Adaptation automatique d'un éditeur à divers encodages ?

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 février 2012. Dernière mise à jour le 2 février 2012

https://www.bortzmeyer.org/encodage-auto.html

On lit parfois que le changement de l'encodage par défaut sur une machine Unix est simple, on modifie une variable d'environnement, par exemple LC_CTYPE et c'est tout. C'est oublier les fichiers existants : changer la variable d'environnement ne va pas magiquement les convertir. Est-ce que l'éditeur emacs s'adaptera tout seul au cas où on a un mélange de fichiers avec l'ancien et le nouvel encodage ?

Ce problème est un des obstacles quand on veut migrer un $HOME rempli de fichiers accumulés depuis des années, par exemple depuis Latin-1 vers UTF-8. Convertir tous les fichiers n'est pas forcément réaliste, surtout si on travaille en commun avec des personnes qui ont un autre encodage par défaut. Mais un éditeur comme Emacs n'est-il pas assez intelligent pour s'adapter tout seul ? Testons-le, sur une Debian version stable (nom de code "*squeeze*"). Lorsqu'on ouvre un fichier avec Emacs, un caractère à gauche de la barre d'état indique l'encodage (« 1 » pour du Latin-1, « u » pour de l'UTF-8, etc ; si on veut qu'Emacs soit plus explicite, M-x describe-current-coding-system affiche l'encodage actuel).

Avec du texte seul sans aucun marquage, c'est fichu : Emacs ne peut pas connaître l'encodage (sauf à tenter des heuristiques plus ou moins fiables) donc c'est manuellement, lorsqu'on ouvre le fichier, qu'on doit changer l'encodage (C-x RET f ou bien dans les menus Options -¿ Mule -¿ Set coding systems) s'il ne correspond pas à l'encodage par défaut.

Avec du XML et l'excellent mode Emacs nxml-mode <http://www.thaiopensource.com/nxml-mode/>, tout marche bien. Emacs bascule automatiquement dans le bon mode en regardant si la première ligne est <?xml version="1.0" encoding="utf-8"?> ou bien <?xml version="1.0" encoding="iso-8859-1"?>. (C'est bien l'en-tête qui compte : si on triche et qu'on met du Latin-1 en le marquant comme UTF-8, Emacs passe en mode UTF-8 quand même.) On peut donc travailler avec des fichiers XML qui sont un mélange d'encodages sans problèmes.

Avec LaTeX, formidable, le mode LaTeX détecte l'encodage et Emacs s'adapte. Comme pour XML, il n'inspecte pas le document, il regarde juste s'il y a \usepackage[latin1]{inputenc} ou bien \usepackage[utf8]{inputenc}.

Et en Python, langage de programmation qui permet (voir le PEP-263 <http://www.python.org/peps/pep-0263.html>) de mettre de l'Unicode dans le source ? Le mode Python d'Emacs détecte la mention `# -*- coding: latin-1 -*-` ou bien `# -*- coding: utf-8 -*-` et s'adapte correctement.

Décidemment, tout semble bien se passer, à part le triste début avec le texte seul. Essayons un autre format de données qu'XML, JSON (RFC 4627[1]). JSON ne permet qu'un seul encodage, UTF-8 (et n'a donc pas d'option pour en indiquer un autre). Patatras, malgré cela, le mode JSON d'Emacs n'est pas assez intelligent pour passer en Unicode automatiquement et l'édition devient pénible.

Et enfin, pour du texte reST (reStructuredText, celui utilisé pour CNP3 <https://www.bortzmeyer.org/cnp3.html>) ? Les programmes qui traitent ce format trouvent tout seuls si l'entrée est en UTF-8 ou pas (je ne connais pas de moyen de l'indiquer dans le source). Mais le mode Emacs `rst` (paquetage `python-docutils` chez Debian) ne le fait hélas pas.

Si de courageux lecteurs veulent tester Emacs avec d'autres langages, qu'ils n'hésitent pas à m'envoyer les résultats, que je publierai ici, avec mes remerciements.

On l'a vu, le problème dans certains formats (texte brut, notamment), est l'absence de mécanisme pour marquer explicitement l'encodage. Emacs ne semble pas pratiquer le "*content sniffing*", cet examen des octets pour essayer de déduire l'encodage (notons que cette méthode est imparfaite, par exemple elle ne permet pas de distinguer Latin-1 de Latin-9). Emacs a une solution générique pour ce problème de marquage de l'encodage (merci à @T1B0 <https://twitter.com/T1B0/> pour le rappel), mettre une ligne contenant `-*- coding: THEENCODING -*-` au début du fichier (on précède en général la ligne avec un signe indiquant un commentaire du langage considéré). Cela ne résout pas vraiment le problème du texte seul (la ligne en question sera affichée, déroutant les lecteurs, grep, etc). Mais cela traite bien le problème de reStructuredText. En mettant au début :

```
.. -*- coding: utf-8 -*-
```

Tout marche.

En conclusion, si on change l'encodage de son système, on n'y coupera pas de convertir les fichiers texte.

Et les autres éditeurs ? Mes lecteurs ont-ils des expériences à ce sujet ? J'ai fait quelques essais avec vim et il ne semble pas capable de s'adapter, par exemple à un fichier XML en UTF-8. En cas de modification du texte, il écrit selon la "*locale*" courante, produisant des fichiers XML qui ne sont pas bien formés. Mais il a des capacités de détection automatique de l'encodage par heuristiques successives. Par défaut, elles ne sont pas activées mais, si on met dans son `/.vimrc` :

```
set fileencodings=ucs-bom,utf-8,iso88591
```

Alors, vim cherchera un BOM, puis testera UTF-8 puis enfin se rabattra sur Latin-1. Merci à Olivier Mengué pour cet utile information !

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc4627.txt`