Codages d'entrée
aspects pratiques

Daniel Flipo

Journée GUTenberg

Paris, 8 octobre 2007

Cadre de l'exposé

Codage *d'entrée*
(pdf)LaTeX → Impératif : déclarer le codage d'entrée !
Système *Unix* → `\usepackage[`*codage*`]{inputenc}`
Éditeur : *Emacs*

codage=*latin1*, *latin9*, *ansinew*, *applemac*, *utf8*, *utf8x* (nécessite ucs).

Notion de *locales* : variable d'environnement `LANG` (ou `LC_ALL` ou `LC_CTYPE`)
Affichage (terminal) : `locale` ou `printenv` ou `echo $LANG`
Exemples (`LANG`) : `fr_FR.UTF-8`, `fr_FR.ISO-8859-1` (`fr_FR`), `fr_FR.ISO-8859-15` (`fr_FR@euro`), `en_GB`.
Initialisation par la distribution, l'administrateur ou l'usager.
`export LANG=fr_FR.UTF-8` dans le fichier `.bash_profile` (*shell* bash)
`setenv LANG fr_FR.UTF-8` dans le fichier `.(t)cshrc` (*shell* csh ou tcsh)
À tester avant déconnexion : `source .bash_profile` ou `source .cshrc`

Codages sous Emacs

- Laisser Emacs en mode « multibyte » (mode par défaut).
- Le codage utilisé est indiqué dans la barre d'info au dessus du mini-buffer, exemples : `-1` (latin-1), `-0` (latin-0 ou 9 ou iso-8859-15), `-u` (utf-8), `-=` (emacs-mule), `--` (ascii), ceci est caractéristique du mode « multibyte ».
- Le codage d'un *nouveau fichier* est défini à sa création à partir des *locales*, sauf utilisation de l'option « Set Language Environment » (déconseillé).
- Le codage d'un fichier *existant* est reconnu à la lecture du fichier, sauf ambiguïtés (gérées par une liste de priorité dépendant des *locales*). Exemple : latin-1 ou latin-9 ou iso-8859-5 (cyrillique) sont indistinguables.
- On peut forcer un codage en ajoutant en fin de fichier

  ```
  %%% Local Variables:
  %%% coding: iso-8859-5
  %%% End:
  ```

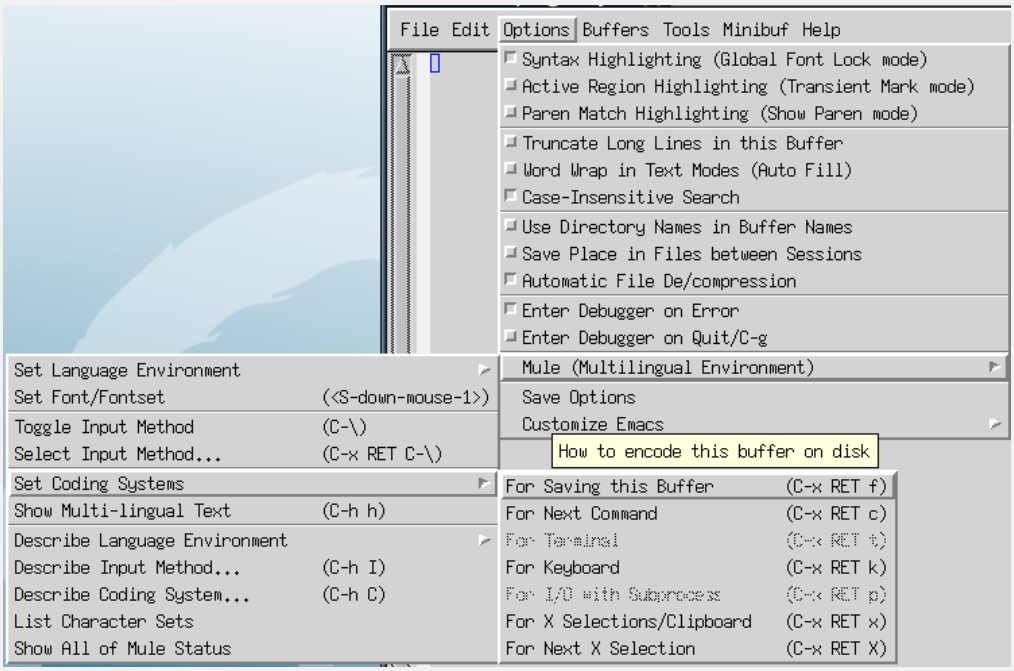
  Ces variables ne sont prises en compte qu'à *l'ouverture* du fichier.

Démo sous Emacs

Conclusions :

- La compilation d'un texte codé en utf-8 ne pose aucun problème.
- Les copier-coller entre deux fichiers ouverts sous Emacs mais codés différemment sont possibles (Emacs transcode à la volée) mais parfois avec des surprises... Harmoniser les codages avant de copier-coller est plus sûr.
- Sauvegarder *immédiatement* après le copier-coller (échec éventuel ?).
- *Vérifier* s'il y a eu changement de codage dans le fichier cible lors de la sauvegarde. Adapter éventuellement
  - le champ `coding:` (pour Emacs),
  - l'option d'`inputenc` (pour LaTeX).
- Faire des copier-coller entre fichiers de nature différente (`.doc`, `.odt`, `.pdf` ou `.html`) et `.tex` est *risqué* (cas de l'apostrophe courbe, des tirets longs, etc., sous Emacs-21). Conseils :
  - passer par un fichier tampon ouvert sous Emacs et procéder comme ci-dessus après suppression éventuelle des caractères parasites (ne *jamais* accepter de sauver sous un codage exotique comme `x-ctext` ou `emacs-mule`) ;
  - passer dès que possible à Emacs-22 qui possède un support unicode nettement plus complet.

Conversions sous Emacs



Menu « Options » : Mule → Set Coding Systems → For Saving this buffer et choisir le codage voulu (<TAB> pour voir la liste).

Autres éditeurs : Kile, TeXmaker. . .

- Kile et TeXmaker prennent en compte les *locales* pour le codage des *nouveaux* fichiers (comme Emacs).
- Les *anciens* fichiers sont *aussi* ouverts dans le codage défini par les *locales*, ce qui les rend illisibles si leur codage est différent.
- Kile 1.9 permet de choisir un codage pour lire un fichier (fenêtre en bas à gauche ou menu Outils → Encodage).
- Sous TeXmaker 1.5, aller dans Options → Configurer TeXmaker → Éditeur : codage « System » par défaut, choisir un codage adapté et ouvrir *ensuite* le fichier.
- Sous Kile 1.9 les copier-coller entre fichiers de codages différents fonctionnent, avec message lors de la sauvegarde s'il y a risque de perte de caractères.
- Le fichier test.html se copie sans problème dans les fichiers codés en utf-8 ouverts sous Kile 1.9 ou sous TeXmaker 1.5 (comme sous Emacs-22).

Conversions externes

Les commandes Unix `iconv` et `recode` (à lancer dans un terminal) permettent de convertir un fichier d'un codage à une autre.
L'option `--help` affiche une aide en ligne et `-l` donne la liste des codages disponibles. Exemples :

```
iconv --help
iconv -l
iconv -f MAC -t LATIN1 -o file.out file.in
iconv -f LATIN1 -t UTF8  <file1.tex  >file2.tex
```

```
recode --help
recode -l
recode MacRoman..latin1 file   (*écrase* file !)
recode latin1..UTF-8   *.tex   (faire une archive avant !)
recode latin1..UTF-8   <file1.tex  >file2.tex
recode -d  UTF-8..TeX  <file1.tex  >file2.tex
```


Tri de caractères (bibliographies, index)

LaTeX fonctionne en utf-8, mais tout n'est pas réglé !

Bibliographies

- bibtex a été conçu pour l'ASCII, il peut gérer les accents codés à la TeX (`\'e`) mais les tris sont faits selon l'ordre ASCII !
- bibtex8 trie correctement les fichiers .bib codés en 8 bits, le `.bbl` produit est au même codage.

  ```
  \usepackage[latin1,utf8x]{inputenc} % préambule
  \inputencoding{latin1} % bascule vers latin-1
  \bibliography{mabib}   % bibtex8 -c 88591lat.csf fichier(.aux)
  ```
- Rien (sauf Bibulus ?) pour Unicode.

Index

- makeindex peut fonctionner (avec béquilles) en codage 8 bits et en utf-8 (ex. `\index{eleve@élève}`)
- xindy, lui, trie sans problème les caractères 8 bits et utf-8 (sauf les caractères chinois). À installer donc (non inclus dans TeXLive) !