a pas ces fichiers de césure alors il est normal qu'il ne puisse pas formater le document comme vous.

Il faut absolument éviter les subterfuges de gourous qui pourraient laisser croire que TEX a réussi à couper les mots français, même accentués, car en aucun cas la coupure du texte ne peut être pleinement française, donc autant qu'elle reste dans la langue du destinataire.

*La mise en page d'un texte accentué ne dépend pas, en premier, de la sélection des polices mais de la présence et de l'efficacité des motifs de césure sur la machine cible.*

### 6.4. Pour les TEXperts en portabilité : tester les noms de polices ... est sans intérêt

Conseiller aux TEXperts en portabilité de tester le nom des polices du site cible n'a pas grand intérêt. En effet, hormis le fait que vous n'avez pas besoin de tester le nom des polices de votre destinataire puisque le texte que vous lui avez envoyé a été fait – selon notre conseil – avec des CMR, il est hasardeux de tester le nom des polices, car comme on le laissait entendre un peu plus haut, chaque utilisateur PostScript peut avoir défini des noms de polices internes (virtuelles ou non) qui lui sont spécifiques. Ainsi ce test a de fortes chances de ne jamais aboutir au bon résultat.

## 7. Portabilité des textes en LATEX

### 7.1. Portabilité des styles

Ce que nous avons dit pour les textes reste valable, *a fortiori*, pour les styles ; nous n'y reviendrons pas.

Tout style qui imposerait qu'une langue porte un numéro interne particulier est effectivement à proscrire car le document qui l'utiliserait ne serait plus portable.

*Le style* `french` *répond à tous ces critères : il est donc portable.*

### 7.2. Les « dirty tricks » de NFSS

Comme il a été dit précédemment, le nouveau dispositif de sélection des polices proposé par Frank MITTELBACH et Rainer SCHÖPF n'était pas un composant standard de LaTeX. Alors on peut toujours rechercher quelques « *dirty tricks* » qui vont permettre à un correspondant éloigné de composer le document même s'il n'a pas NFSS mais à quoi cela va-t-il vraiment servir ?

Lorsque l'on cherche la portabilité à tout prix, on s'astreint au maximum à respecter les standards. Les versions 1 et 2 de NFSS n'en étaient pas un. Maintenant ce n'est plus le cas puisque NFSS est désormais intégré, avec quelques changements encore, dans LaTeX2$_\varepsilon$.

## 8. Suppliques aux auteurs de styles et aux aménageurs de formats

Daniel TAUPIN relève trois « erreurs stratégiques » fréquentes : l'incompatibilité avec les versions antérieures, « la difficulté de détecter la valeur à attribuer au registre `\language` » et enfin le défaut d'adaptation des macros d'accentuation aux fontes utilisées.

### 8.1. Un impératif: la compatibilité avec le passé

Certes, le souhait le plus cher de l'utilisateur est que tout son investissement ne soit jamais remis en question par une nouvelle version logicielle. Mais est-ce que cette remarque-critique s'applique réellement à TeX ou à LaTeX ? Depuis TeX v2 en 1981, aucune commande décrite dans le TeXbook n'a changé d'effet. Quant à LaTeX, plus jeune (1986), il serait aussi bien difficile d'y trouver quelque changement de fond ; seules quelques bogues fonctionnelles ont été corrigées.

Il n'en est pas de même, bien entendu, lorsqu'il s'agit de styles non-standard ou de composants logiciels « *à l'essai* » tels que NFSS (V1 et V2). Mais il faut considérer que cette « évolutivité » est très profitable car ces styles s'enrichissent au fur et à mesure des essais et des nouvelles versions.

Nombreuses options de style sont toutefois restées très stables dans le temps, citons par exemple `a4` ou `psfig` ; d'autres ont beaucoup évolué tout en s'efforçant de rester compatible, c'est le cas de `french.sty` par exemple.

N'oublions pas que lorsque nous parlons de compatibilité nous ne considérons que la face émergée de l'iceberg, c'est-à-dire l'interface utilisateur et non le codage interne. Chaque fois que vous verrez une commande comportant le caractère `@` dans son nom, sachez qu'il s'agit d'une entité interne qui ne fait pas partie du jeu de commandes de l'utilisateur et que sa pérénité n'est jamais assurée ; elle ne doit pas être employée par un utilisateur mais est destinée au développeur.

LaTeX2$_\varepsilon$ introduit de nouveaux concepts et de nouvelles commandes. L'utilisateur a le choix de travailler en mode compatible c'est-à-dire comme avec les anciennes versions de LaTeX soit en mode 2$\varepsilon$ avec toutes les nouvelles facilités. Le choix a été fait ici très clairement d'envisager un avenir qui soit nettement différent du passé. D'ailleurs LaTeX3 ne ressemblera probablement plus du tout à notre LaTeX d'aujourd'hui.

## 8.2. Commutation des modes de coupure de mots

Nous avons déjà exprimé le point de vue qu'il ne fallait pas chercher à « deviner » les langues installées dans le moteur TeX des sites recevant nos documents. De nombreux arguments vont dans ce sens ; le plus sérieux est probablement qu'il n'existe, à l'heure actuelle, aucun mécanisme standard de commutation de langues et que, donc, tenter de le faire s'apparente à du *bidouillage* de bas étage.

On peut admettre qu'un auteur n'accepte pas que son document puisse être déformé par un lecteur éloigné, pour la simple raison que la composition du document aura été effectuée dans d'autres conditions, dans une autre langue par exemple. Que peut faire l'auteur d'un texte français désireux d'être lu « *à l'identique* » ? Deux possibilités sont envisageables :

1° Ne transmettre que le document « piloté » et non pas le texte source.

2° Au cas où il transmet le source, interdire la composition du texte si l'installation réceptrice ne contient pas le français :

```
\ifx\french\undefined\endinput\fi
```

(un message d'explication à destination de l'opérateur peut éventuellement être émis avant l'ordre `\endinput`).

La commutation des modes de coupure de mots se fait à la base, comme nous l'avons déjà vu, par l'ordre `\language` mais il existe d'autres façons d'activer la coupure de mots dans une langue donnée. Le choix de ce nom, `\language`, n'est d'ailleurs pas heureux puisque cet ordre n'agit que sur

les motifs de césure. Son mode de fonctionnement qui est assez primitif (`\language=`*numéro interne de la langue*) en fait une commande réservée à des usages internes. Il faut donc en proscrire l'usage dans les documents utilisateurs. Par contre, des interfaces logiques, plus conviviaux, sont possibles par l'intermédiaire du fichier de configuration `language.dat` dont voici le prototype fourni dans la distribution des « fichiers de style french » :

```
%============================================================================%
%| Language | patterns file | exceptions file % comments                     |%
%============================================================================%
 =usenglish  ushyphen.tex % this is the original default TeX language...
                          % Donald Knuth's original hyphen.tex file is unused
    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 without hyphen files
%============================================================================%
```

Comme TEX n'autorise l'enregistrement des motifs de césure qu'à la création des *formats* (`initex`), ce fichier de configuration n'est pleinement utilisé qu'à ce niveau. Seuls les fichiers d'exceptions peuvent être redéfinis par la suite à la composition du document. En cas de transmission de texte réellement multilangue, l'auteur aura souvent intérêt à joindre à son source (LA)TEX une copie de ce fichier en commentaire. Le destinataire disposera ainsi de toutes les informations nécessaires pour recomposer le document à l'identique, seule la partie logicielle peut lui faire défaut : `hyconfig.tex`, fichiers de césure et options de style, mais il choisira très certainement de le recomposer *au plus vite*, avec les éléments en sa possession sur son site, sans même envisager d'installer les motifs de césure *adhoc*.

### 8.3. Les macros d'accentuation

Puisque nous transmettons nos documents entièrement en 7-bits, nos caractères accentués seront représentés par des macros-instructions d'accentuation (comme `\'e` pour `é`). Dans l'hypothèse où le document n'est pas converti en 8-bits par le destinataire, il est impératif que ces macros-instructions produisent, c'est-à-dire impriment, le bon dessin de caractère, ou plutôt la glyphe (pour employer une terminologie plus adaptée). Or la position des accents dans les fontes peut varier. Cela s'exprime par un type de codage différent : `OT1` pour les fontes CMR, `T1` pour les fontes DC (selon LATEX2$_\varepsilon$ et la distribution des fichiers du style `french`). On trouvera donc des macro-instructions différentes selon que le codage de fonte est `T1` ou `OT1` ; c'est précisément ce que fait LATEX2$_\varepsilon$, tout comme l'interface

`kbconfig` (`keyboard.dat` et `keyboard.sty`) proposé dans la distribution `french`.

Il n'y a pas lieu de s'inquiéter, a priori, de la validité des glyphes produites par ces macro-instructions d'accentuation, d'autant que nous avons dit, maintes fois précédemment, qu'il fallait utiliser les fontes les plus standard, c'est-à-dire celles par défaut (donc des fontes CMR ou éventuellement des fontes virtuelles utilisant les fontes CMR), lorsqu'on désire transmettre un document à des destinataires dont on ne connaît pas l'installation TEX.

En résumé :

> *Il faut laisser le système* LATEX *utiliser les polices définies par défaut à sa génération.*

## 9. Conclusions

On l'a vu, toutes ces questions de portabilité n'ont lieu d'être que du fait de la présence de caractères accentués dans notre langue. Certains en gardent un très mauvais souvenir, comme celui que j'ai entendu lors d'une réunion au ministère de la recherche : « Chacun sait que TEX ne sait pas couper les mots contenant des lettres accentuées ». Les informations de mon interlocuteur étaient périmées depuis presque 7 ans ! Il ne faut pas que ce genre de faux message perdure, tout comme celui qui mettrait en garde contre la non portabilité de TEX. À l'inverse, comme dans toute opération de transport, il est nécessaire de prendre quelques précautions de bon sens. Voici donc les quelques conseils à retenir.

## 10. Trois conseils pour la portabilité des TEXtes

1° Solution maximaliste : ne transmettre que le document « piloté » complet.

2° Solution intermédiaire lorsque les sites disposent des mêmes matériels de restitution (et des mêmes pilotes) : transmettre en 8-bits le document objet (`.dvi`), solution très peu utilisée actuellement.

3° minimaliste : transmettre le document source en format 7-bits, après application des règles suivantes :

- n'utiliser que les fontes standard par défaut (à priori CMR) donc ne jamais donner de nom de fonte spécifique ;

– à la rigueur utiliser les fontes PostScript mais en faisant ce choix uniquement par l'option de style LaTeX ;

– éviter à tout prix ce qui est hors norme, mais si cela s'avère impossible, joindre alors tous les éléments complémentaires (ne pas oublier les figures) et pour s'assurer que l'ensemble est correct, le tester dans un environnement différent du quotidien.

## 11. Remerciements

Je tiens à remercier les personnes qui ont bien voulu relire et critiquer ce document ainsi que ceux qui ont contribué à la réalisation des annexes, particulièrement : Denis Girou et Philippe Louarn.

## Références bibliographiques

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

[2] J. Braams, « Babel, a multilingual style-option system for use with LaTeX's standard document styles », *TUGboat* Volume 12 N° 2, 1991.

[3] M.-J. Ferguson, « A Multilingual TêX», *TUGboat*, 6, N° 2, 1985, pp. 57–59.

[4] M.-J. Ferguson, « A Multilingual TêX», *Rapport technique de l'INRS-Télécomunications*, 87-23, 1987.

[5] M.-J. Ferguson, « Fontes latines européennes et TeX 3.0 », *Cahiers GUTenberg* N° 7, 1990.

[6] D. Flipo, B. Gaulle et K. Vancauwenberghe, « Motifs français de césure typographique », *Cahiers GUTenberg* N° 18, 1994.

[7] D. Flipo, L. Siebenmann, « Hyphenation in presence of accents and diacritics », *Proceedings of the 7th European TeX Conference, Czechoslovak TeX Users Group*, Septembre 1992, pp. 87–96.

[8] B. Gaulle : *Manuel d'utilisation du style* `french`, document électronique fourni avec la distribution logicielle V3.0, 1992.

[9] B. Gaulle : *Requirements in multilingual environments, Part I: Definitions*, document électronique (VT15D02) faisant parti du projet LaTeX 3.0, 1994.

[10] M. Goossens : « LaTeX2$_\varepsilon$, un aperçu », *Cahiers GUTenberg* N° 18, 1994.

[11] M. Goossens, F. Mittelbach and A. Samarin : *The LaTeX Companion.* Addison-Wesley, New York, 1994.

[12] D. E. KNUTH, *The TEXbook* , Addison-Wesley Publishing Company, 1991.

[13] P.A. MACKAY, « Un regard sur les pixels », *Cahiers GUTenberg* N° 12, 1991.

[14] D. E. KNUTH : The TEXbook, *Addison Wesley*, 1991.

[15] F. MITTELBACH et R. SCHÖPF, « A new font selection scheme for TEX macro packages », *TUGboat* volume 10 N° 2, 1989 et TUGboat volume 11 N° 2, 1990.

[16] F. MITTELBACH, « Interface description of NFSS2 », *extension NFSS2*, 1993

## **Annexe** A. Conversion 7-bits vers 8-bits en `lex`

```
\{\\\‘a\}|à   printf("à");
\{\\\"a\}|ä   printf("ä");
\{\\\^a\}|â   printf("â");
\{\\\:a\}|å   printf("å");
\{\\\‘A\}|À   printf("À");
\{\\\"A\}|Ä   printf("Ä");
\{\\\^A\}|Â   printf("Â");
\{\\\:A\}|Å   printf("Å");
...
\{\\\"\\i\}|ï   printf("ï");
\{\\\^\\i\}|î   printf("î");
\{\\\"I\}|Ï   printf("Ï");
\{\\\^I\}|Î   printf("Î");
...
\\"c "c|\\c\{c\}|ç   printf("ç");
\\"c "C|\\c\{C\}|Ç   printf("Ç");
\\"oe "|\{\\"oe"\}|\\"oe"\{\}|œ   printf("œ");
\\"OE "|\{\\"OE"\}|\\"OE"\{\}|Œ   printf("Œ");
\\"ae "|\{\\"ae"\}|\\"ae"\{\}|æ   printf("æ");
\\"AE "|\{\\"AE"\}|\\"AE"\{\}|Æ   printf("Æ");
\\"o "|\{\\o\}|\\o\{\}|ø   printf("ø");
\\"O "|\{\\O\}|\\O\{\}|Ø   printf("Ø");
"!‘"|   printf("");
"?‘"|   printf("");

"<<"|   printf("");
">>"|   printf("");


\\"ss "|\{\\"ss"\}|\\"ss"\{\}|ß   printf("ß");
\\"pounds "|\{\\"pounds"\}|\\"pounds"\{\}|¿   printf("¿");
\\"copyright "|\{\\"copyright"\}|\\"copyright"\{\}|©   printf("©");
\\"P "|\{\\P\}|\\P\{\}|¶   printf("¶");
```

## **Annexe** B. Conversion 7-bits vers 8-bits en `sed`

```
s/\\‘a/à/g
s/\\‘{a}/à/g
s/\\‘A/À/g
s/\\‘{A}/À/g
s/\\^a/â/g
s/\\^{a}/â/g
s/\\^A/Â/g
s/\\^{A}/Â/g
s/\\"a/ä/g
s/\\"{a}/ä/g
```

```
s/\\"A/Ä/g
s/\\"{A}/Ä/g
...
s/\\c c/ç/g
s/\\c{c}/ç/g
s/\\c C/Ç/g
s/\\c{C}/Ç/g
...
s/\\^\\i /î/g
s/\\^{\\i}/î/g
s/\\^I/Î/g
s/\\^{I}/Î/g
s/\\"\\i /ï/g
s/\\"{\\i}/ï/g
s/\\"I/Ï/g
s/\\"{I}/Ï/g
...
s/\\oe /œ/g
s/\\oe{}/œ/g
s/\\OE /Œ/g
s/\\OE{}/Œ/g
s/\\ss /ß/g
s/\\ss{}/ß/g
s/\\copyright /©/g
s/\\copyright{}/©/g
s/{\\it\\$}/$/g

s/<</«/g
s/>>/»/g
```

## **Annexe** C. Conversion 7-bits vers 8-bits en EDT/TPU

```
PROCEDURE TRAITEMENT ! procedure de traduction TeX vers Iso-latin-1

     work_buffer := create_buffer('WORK');
     SET (NO_WRITE, work_buffer) ;
     position(work_buffer);
     copy_text("test texte" )  ;

     position (beginning_of (main_buffer));

replace("\`a","à");
replace("\`{a}","à");
replace("\`A","À");
replace("\`{A}","À");
replace("\^a","â");
replace("\^{a}","â");
```

```
replace("\^A","Â");
replace("\^{A}","Â");
...
replace("\c c","ç");
replace("\c{c}","ç");
replace("\c C","Ç");
replace("\c{C}","Ç");
...
replace("\^{\i}","î");
replace("\^I","Î");
replace("\^{I}","Î");
replace("\"+ASCII(34)+"{\i}","ï");
replace("\"+ASCII(34)+"I","Ï");
replace("\"+ASCII(34)+"{I}","Ï");
...
replace("\oe ","œ");
replace("\oe{}","œ");
replace("\OE ","Œ");
replace("\OE{}","Œ");
...
replace("\ss ","ß");
replace("\ss{}","ß");
replace("\copyright ","©");
replace("\copyright{}","©");

endprocedure;

procedure replace (string_old,string_new)
loop
     locate_string := search ( string_old, forward, exact) ;
     if locate_string <> 0 then
            position(locate_string);
    this_occurrence := erase_character (length (string_old));
            copy_text (string_new);
            endif ;
     exitif (locate_string = 0) ;
 endloop;
     position (beginning_of (main_buffer));
endprocedure;

!  Les commandes d'exécution :
 input_file := GET_INFO (COMMAND_LINE, 'file_name') ;
 main_buffer := CREATE_BUFFER ( 'main', input_file) ;
 POSITION (BEGINNING_OF (main_buffer));
 TRAITEMENT ;
EXIT;
```

# **Annexe** D. Conversion 7-bits vers 8-bits en `lisp`

```
;; Macro EMACS pour convertir du TeX 7 bits en 8 bits
(defun tex7a8 ()
  "Fonction pour convertir un fichier TeX 7 bits en codes 8 bits."
  (interactive)
  (setq case-fold-search nil)
  (beginning-of-buffer)
  (while (not (eobp))
    (end-of-line)
    (if (not (eobp))
        (next-line 1)))
  (beginning-of-buffer) (replace-string "\\`a"  "à")
  (beginning-of-buffer) (replace-string "\\`A"  "À")
  (beginning-of-buffer) (replace-string "\\^a"  "â")
  (beginning-of-buffer) (replace-string "\\^A"  "Â")
  (beginning-of-buffer) (replace-string "\\c c" "ç")
  (beginning-of-buffer) (replace-string "\\c C" "Ç")
...
  (beginning-of-buffer) (replace-string "\\^\\i{}" "î")
  (beginning-of-buffer) (replace-string "\\\"\\i{}" "ï")
  (beginning-of-buffer) (replace-string "\\^I"  "Î")
  (beginning-of-buffer) (replace-string "\\\"I" "Ï")
  (beginning-of-buffer) (replace-string "\\oe{}" "œ")
  (beginning-of-buffer) (replace-string "\\OE{}" "Œ")

  (beginning-of-buffer) (replace-string "{\\`a}"  "à")
  (beginning-of-buffer) (replace-string "{\\`A}"  "À")
  (beginning-of-buffer) (replace-string "{\\^a}"  "â")
  (beginning-of-buffer) (replace-string "{\\^A}"  "Â")
  (beginning-of-buffer) (replace-string "{\\c c}" "ç")
  (beginning-of-buffer) (replace-string "{\\c C}" "Ç")
...
  (beginning-of-buffer) (replace-string "{\\^\\i}" "î")
  (beginning-of-buffer) (replace-string "{\\\"\\i}" "ï")
  (beginning-of-buffer) (replace-string "{\\^I}"  "Î")
  (beginning-of-buffer) (replace-string "{\\\"I}" "Ï")
  (beginning-of-buffer) (replace-string "{\\oe}" "œ")
  (beginning-of-buffer) (replace-string "{\\OE}" "Œ")

  (beginning-of-buffer) (replace-string "\\`{a}"  "à")
  (beginning-of-buffer) (replace-string "\\`{A}"  "À")
  (beginning-of-buffer) (replace-string "\\^{a}"  "â")
  (beginning-of-buffer) (replace-string "\\^{A}"  "Â")
  (beginning-of-buffer) (replace-string "\\c{c}"  "ç")
...
  (beginning-of-buffer) (replace-string "\\^{\\i}" "î")
  (beginning-of-buffer) (replace-string "\\\"{\\i}" "ï")
  (beginning-of-buffer) (replace-string "\\^{I}"  "Î")
```

```
 (beginning-of-buffer) (replace-string "\\\"{I}" "Ï")
 (beginning-of-buffer) (replace-string "\\oe{}" "œ")
 (beginning-of-buffer) (replace-string "\\OE{}" "Œ")


 (beginning-of-buffer) (replace-string "<<"  "«")
 (beginning-of-buffer) (replace-string ">>"  "»")
)
```