Le Francophones−HOWTO : Linux & la langue française

Écrit par Guylhem Aznar, <guylhem(à)metalab.unc.edu>

v3.2.2 ; Décembre 2001

Table of Contents

1. Présentation

1.1. Introduction

Ce document a été rédigé pour aider à configurer un ordinateur doté du système d'exploitation Linux pour des utilisateurs francophones.

Il y est expliqué par exemple comment configurer périphériques et applications afin de prendre en compte les particularités et les spécificités de notre langue...

Toutefois, ce document ne traite pas de l'installation du système d'exploitation Linux ; il est implicitement supposé que vous ayez correctement installé une distribution de Linux, RedHat, Debian ou Suse, cette liste n'étant pas exhaustive, et que vous vous apprêtiez maintenant à mieux la configurer ; veuillez donc vous référer à d'autres HOWTOs pour l'installation.

Les adresses électroniques ont été volontairement supprimées ou modifiées en remplaçant les « a enroulés » par des « (à) » afin d'éviter les publipostages très gênants.

Si quelque terme employé vous semble un peu obscur, je vous invite à vous reporter au lexique (« Vocabulaire : un petit lexique »), expliquant certains mots tels 8 bits, AZERTY ou QWERTY, qui sont employés tout au long de ce document.

1.2. Fichiers utilisés dans ce HOWTO

Tous les fichiers cités dans ce HOWTO, sont disponibles sous licence GPL sur ma page

Je vous recommande de télécharger tous ces fichiers pour les utiliser selon les exemples proposés.

1.3. Les Français ne sont pas les seuls francophones !

1.3.1. a) Différentes versions

Pour chaque manipulation dépendant du pays concerné, un certain nombre de versions peuvent être proposées :

- une canadienne normalisée (cn)
- une belge wallonne (be)
- une française (fr)
- une québécoise (qc)
- une suisse romande (sf)
- une états−unienne internationale (usx)

1.3.2. b) Les états−uniens : raccourcis claviers

La version états−unienne internationale permet à tous les francophones utilisant un clavier « 7 bits » aussi nommé « QWERTY » de bénéficier de l'étendue des caractères 8 bits.

Est disponible aussi une version pour ceux qui utilisent un clavier « Happy Hacking ».

1.3.3. c) Le Luxembourg : comme la France

Le Grand Duché du Luxembourg utilisant les conventions françaises, j'invite les lecteurs luxembourgeois à se reporter aux exemples français.

1.3.4. d) Le Canada : 3 claviers

Il est classique de parler des claviers « canadien international », surtout destinés aux anglophones (qui leur préfèrent toutefois les claviers QWERTY standard) et les claviers « canadien français », aux lettres accentuées.

Mais, de nouveaux claviers dits « canadiens normalisés » ont fait leur apparition récemment pour fusionner les modèles « canadien international », non concerné par le cadre du Francophones HOWTO, et « canadien français ».

Ces nouveaux claviers répondent aux normes CAN/CSA Z243.200−92 et ISO 9995−7 ; ils sont prévus pour être utilisés dans n'importe quelle langue, en remplaçant par exemple les intitulés des touches par des pictogrammes : « Del », en anglais, qui se traduisait par « Suppr » en français devient ainsi trois barres obliques fines orientées sud−ouest/nord−est.

Plus d'explications sont disponibles sur le site de l'Office de la Langue Française.

Toutefois je n'ai pas pu réussir à me procurer un tel clavier ; mes fichiers de configuration sont donc uniquement basés sur les (illustrations) et les normes théoriques, et non les normes pratiques.

Par exemple, si j'ai compris que la touche « Control » de droite est utilisée comme une « AltGr » mais sert à obtenir encore d'autres caractères, je n'ai aucune idée du comportement de ces touches lorsque seule l'autre a une sérigraphie prévue...

Si on peut m'envoyer un tel clavier, et surtout son pilote, je promets de mettre les fichiers de configuration à jour :−)

1.4. Attention

1.4.1. a) ROOT

Certaines manipulations recommandées par ce document ne peuvent être effectuées que sous le compte privilégié « root » ; la plus grande prudence est alors de rigueur car toute action inadéquate peut mettre en péril le contenu des disques.

Sauvegarder les fichiers existants, par exemple en les copiant sous le même nom suivi du préfixe .old, avant de les modifier ou remplacer.

1.4.2. b) CHERCHER

Le paquetage GNU nommé find offre un programme appelé *locate*, grâce auquel l'utilisateur peut retrouver le nom complet, chemin de répertoires, d'un fichier dont il ne connaît que le nom.

Certaines distributions Linux fournissent cela en /usr/bin/locate.

En tant que root, lancer *updatedb* invoque un *find /* sur les disques montés et place les noms de tous les fichiers sous forme hash dans /usr/lib/locate/find.codes (ou /var/lib/slocate/slocate.db pour slocate, utilisé dans les distributions modernes.)

locate permet ensuite de localiser l'un d'eux :

```
        (guylhem@victis:guylhem)$ locate noms_des_fichiers_à_retrouver
```

2. Les problèmes

2.1. Présentation

Le problème de tout utilisateur francophone après l'installation de Linux est de lui faire comprendre qu'il utilise des normes différentes des standards par défaut états−uniens..

Cela a des conséquences sur :

- la police de caractères donc les caractères spéciaux que l'on utilise, accentués comme É È À Ç ou bien ligaturés comme Æ , voire « e dans l'o » qui vient d'apparaître sous Linux grâce à la norme ISO−8859−15 aussi nommée latin−9
- la correspondance entre la sérigraphie des cabochons ou « touches » du clavier sur lesquelles vous appuyez, et les codes que Linux renvoie, celui−ci supposant un clavier QWERTY états−unien par défaut
- le format de page pris en charge par les programmes que vous utilisez pour imprimer, différent entre les États−Unis; (14×8,5 pouces pour du papier dit légal) et l'Europe (29,7×21 cm pour des feuilles a4)
- le format des dates et de l'heure, la position des jours et des mois étant variable selon les régions de la francophonie
- le format de la monnaie, aussi variable selon les francophones
- la langue utilisée pour les messages d'erreur

2.2. Solutions

Tout est préconfiguré pour un états−unien : bien que ceux−ci ne représentent pas la majorité de la population mondiale, ils ont été précurseurs en informatique.

Il faut donc se faire reconnaître comme une « exception » par les différents programmes, par des options ou des variables à exporter voire dans les pires des cas une modification du code source suivie d'une recompilation.

Heureusement, la philosophie GNU de Linux rend le système très ouvert à des modifications de toutes sortes et comme vous allez le découvrir au fil de ce document, et il est assez simple d'obtenir un résultat très correct.

2.3. Aider

Pour le moment, le futur de l'internationalisation de Linux porte sur la traduction des messages d'erreur, des programmes comme KDE ou GNOME, des pages de manuel et de la documentation, alors si vous vous sentez l'âme d'un traducteur, n'hésitez pas à contacter l'équipe de traduction des HOWTOs ; pour cela, adressez−vous à Éric Dumas <dumas(à)freenix.fr> ou <dumas(à)linux.eu.org>.

3. Le vocabulaire : petit lexique

Tout d'abord, un mot sur le vocabulaire employé :

3.1. 8 bits

Des caractères sont dits « 8 bits » s'ils correspondent à des accents ou à des signes spéciaux comme « § », non inclus dans le standard *ASCII* « 7 bits ».

Ce standard, sans accents ni caractères spéciaux, correspond aux 128 (2^7) premiers caractères des 256 (2^8), de 0 à 255.

Pour des raisons historiques (place disponible, inutilité pour les états uniens) seuls les 128 (2^7) premiers caractères ont été normalisés par l'ASCII : par exemple le A qui est le 65e caractère de la table ASCII s'écrit 0100 0001 en notation binaire.

Pour les caractères situés hors de la norme ASCII, différentes « normes » ou « standards » incompatibles, comme l'Unicode, l'iso−8859, le latin, le cp (...) fixent ces correspondances.

Par exemple, dans le standard ASCII étendu par l'iso−8859−1, le 224ème caractère équivalent noté « eb » en hexadécimal correspond à « ë ».

3.2. La police de caractères

Une police de caractères est une correspondance entre des codes informatiques et des caractères (alphanumériques...) affichables par l'ordinateur.

La police de caractères est l'ensemble des représentations graphiques des caractères d'un standard.

3.3. Les polices de caractères

Il existe plusieurs polices de caractères 8 bits pouvant répondre au besoins des francophones.

Je vous recommande de lire à ce sujet l'excellent site de Roman Czyborra qui a fait une thèse sur ce sujet et qui vous donnera une explication fort détaillée

Pour résumer, il est nécessaire de choisir une police de caractères pour que tous les caractères français soient disponibles.

Voici la liste des polices dites « ISO 8859 », mais attention, il existe leurs équivalents en « code page » pour windows (comme cp1252).

- ISO 8859−1 : latin 1 : langues d'Europe de l'ouest, Afrikaans, Swahili. Le Swahili, n'utilise aucun caractère accentué (comme l'Anglais, le Malais et l'Indonésien), donc n'importe quel version fait l'affaire. Pour l'Afrikaans, je ne sais plus exactement quel était le choix initial (c'est−à−dire où avait été rangé le 'n au départ), mais il ne me semble pas que ce soit dans Latin 1. Dans la pratique, bien sûr, les Afrikaaner utilisent Latin 1 ; la même chose se passe pour un certain nombre des cas qui suivent.
- ISO 8859−2 : latin 2 : langues d'Europe centrale et orientale. Dans les faits : Polonais, Tchèque, Slovaque, Slovène, Croate, Hongrois, Roumain, probablement aussi le Sorbe ou Sorabe (Wende en allemand). Officiellement, je crois que l'Albanais s'écrit aussi en Latin−2, mais comme dans les faits c'est identique au Latin 1... Il y a un problème pour le Roumain, car le Latin 2 contient formellement les caractères avec une cédille, quand en roumain on utilise une virgule souscrite.
- ISO 8859−3 : latin 3 : Espéranto et Maltais. Initialement, c'était prévu pour les langues d'Europe du Sud, y compris le Turc, les langues d'Espagne (donc Catalan, Basque), le Français. Mais dans la pratique, c'est limité à ces deux langues.
- ISO 8859−4 : latin 4 : langues baltes (incomplet). Pas utilisé dans la pratique, remplacé par Latin 7 (iso−8859−13), ou plutôt en pratique la codepage 1257 Windows. Pour l'Este ou Estonien, on peut aussi utiliser Latin 9.
- ISO 8859−9 : latin 5 (comme latin 1, moins l'Islandais, plus le Turc). Usage : le Turc et peut−être l'Azéri, mais il manque le schwa, un « e » renversé. Codage officiel aux Pays−Bas du fait des Turcs qui y habitent.
- ISO 8859−10 : latin 6 : langues nordiques, sauf le Lapon Skolt. Pas utilisé dans la pratique : ne servirait que pour les langues sames (ou lapones), mais il manque des caractères, et le public concerné est de toute manière assez limité.
- ISO 8859−11 : Thai. Pas encore officiellement adopté.
- ISO 8859−12 : Indien (n'existe pas encore). Prévu pour le devanagari, pour écrire l'Hindi, la langue principale de l'Inde, et aussi un grand nombre d'autres langues indiennes, dont le Sanskrit.

- ISO 8859−13 : latin 7 : langues baltes. Plus exactement : couvre les langues du pourtour de la mer Baltique, donc aussi le Polonais, l'Allemand et les langues de scandinavie. Mais dans la pratique il est réduit en utilisation aux langues baltes et à l'Este ou Estonien.
- ISO 8859−14 : latin 8 : langues celtes. Pour le Breton, le Gaélique et les dialectes parlées au Pays de Galles et en Irlande.
- ISO 8859−15 : latin 9 : comme latin 1, en y comblant quelques manques. Pour le Français, le Finnois (en concurrence avec Latin 1) et l'Este ou Estonien (en concurrence avec Latin 7).

De cette longue liste, il convient de ne retenir que :

- ISO 8859−1 latin 1 : police ouest−européenne utilisée par la majorité des systèmes UNIX, elle permet de disposer de tous les caractères Français sauf les « o e liés », l'euro et l'y tréma majuscule
- cp850 : équivalent sous DOS de cette police, mais incompatible : il faut utiliser un outil de traduction de fichiers tel GNU recode.
- ISO 8859−15 latin 9 : police de l'union européenne, il s'agit d'une version révisée de l'ISO 8859−1 latin 1 pour inclure tous les caractères des pays européens ; ainsi on y retrouve les caractères Français manquant, l'euro, et des caractères finlandais manquants.
- latin 0 : l'ISO 8859−15 latin 9 est souvent incorrectement abrégée en latin0, qui correspondait au nom du projet, de par son ambition à remplacer le latin 1 un jour. Je vous renvoie au site de l'ISO et à la documentation complète de ce standard
- cp1252 : équivalent sous Windows de cette police, aussi incompatible, qui apporte les mêmes caractères manquants, du fait d'une révision récente de cette norme.
- ISO 8859−9 latin 5 : police officielle en Turquie et en Hollande, où les caractères Islandais peu utilisés ont été remplacés par les caractères turcs.
- cp1254 : équivalent sous Windows de cette police, aussi incompatible, qui apporte les caractères manquant au Français précédemment cités et remplace les caractères Islandais peu utilisés par les caractères turcs.

Les choix de polices de caractères sont souvent des questions ou la raison se heurte aux habitudes.

Pour ma part, si vous hésitez, je vous recommande l'ISO 8859−15 latin 9, pour diverses raisons :

- Il supporte l'intégralité des caractères utilisés en Français de manière standard sur les UNIX, pour les échanges de fichiers entre Solaris et Linux par exemple
- Il s'agit de la police 8 bits choisie par l'union européenne si l'Unicode, évolution du 8 bits, n'est pas disponible
- Il permet d'écrire des textes avec l'intégralité des caractères utilisés dans l'union
- Il est en train de devenir le nouveau standard dans la messagerie électronique au sens large, supplantant progressivement l'ISO 8859−1 latin 1

Le Francophones HOWTO utilise donc le latin 9 dans tous les exemples, car il y a de grandes chances que votre système ne soit pas encore compatible latin 9... autant donc vous proposer des manipulations et des travaux pratiques utiles :−)

Toutefois, si vous ne pouvez utiliser l'ISO 8859−15 latin 9 et décidez d'en choisir une autre, je vous recommande d'éviter les cp propres au monde DOS/Windows et de vous rabattre sur l'ISO 8859−1 latin 1 qui est encore un standard fort répandu : dans chaque exemple je vous invite alors à remplacer les « latin9 » par des « latin1 »

3.4. Table de clavier

Une table de clavier est un fichier permettant de faire correspondre l'empreinte physique des cabochons, « touches » du clavier, au résultat obtenu sur l'écran de l'ordinateur.

Pour linux, ces fichiers sont dans /usr/lib/kbd/keytables/ ou /usr/share/kbd/keytables/

Par exemple, les claviers français AZERTY ont une table de clavier différente des claviers suisse romands QWERTZ, ou des claviers du reste du monde, QWERTY.

Cela ne concerne que l'apparence du clavier, l'empreinte des cabochons, donc disposition relative des caractères sur le clavier, et non son électronique interne ; si votre clavier ne vous convient pas, il vous suffit donc d'utiliser des autocollants pour le transformer en un autre clavier francophone.

3.5. « Home », « End », « Page_Up » et « Page_Down »

Les touches « Home », « End », « Page_Up » et « Page_Down » dont les noms varient selon les claviers sont les touches permettant respectivement d'aller en début/fin de ligne et de monter/descendre d'une page.

Les codes affectes à ces touches sont respectivement 102 et 107 pour « Home » et « End » en mode terminal.

Dans les pictogrammes des claviers ISO 9995−7, ces touches sont représentés par des flèches aux traits appuyés.

Lorsqu'on presse une touche, le clavier émet un code, qui est intercepté par la table de clavier.

La table de clavier, si elle reconnait le code, renvoie la chaine associée à la touche, en l'occurence« "\e[4 » pour « End ».

Lorsque bash ou un autre programme reçoit cette chaine, il ne la connait pas !

Donc il faut lui dire, dans le .inputrc que "\e[4 " signifie "aller à la fin de la ligne"

Donc pour que ces touches fonctionnent dans le shell bash, on ajoutera donc dans le fichier .inputrc (dans votre / ou dans /etc/inputrc) les lignes:

```
set meta−flag on
set convert−meta off
set input−meta on
set output−meta on
set bell−style visible

"\e[1~":beginning−of−line
"\e[3~":delete−char
"\e[4~":end−of−line
"\e\C−h": backward−kill−word
"\e\e[3~": kill−word
```

Les premières lignes permettent d'afficher les caractères 8 bits, et de remplacer le « bip » en cas d'erreur par un éclair blanc à l'écran.

3.6. Touches mortes

Les touches mortes sont des touches qui ne produisent pas de caractères en elles−mêmes, mais qui modifient la façon dont la frappe suivante va être interprétée, par exemple, le tréma ou l'accent circonflexe sont en général des touches mortes.

Une touche morte est une touche qui doit donc être suivie d'un espace pour obtenir l'accent seul, ou dans le cas où elle est suivie d'une voyelle y positionne un accent, par exemple :

```
        « ^ » puis « e » donne « ê »
        « ^ » puis « espace » donne « ^ »
```

Les touches mortes sont :

- be et fr: l'accent aigu en AltGr de 1 pour les Français et en AltGr de ù pour les Belges, l'accent grave en AltGr de 7 pour les français & en AltGr de carré/cube pour les Belges, le tréma et l'accent circonflexe tous deux à côté du P
- sf et usx: les accent aigus, graves, circonflexes, les trémas et la tilde
- qc: les accent graves, circonflexes, les trémas et la cédille

3.7. « BackSpace » et « Delete »

En raison d'une erreur volontaire et historique sous Linux, pour « émuler » le fonctionnement d'un clavier de console VT, ces 2 touches sont fréquemments mélangées, au mieux fonctionnant à l'identique, au pire ne fonctionnant pas du tout.

Commençons donc par bien les définir :

- BackSpace : touche placée en haut à droite du pavé principal, au dessus d' « Entrée », au cabochon marqué d'une flèche vers la gauche.
- Delete : touche du pavé curseur au cabochon marqué « Suppr » ou « Del », à gauche de « Fin » ; en ISO 9995−7 elle correspond à trois barres obliques fines orientées sud−ouest/nord−est.

Le pavé numérique offre une touche au marquage identique et à effet identique lorsque « Verr Num » ou « Num Lock » est désactivé.

Ces deux touches correspondent chacune à un de ces codes :

- BS : caractère ASCII 0x08, ou control−h.
- DEL : caractère ASCII 0x7F, ou control−?
- ^D : caractère ASCII 0x04, ou control−d.

Ceci dans le but d'effacer à gauche du curseur avec BackSpace, à droite du curseur avec Delete.

3.8. « UTC » et « GMT »

Chaque partie du monde a sa propre norme horaire, basée sur des fuseaux.

La France se trouve en zone MET, « heure moyenne européenne » soit GMT+1 ; la zone « MET DST » correspond à l'heure d'été activée soit GMT+2.

- GMT est l'heure solaire moyenne de Greenwich. L'heure est comptée à partir de midi et est donc décalée de 12 h par rapport au temps universel. GMT est basée sur la rotation de la terre, ce n'est donc pas un temps régulier. En effet, la vitesse de rotation de la Terre subit des variations faibles mais assez chaotiques, en plus elle diminue sur le long terme.
- UTC, « temps universel coordonné », est compté à partir de minuit toujours par rapport au méridien zéro. Il est basé sur des étalons atomiques, mais des secondes intercalaires sont ajoutées occasionnellement pour faire en sorte qu'il ne s'écarte jamais de plus de une seconde de l'heure solaire moyenne. Le site de l'opdaf et le lexique donnent plus de renseignements.

C'est le temps UTC, défini à Paris, qui sert de référence aux différentes heures légales.

L'acronyme « GMT » est employé 99% du temps pour désigner UTC, mais c'est une erreur ou un abus de langage.

Enfin, à titre purement indicatif, il existe l'heure astronomique, aussi comptée sur des étalons atomiques mais décalée de 4 minutes par jour de l'heure terrestre car se basant sur la position de la terre dans l'espace par rapport aux étoiles.

4. Réglage du clavier sous Linux

4.1. Introduction

Il existe deux programmes pour configurer votre clavier : un pour la console : *loadkeys* et un pour XFree : *xkbd*.

Pour installer un clavier français sous Linux, tapez selon votre clavier une des lignes suivantes :

```
        loadkeys tables-clavier/fr/fr-latin9.map
        loadkeys tables-clavier/qc/qc-latin1.map
        loadkeys tables-clavier/cn/cn-latin1.map
        loadkeys tables-clavier/be/be-latin9.map
        loadkeys tables-clavier/sf/sf-latin1.map
        loadkeys tables-clavier/usx/usx-latin1.map
```

```
        loadkeys tables-clavier/usx/usx-happy-hacking-latin9.map
```

4.2. Attention au stty !

Avec les tables latin 9 recommandées dans ce HOWTO, pour linux−console aussi bien que pour X window, *N'UTILISEZ PLUS* :

```
        stty erase ^H
```

ou :

```
        stty erase ^?
```

« stty erase » ne sert en effet qu'à établir une correspondance entre le code que renvoie une touche et la fonction *erase* pour effacer le caractère.

Les tables ici fournies fonctionnent correctement et ne nécessitent donc pas de *stty erase* qui risquerait surtout de perturber leur bon fonctionnement !

Supprimez−donc ces lignes de vos fichiers d'initialisation de l'interpréteur de commandes « shell », comme .bashrc, .bash_profile ou .tcshrc .

4.3. Faire de ces modifications les défauts au démarrage

D'habitude, loadkeys est exécuté par des scripts au démarrage ; par exemple dans */etc/rc.d/init.d/keytable* ou */etc/rc.d/rc.local*, *init.d/keytable* ou *rc.keymap*.

La commande :

```
        find /etc/rc.d -type f | xargs grep loadkeys
```

vous fournira le nom exact du fichier lançant loadkeys ; il vous suffit alors de l'éditer pour que vos modifications soient désormais prise en en compte à chaque démarrage.

Les distributions RedHat et Debian constituent des exceptions : la table de clavier par défaut étant définie dans */etc/sysconfig/keyboard* pour la RedHat et */etc/kbd/config* pour les Debian.

Remplacer donc dans le fichier propre à votre distribution la table de clavier en question par la bonne table comme fr−latin9.map (selon votre modèle de clavier)

Vous devrez mettre une copie de la table que vous utilisez dans le répertoire où votre distribution va chercher les tables de clavier ; en général /usr/lib/kbd/keymaps/i386/azerty/ pour une RedHat et /usr/share/kbd/keymaps/i386/azerty/ pour une Debian.

Allez donc sur télechargez l'archive, enregistrez−la dans votre répertoire /, puis détarrez−la avec la commande:

```
        tar -xzvf francophones-howto.tgz
```

Cela crée un répertoire /french : placez−vous dans ce répertoire puis tapez :

```
        cp tables-clavier/linux-console/fr-latin9.map /usr/lib/kbd/keytables/i386/azerty/
```

Pour afficher les caractères latin9, vous devrez exécuter les instructions de la section « polices de caractère ».

4.4. Polices de caractère

Si vous utilisez console−tools−1998.08.11.tar.gz, vous pourrez remplacer « setfont » par « consolechars » : ces programmes servent à charger une police sous linux console.

```
        setfont /usr/lib/kbd/consolefonts/xxxE-NN.psf.gz
```

Je vous conseille pour des raisons développées plus bas :

```
        setfont /usr/lib/kbd/consolefonts/lat9-16u.psf.gz
```

Je ne vous conseille pas les fichiers « lat1 », qui ne supportent pas l'euro ni les autres caractères français.

Vous trouverez aussi des fichiers « .uni » qui ne sont utiles que pour créer des polices ou pour ceux qui utilisent l'unicode : ils fixent des correspondances entre les « dessins » des caractères dans la police et les codes des dits caractères, un même dessin pouvant correspondre à plusieurs codes dans l'example de d'unicode.

Pour les polices toutes faites, vous pouvez forcer cette correspondace avec l'option « −u fichier.uni ».

Si vous tenez à les installer, copiez de la même manière tous les fichiers commençant par « lat9 » et se terminant par « .uni » dans le répertoire /usr/lib/kbd/consoletrans avec la commande:

```
   cp lat9*.uni /usr/lib/kbd/consoletrans
```

D'habitude, setfont est exécuté par des scripts au démarrage ; par exemple dans */etc/rc.d/init.d/keytable* ou */etc/rc.d/rc.local*, *init.d/keytable* ou *rc.keymap*.

La commande :

```
        find /etc/rc.d −type f | xargs grep setfont
```

vous fournira sur le nom exact du fichier lançant setfont ; il vous suffit alors de l'éditer pour que vos modifications soient désormais prise en en compte à chaque démarrage...

Les distributions RedHat et Debian constituent des exceptions : la police par défaut étant définie dans */etc/sysconfig/i18n* pour la RedHat et */etc/kbd/config* pour les Debian.

Dans */etc/sysconfig/i18n* rajouter une ligne de la forme :

```
        SYSFONT=xxxE−NN.psf
```

- xxx représente le type de police ; il doit s'agir d'une lat pour les francophones ce qui signifie ISO 8859 ou latin. Sans cela il n'y a plus de caractères « étendus » comme les voyelles accentués, symboles de ponctuation comme paragraphe...
- E représente l'encodage latin, par exemple 1 pour latin1, 0 ou 9 pour latin9
- NNN représente la taille individuelle de chaque caractère ; 08 est presque illisible mais 16 est un peu gros... À régler selon les goûts de chacun.

Je vous conseille la police lat9−16u.psf ou son équivalent lat0−16.psf installée par défaut sur la plupart des distributions, très souvent compressée donc avec le suffixe « .gz » .

[rajouter une explication sur comment installer la police]

La section « une police, des polices » vous donnera plus de détails sur les polices disponibles.

4.5. Vieilles versions

Si lors du chargement de la table de clavier vous rencontrez un message d'erreur similaire au suivant :

```
(guylhem@victis:guylhem)$ loadkeys fr-latin9.map
Loading /usr/lib/kbd/keytables/fr-latin9.map
unknown keysym 'dead_cedilla'
/usr/lib/kbd/keytables/fr-latin9.map:67: parse error
syntax error in map file
key bindings not changed
```

Cela signifie que votre version de kbd dont le programme loadkeys dépend est trop vieille : il faut vous procurer une version plus récente du programme.

Il existait et existe encore des versions des tables de clavier pour ceux ne voulant ni touches mortes ni accents, mais celles−ci tombent actuellement en désuétude puisque la grande majorité des programmes prennent désormais en charge ces fameuses touches mortes et les caractères accentués.

De plus, les tables fournies avec ce HOWTO permettent de se servir des touches mortes ou de s'en passer si un programme les refuse, les deux cas ayant été prévus :

4.6. Améliorations spécifiques au clavier français latin9

Ça y est, vous pouvez taper du texte latin 9 sous Linux en mode console !

Seules 2 touches ont été modifiées :

4.6.1. a) carré/cube

Elle sert maintenant à oe/OE liés ; en AltGr on y trouve les guillemets français.

Le carré & le cube restent respectivement accessibles en AltGr (ù) et Alt Gr(*), les guillemets étant aussi directement accessibles avec AltGr (w) et AltGr (x), selon la norme ISO 9995 part 3 appliquée à la norme AZERTY française.

En résumé :

- AltGr (S) : «
- AltGr + Shift (S) : »
- AltGr (w) : «
- AltGr (x) : »
- AltGr (m) : ¹
- AltGr (ù) : ²
- AltGr (*) : ³

4.6.2. b) dollar / livre / symbole monétaire international

Étant donné que le symbole monétaire international n'est plus disponible dans la norme d'encodage ISO 8859−15, il a été remplacé par la division centésimale de l'euro : le cent, disponible donc en AltGr ($).

Dans les précédentes versions, le dollar & la livre, des monnaies étrangères, avaient été déplacés pour laisser la place à l'euro et au cent, notre monnaie.

De telles modifications étaient permises par les recommandations officielles de l'EURO WORKSHOP mais allaient à l'encontre de la norme AZERTY : si la position AltGr (e) est déjà occupée sur un clavier, la position de l'euro est laissée au libre choix des programmeurs.

AltGr (e) sert à faire « ê » depuis 1992 sous Linux, mais devant les protestations légitimes d'utilisateurs disposant d'un clavier où l'euro est sérigraphié en AltGr (e), j'ai du déplacer le ê en AltGr + Shift ($).

Désolé d'avoir rompu la « tradition » linuxienne pour conserver la compatibilité à la norme AZERTY & aux recommandations officielles de l'EURO WORKSHOP !

En résumé :

- AltGr (e) : euro
- AltGr ($) : cent
- AltGr + Shift ($) : ê

4.7. Pour les autres claviers latin9

L'euro & les cents ont été rajoutés respectivement en AltGr (e) & en AltGr (c).

La touche AltGr a donc été rajoutée ; pour utiliser ces fonctions avec toute table de clavier, il suffit de charger other−latin9.map après votre table nationale.

La plupart des améliorations du clavier français ont été transposées aux claviers nationaux par des auteurs externes.

Toutefois, je ne possède pas chaque modèle de clavier et ne peut expliquer en détail toutes ces modifications ; des sections spécifiques seront donc rajoutées par la suite par d'autres auteurs.

4.8. Dans tous les cas

« Home », « End », « Delete » et « BackSpace » ont été corrigées & les touches « Windows » ont été correctement définies, pour passer d'une console virtuelle à l'autre.

Des « Composes », séquences à préfixer par la touche compose (« ImprÉcran » ou « Print Screen » par défaut) ont été rajoutées, mais cela reste transparent pour l'utilisateur.

Par exemple:

- COMPOSE ^ suivi d'une des lettres (s,S,z,Z) sert à obtenir les lettres finlandaises rajoutées
- COMPOSE " suivi d'une des lettres (y,Y) sert à obtenir les y trémas franco−néerlandais rajoutés
- COMPOSE − suivi d'une des lettres (e,c,l,y) sert à obtenir l'euro, le cent, la livre, le yen

5. XFree

5.1. Introduction

Il y a quelques années, « xmodmap » servait à changer les tables de clavier sous XFree, un fichier de configuration d'ordinaire nommé /usr/X11/lib/X11/xinit/.Xmodmap étant pris en charge grâce à votre *.xinitrc* ou */etc/X11/xinit/xinitrc*

Pour convertir une table console correcte, on utilisait « mk_modmap » livré avec l'archive kbd et généralement installé en /usr/lib/kbd/keytables

Les utilisateurs désirant personnaliser leur table xmodmap employaient le programme « xkeycaps ».

Puis XFree 2.1 et les version ultérieures ont reconnu la disposition des touches du clavier gérée par « loadkeys », rendant théoriquement xmodmap inutile.

Mais la gestion de la touche « COMPOSE » assurée par XFree 3.1 laissait un peu à désirer, la table héritée de « loadkeys » ne servant guère.

Il fallait donc continuer à employer xmodmap, l'utilitaire logiciel standard permettant d'obliger le serveur X à associer les événements souris/clavier à des caractères.

Sous XFree version 3.2 et postérieures, « xkb » rendit xmodmap définitivement obsolète.

Mais ne voilà−t−il pas les rumeurs qui voulaient que les prochaines versions de XFree ne supporte plus les touches mortes se sont confirmées !!!

Dans la distribution X11 du X Consortium, donc XFree86, qui en est directement issu, il n'existe plus de mécanisme général de gestion des touches mortes, ou des solutions approximatives obligeant à supprimer aussi xkb ou modifier des bibliothèques de programmes !

5.2. Mal programmer

À chaque programme de bien gérer les touches mortes au lieu de faire confiance à X pour cette tâche.

En réalité, il n'est pas compliqué de gérer des touches mortes, il suffit d'utiliser la bonne fonction, XmbLookupString() dans les programmes Motif, au lieu de XLookupString().

5.3. Le clavier en pratique

Quelle histoire compliquée !

De ce fait, elles ne fonctionnent que pour les clients X « internationalisés », c'est−à−dire qui gèrent eux−mêmes les problèmes liés aux méthodes d'entrée de données des différentes langues nationales.

Ceci est très gênant lorsqu'on veut pouvoir saisir du texte comportant des lettres accentuées, alors que le client utilisé n'est pas internationalisé !

5.3.1. a) Solution utilisant xmodmap

Taper :

```
        cp fr−latin9.xmod /usr/X11R6/lib/X11/xinit/Xmodmap
        cp fr−latin9.xmod ~/.Xmodmap
```

5.3.2. b) Solution utilisant xkb (recommandée)

Taper :

```
        cp fr-latin9 /usr/X11R6/lib/X11/xkb/symbols
```

Puis éditer votre *XF86Config* pour y rajouter dans la section « Keyboard » les lignes xkb idoines :

```
Section "Keyboard"
Protocol        "Standard"
AutoRepeat      500 30
LeftAlt         Meta
RightAlt        ModeShift               # Important pour Alt-gr, mais dans les
                                        # versions récentes de XFree, AltGr et
                                        # RightAlt sont synonymes.

ScrollLock      Compose                 # Pour faire des caractères spéciaux.
                                        # Utiliser sinon une des 3 touches des
                                        # nouveaux claviers 105 touches...

RightCtl        Control                 # Garder la touche normale

XkbKeycodes     "xfree86"
XkbRules        "xfree86"
XkbTypes        "default"
XkbCompat       "default"
XkbSymbols      "fr-latin9(pc105)"
XkbGeometry     "pc(pc105)"
XkbModel        "pc105"

EndSection
```

Pour cette partie Xkb, certains préfèrent :

```
        XkbKeymap       "xfree86(fr-latin9)"
        XkbKeycodes     "xfree86"
        XkbSymbols      "en_US(pc101)+fr-latin9"
        XkbGeometry     "pc(pc101)"
```

Bien sur, remplacer « fr » par « cn », « be », « ch » ou « usx » selon votre clavier...

5.3.3. c) Solution du compose

Une autre méthode pour obtenir des caractères 8 bits :

Dans */usr/X11R6/lib/X11/locale/iso8859−1/Compose* se trouve une table de « correspondance » entre deux caractères et un résultat ; les deux caractères sont «mélangés» lors de l'appui sur la touche Compose : par exemple;

```
        COMPOSE suivi de « e » puis « ^ »
```

a toutes les chances de vous donner ê, y compris dans l'ordre opposé (« ^ » puis « e »)

Mais pour utiliser la touche Compose, vous devez l'affecter à une touche !

Le paramétrage préconisé, réglé dans XF86Config, la fait correspondre à la touche « Arrêt Défil » inutile sous X, par la commande :

```
        ScrollLock        Compose
```

dans la section Keyboard.

Attention, sous Linux en mode console, cette touche est utile (pour bloquer temporairement la console virtuelle) et c'est « Impr Écran » ou « Print Screen » qui, ne servant à rien, est affectée à « Compose ».

5.3.4. d) Solution modifiant la libX11

Thomas Quinot <Thomas.Quinot(à)cuivre.fdn.fr> vous propose donc sa *libX11* modifiée, qui gère les touches mortes de manière transparente pour tous les clients.

Elle permet également de traiter les séquences composées inconnues comme des paires de caractères, par exemple :

```
        ~ + / = ~/
```

Pour l'utiliser :

- Récupérez libX11−XF3.3.1.tar.gz ;
- Extrayez le fichier *libX11.so.6.1* ;
- Placez−le dans */usr/X11R6/lib/* ;
- Exécutez *ldconfig* ;
- Désactivez l'extension XKB en utilisant l'option *XkbDisable* dans la section *Keyboard* de votre *XF86Config*.

5.3.5. e) Est−ce nécessaire ?

Personnellement, je vous recommande de ne pas toucher à la libX11, d'oublier xmodmap et de l'alternative xkb + fichiers latin9 en attendant un autre changement de la politique du X Consortium envers les touches mortes (ou en rajoutant les options d'internationalisation au code source de vos programmes.)

Actuellement, si vous n'avez pas de répertoire */usr/X11R6/lib/X11/xkb/symbols/* cela signifie que vous n'utilisez pas xkb : procurez−vous donc une version plus récente de X window !

Sans xkb, XFree se rabattra par défaut sur votre configuration de clavier pour Linux avec loadkeys.

Toutefois, la translation n'est pas parfaite et je vous recommanderais plutôt d'éviter cette option, aussi bien que l'ancien gestionnaire de clavier *Xmodmap*.

Il se peut aussi que sans aucune commande pour xkb, XFree vous considère comme utilisateur d'un clavier 7 bits !

Il convient donc d'utiliser une table adaptée avec xkb.

5.4. « latin9 » ou « latin1 » ?

Je vous conseille de choisir les tables *latin9* que je maintiens à jour, plus récentes que les versions *latin1*, permettant d'utiliser de multiples améliorations, détaillées dans la section clavier sous linux−console, pour disposer du plus de confort possible et plus adaptées car maintenu par des francophones.

Les nombreuses améliorations dont elles bénéficient permettent de pallier au déficiences de nombreux programmes.

Par exemple, certains programmes en Motif comme Netscape (tm) ne savent pas encore gérer les touches mortes ... donc impossible de taper par exemple des ê ou des î sur les claviers des Belges et Français, puisque la touche morte « accent circonflexe » est le seul moyen d'obtenir ces caractères !

« fr−latin9 » répond à ce problème en proposant une solution de remplacement : AltGr (voyelle) permet d'obtenir la voyelle avec un accent grave, AltGr+Shift (voyelle) permet d'obtenir la voyelle avec un tréma !

5.5. Table de correspondances

- AltGr (voyelle) : voyelle accent grave
- AltGr + Shift (voyelle) : voyelle tréma
- AltGr (touche au dessous de la voyelle) : voyelle majuscule accent grave
- AltGr + Shift (touche au dessous de la voyelle) : voyelle majuscule tréma
- AltGr + Shift (minuscule accentuée) : majuscule accentuée

Par exemple :

- AltGr (a) : â
- AltGr (q) : Â
- AltGr + Shift (ù) : Ù

L'astuce marche aussi pour le c cédille :

* AltGr + Shift (ç) : Ç

Donc même dans les pires conditions, si aucune touche morte ne fonctionne, on peut quand même accéder à toutes les voyelles accentuées, majuscules et minuscules !

Vous y trouverez en plus les guillemets français (« »), les symboles employés en physique (ß, å, delta, phi ...), des signes de monnaies (yen, cent) ou de copyright (*tm*, ©) ainsi que des lettres d'autres langues (Ø, Æ ...).

Tous ces signes s'obtiennent grâce à AltGr (lettre), par exemple AltGr (b) donne ß, AltGr (r) donne *tm*, AltGr (y) donne yen...

Pour vous rapeller simplement de tout celà, tapez par exemple :

```
        more /usr/lib/kbd/keytables/i386/azerty/fr−latin9.map
```

Pour ce qui est du choix entre latin1 et latin9... à vous de voir !

Un conseil : essayez l'un puis l'autre si vous avez le temps.

5.6. Les polices ISO 8859−15 latin 9 sous X

Il est bien d'avoir une table de clavier compatible latin 9, mais sans police adaptée jamais vous ne verrez les caractères rajoutés dans cette norme.

Il faut donc utiliser l'utilitaire de conversion « bdftopcf » qui en quelque sorte permet de compiler les polices de caractères fournies.

Placez−vous dans le répertoire de l'archive que vous avez récupéré puis sous root, taper par exemple pour Xlat9−10x20.bdf ou un autre bdf :

```
        bdftopcf Xlat9−10x20.bdf > Xlat9−10x20−lat9.pcf
        gzip Xlat9−10x20−lat9.pcf
        mv Xlat9−10x20−lat9.pcf.gz /usr/X11R6/lib/X11/fonts/misc
        mkfontdir /usr/X11R6/lib/X11/fonts/misc
```

Celà crée un fichier « Xlat9−10x20−lat9.pcf » qu'il faut compresser avec gzip, on obtient alors le fichier « Xlat9−10x20−lat9.pcf.gz » qui doit être mis dans le répertoire adéquat, pour lequel l'index des polices de caractères sera mis à jour.

Alors, ajoutez la table de composition latin9 aux compositions par défaut de X, spécifiques à l'iso 8859−1 :

```
        cat XCompose−lat9 >> /usr/X11R6/lib/X11/locale/iso8859−1/Compose
```

Nous faisons un cat vers l'ISO 8859−1, car X ne reconnaît pas l'ISO 8859−15 latin9.

De là, redémarrer X ou taper sous son compte d'utilisateur normal :

```
        xset +fp /usr/X11R6/lib/X11/fonts/misc
```

Si vous utilisez xmodmap, lancez :

```
        xmodmap ~/.Xmodmap
```

Pour choisir les polices latin9, lancer xfontsel et chercher les encodages iso8859−15.

Ou plus simplement, taper :

```
        grep Xlat9−10x20.pcf /usr/X11R6/lib/X11/fonts/misc/*
```

Puis utiliser le second paramètre comme nom de police, par exemple avec rxvt, xiterm, aterm ou wterm:

```
        aterm −fn −misc−fixed−medium−r−normal−−20−200−75−75−c−100−iso8859−15
```

Cela lance un terminal X utilisant une des polices latin9.

6. Réglage du clavier pour les applications en mode texte

6.1. Tout d'abord

Le huitième bit doit survivre à l'entrée du noyau, assurez−vous−en donc avec :

```
        stty cs8 −istrip −parenb
```

Ensuite, il convient de s'assurer que l'application est apte à supporter les caractères 8 bits : par exemple, ispell n'est pas compilé pour des caractères 8 bits par défaut : il faut le recompiler sans l'option :

```
        -NO8BIT
```

6.2. Une police, des polices...

Avant d'aborder cette section, je vous conseille de lire l'introduction sur les polices pour appréhender un peu mieux le vocabulaire de base et les différentes normes de polices.

6.2.1. a) Historique des polices

De l'ASCII à l'Unicode, en passant par l'ISO 8859, le latin et le cp, on peut résumer l'histoire des polices en grandes étapes :

Au début était l'ASCII, qui codait ses caractères sur 7 bits.

Prévu pour des états−uniens, il était impossible d'incorporer des caractères accentués à la norme déjà présente de 127 caractères.

Ainsi vint le 8 bits qui étendit le nombre de caractères à 255 et les normes ISO 8859, latin et cp, pour les plus connues, qui vinrent pallier à cette déficience du fait des 128 positions supplémentaires disponibles en encodant les caractères sur 8 bits.

Mais plusieurs de ces normes étaient nécessaire du fait de l'impossibilité de faire tenir les caractères de toutes les langues du monde sur 8 bits.

La norme ISO 8859−1 latin 1, la plus utilisée, se destine par exemple au langues d'Europe de l'ouest, du nord, d'Amérique, la norme ISO 8859−2 latin 2 est prévue pour l'Europe de l'est...

L'Unicode est l'un de ces standard, destiné à terme à remplacer les différentes évolutions de l'ASCII ; comme le passage du 7 bits au 8 bits il correspond à une extension de la place disponible, tout en reprenant l'ancienne partie comme ASCII et ISO 8859−1 latin 1 pour rester compatible avec la majorité du parc installé.

L'Unicode offre l'avantage de proposer les caractères nécessaires à toutes les langues du monde, mais reste encore peu utilisé ou implémenté.

En effet, il est impossible de l'utiliser en pratique sous linux−console, la mémoire exigüe de la carte vidéo ne permettant pas de stocker tous les caractères de l'Unicode, et sous X la gestion actuelle des polices rendrait son utilisation trop inconfortable.

Attendons un peu que les polices « True Type » se répandent sous X, grâce à des outils comme freetype, xfstt (...) et le très attendu XFree 4 qui devrait les supporter en natif.

Actuellement, la nouvelle norme européenne ISO 8859−15 latin 9 est censée permettre la transition vers l'Unicode en apportant à l'ancien ISO 8859−1 latin 1 les caractères qu'il manquait actuellement.

Toutefois elle est partiellement incompatible avec l'Unicode puisque l'ISO 8859−1 latin 1 avait été choisi comme base pour l'Unicode, mais que cette police veut compléter les failles de l'ISO 8859−1 latin 1 en y rajoutant des caractères déjà présents dans l'Unicode...

Quelle complexité pour de simple lettres !

6.2.2. b) ISO 8859−15 latin 9

De nous jours, il vaut mieux installer par défaut la nouvelle norme ISO 8859−15 latin 9, destinée à remplacer l'ensemble des polices européennes par une seule.

Son but est voisin de l'Unicode, mais elle a l'avantage d'être plus légère que celui−ci et d'être applicable tout de suite.

Elle apporte notamment des caractères qui manquaient beaucoup pour les Français et les Finlandais :

- « e dans l'o » en majuscule et minuscule pour les Français, par exemple pour les mots Suf, cSur, bSuf, Sil ...
- « s chapeau inversé » en majuscule et minuscule pour les Finlandais
- « z chapeau inversé » en majuscule et minuscule pour les Finlandais
- « l'EURO » pour tous les pays européens

Cette nouvelle norme, ISO 8859−15 latin 9 aussi connue sous le nom générique latin0, reste cependant encore très peu répandue par rapport à la norme actuelle ISO 8859−1 latin 1.

6.2.3. c) Utiliser les polices ISO 8859−15 latin 9

Vous n'avez besoin que d'une seule police !

Or il en existe plusieurs types répondant différemment à la même norme iso−8859−15 latin9 « latin0 » sous linux :

- par tradition l'opposition latN/latNu selon l'inclusion ou non d'une table de correspondance Unicode.
- en réalité les différences vont bien plus loin, de l'ordre dans lequel sont les caractères de la police, ce qui joue sur l'aspect des applications semi graphique, les correspondances avec des caractères latin1 ± latin9 ± Unicode ce qui a un rôle dans la compatibilité croisée ou exclusive, par exemple purement Unicode...

J'ai donc complété par 2 types de ma création pour répondre au mieux aux différents besoins.

	Encodage dans l'ordre iso	Inclusion d'une table Unicode	Compatibilité exclusive	Compatibilité croisée
lat9	oui	non	non	lat1 + lat9
lat9u	non	oui	non	lat1 + lat9
lat9v	oui	oui	Unicode	non
lat9w	oui	oui	non	lat1+lat9+Unicode

Voici plus d'explications :

- les lat9 tout court ne contiennent pas de table de correspondance Unicode, ce qui fait que tous les symboles monétaires internationaux apparaîtront comme des euro, ce qui peut poser des problèmes pour les fichiers issus d'un ordinateur sous windows ; le cp1252 ou le latin1 sont « incompatibles » avec le latin9, mais ce dernier a l'avantage de fonctionner tout de suite sous linux. Dans les versions récentes du noyau, selon le site de Yann Dirson, il semble donc que toutes les polices devraient contenir une table de correspondance Unicode...
- des lat9u encodées dans un ordre non standard, mais identique à celui des polices latXu de kbd, par rapport aux polices précédentes, en amenant les mêmes problèmes avec le cp1252 et le latin1 mais en incluant une table de correspondance Unicode, ce qui permet d'avoir de jolis caractères semi graphiques sous Yast, mc, ou d'autres programmes basés sur ncurses ou dialog.
- des lat9v encodées dans l'ordre officiel iso comme les lat9 mais avec une table Unicode complète : là aucun symbole monétaire international ne sera remplacé par l'euro ! Ces polices sont utiles pour éviter les problèmes d'export, grâce à l'Unicode *pur*, avec une table de clavier Unicode adaptée : tous les caractères latin9 sont remplacés par des caractères Unicodes et affichés ainsi, ce qui a le mérite d'être compatible avec le latin9, l'Unicode, d'être élégant techniquement mais inutilisable « normalement » sous linux−console. Presque aucun système n'est entièrement prêt pour l'Unicode !

Donc n'utilisez les polices lat9, lat9u ou lat9v que si vous savez ce que vous faites, je vous recommande pour ma part le meilleur compromis : les polices lat9w qui permettent de passer au latin9 immédiatement, en posant quelques problèmes pour les caractères latin1 qui seront remplacés par leurs équivalents latin9, mais sans rendre l'Unicode obligatoire comme les lat9U, sans emmêler les caractères graphiques comme les lat9, et en permettant aussi d'afficher les caractères spécifiques au latin9 en Unicode *pur*, à l'exception des caractères spécifiques au latin1 donc...

Je vous rassure, vous n'êtes pas obligés de comprendre tout ce qui suit pour « passer à l'euro », si effectivement vous ne savez toujours pas de quelle police vous avez besoin, un conseil : utiliser les lat9w.

Pour utiliser la police lat9−16.psf ou un autre lat9 .psf, vous devez pallier l'absence d'une table Unicode en en chargeant une.

```
loadunimap fonts/linux-console/lat9.uni
setfont fonts/linux-console/lat9-16.psf
```

Pour utiliser la police lat9w−16.psf ou une autre lat9u, lat9v ou lat9w :

```
setfont /fonts/linux-console/lat9w-16.psf
```

6.2.4. d) Unicode

Une alternative est l'Unicode, mais les programmes l'utilisant sont encore trop peu nombreux pour qu'elle soit valable.

Citons quand même yuedit et le projet « 9 » (9term, 9wm... rien à voir avec l'opérateur télécom français du même nom) qui utilisent l'Unicode sous X, avec plus ou moins de succès.

En revanche, sous linux−console, tout programme peut utiliser l'Unicode.

Mais, hélas!, aussi grande que soit la mémoire des cartes vidéos actuelles, elle ne peut charger l'intégralité des caractères d'une police Unicode.

Il existe donc des « jeux de caractères » limités mais bien utiles pour les langues baltiques, asiatiques, est−africaines...

6.2.5. e) Utiliser les possibilités Unicode

Essayez par exemple ce fichier *unicode−lance* pour passer en mode Unicode:

```
#!/bin/sh
echo −e '\033%8'
kbd_mode −u
loadkeys fr-unicode.map
setfont lat9u-16.psf
```

et ce fichier *latin−retourne* pour revenir en mode latin normal:

```
#!/bin/sh
echo −e '\033%@'
kbd_mode −a
loadkeys fr−latin9.map
setfont lat9w−16.psf
```

Essayez par exemple de lire le fichier *unicode.txt*, encodé en Unicode, une fois *unicode−lance* activé.

Vous ne verrez sinon qu'un texte mal encodé, sans caractères 8 bits, comme cet exemple :

```


Ceci est un essai de texte en UNICODE !

On constate que les bÃ©tas (Ã~_) et autres lettres 8 bits ne sortent pas bien
si l'on regarde ce fichier en mode latin !
Les accents Ã©ÃšÃ¦ Ã¹ non plus d'ailleurs...
Enfin, heureusement que l'on ne marche pas sur des Å~RUFS, enfin, Å~Sufs !
Ã~Ga marche moyennement disons !

```

6.3. Les locales, messages en Français

Linux en 'version française' existe... mais ne semble pas facile à obtenir.

La librairie locale(7) [cf aussi perllocale, setlocale, getlocale] utilisée par la librairie C se sert de variables d'environnement pour définir les préférences linguistiques et nationales.

Pablo Saratxaga répondait ainsi à Pascal Rigaux en prenant l'exemple de sort :

```
 PR> J'ai découvert aujourd'hui la variable d'environnement LANG. C'est assez
 PR> surprenant de se retrouver avec de l'aide en français (même si elle n'est pas
 PR> toujours aussi précise que la version anglaise) quand on fait ls --help.

C'est agréable n'est-ce pas ?

 PR> Par contre je ne comprend pas pourquoi un programme comme sort(1) n'utilise pas
 PR> cette information pour obtenir un tri acceptable pour les accents (du genre ne
 PR> pas mettre « être » après « zen »). Il devrait au moins proposer une option de
 PR> ce genre.

Il faudrait regarder les sources de sort...
La fonction à utiliser est strcoll() :

                                ****

STRCOLL(3)         Manuel du programmeur Linux         STRCOLL(3)


NOM
       strcoll - Comparaison de deux chaînes suivant la localisa-
       tion en cours.

SYNOPSIS
       #include <string.h>

       int strcoll (const char *s1, const char *s2);

DESCRIPTION
       La fonction strcoll() compare les deux chaînes s1  et  s2.
       Elle renvoie un entier inférieur, égal ou supérieur a zéro
       si  s1 est respectivement inférieure, égale, ou supérieure
       a  s2.   La  comparaison est effectuée en se basant sur la
       localisation en cours pour la catégorie LC_COLLATE.  (Voir
       setlocale(3)).

                                ****

qui est donc l'équivalent de strcmp() mais qui tiens compte des locales.
Il faudra s'écrire un strcasecoll() aussi pour bien faire, en utilisant
toupper()/tolower() qui d'après la page de manuel supporte les locales.

Qui s'y colle ? Ça devrait être assez simple à réaliser sur un système GNU
en tout cas.
```

Il existe différente variables à exporter, avec chacune une fonction spécifique ; le format standardisé est:

```
        langue[_PAYS[.CHARSET]][@variante]
```

Les crochets dénotent l'optionalité, par exemple: 'fr', 'fr_BE', 'fr_CH.ISO−8859−15', no@bokmaal, no@nynorsk,...

Les fonctions de la libc, celle de GNU en tout cas, iront chercher dans « l'ordre décroissant » si on peut dire; par exemple si on spécifie 'fr_CH.ISO−8859−15' elles chercheront d'abord avec cette valeur, puis avec 'fr_CH' puis avec 'fr'.

- LC_COLLATE définit les équivalences de caractères pour les comparaisons (æ peut être équivalent à ae), pour les ligatures et pour les césures.
- LC_CTYPE définit les caractères affichables
- LC_MONETARY définit le format et le symbole de la monnaie utilisée
- LC_NUMERIC définit le format numérique : regroupement, marqueur décimal...
- LC_MESSAGES définit la langue des messages
- LC_TIME définit le format de la date, les noms des jours et des mois
- LC_ALL valeur par défaut des variables précédentes : si une LC_ n'est pas définie, LC_ALL est prise en compte, sinon la libc se rabat sur LANG.
- LANG différent des variables précédentes, contient le code langue au format iso : fr,en,de...
- LANGUAGE liste des locales par ordre de préférence séparées par deux points (fr:es:dk:en), c'est une particularité GNU, fort utile si un document n'existe que dans une langue comme certaines pages de manuel

Les valeurs utilisables pour les francophones sont:

- fr français générique
- fr_FR français de France
- fr_BE français de Belgique
- fr_CH français de Suisse
- fr_LU français du Luxembourg
- fr_CA français du Canada

Par exemple, si vous être un Français de France, il suffit d'indiquer sous bash :

```
        export LANGUAGE=fr_FR
```

Exemple :

```
        bash# export LANGUAGE=es_ES
        bash# ls fichier_n_existant_pas
        ls: fichier_n_existant_pas: No existe el fichero o el directorio
        bash# export LANGUAGE=de_DE
        bash# ls fichier_n_existant_pas
        ls: fichier_n_existant_pas: Datei oder Verzeichnis nicht gefunden
        bash# export LANGUAGE=en_US
        bash# ls fichier_n_existant_pas
        ls: fichier_n_existant_pas: No such file or directory
        bash# export LANGUAGE=fr_FR
        bash# ls fichier_n_existant_pas
        ls: fichier_n_existant_pas: Aucun fichier ou répertoire de ce type
```

Toutefois, avec le passage à l'euro, ces locales ne sont plus adaptées.

Je vous conseille donc d'utiliser dans les pays de l'union passant à l'euro le suffixe « @euro » :

```
   export LC_ALL="fr_FR@euro"
```

Pour cela, installer les nouveaux fichiers sources de locales dans */usr/share/i18n/* en lançant :

```
   cp charmaps/ISO-8859-15 /usr/share/i18n/charmaps/ISO-8859-15
   cp locales/* /usr/share/i18n/locales/
```

Alors, régénérer les locales « binaires » de */usr/share/locales*, par exemple :

```
   localedef -i fr_FR@euro -f ISO-8859-15 fr_FR@euro
```

fr_FR@euro inclus les caractères latin 9, c'est utile pour l'utilisation de LC_COLLATE et LC_CTYPE en fait, et la nouvelle monnaie unique.

Pour le moment, le Franc Français (FRF) est toujours la monnaie officielle en France par exemple, mais lorsque ce sera l'euro, il suffira de remplacer la locale fr_FR par fr_FR@euro.

```
   cd /usr/lib/locales
   rm -fr fr_FR
   mv fr_FR@euro fr_FR
```

La solution « @euro » est supportée par les autres UNIX comme Solaris.

Pour les shell scripts, il peut être utile de rajouter « LC_TIME=C » pour empêcher la date d'être localisé, embêtant pour les grep et autres qui cherchent « Mon » « Tue »... et non « Lun », « Mar »...

Les données correspondantes aux préférences se trouvent généralement dans */usr/share/locale*. S'ils manquent, vous pouvez récupérer leur source

Concernant les autres fichiers abordés dans cet article, vous les trouverez avec les autres fichiers de cet HOWTO sur ma page.

6.4. Son linux en français

Pour se faire *SON* linux en français il faut en pratique disposer des sources de la libc et d'un minimum de compétences en langage C ; ces manipulations sont inutiles pour l'utilisateur standard de Linux qui se contentera d'utiliser les locales de la section précédente.

Mais voici donc, pour la bonne bouche, quelques renseignements collectés...

Laurent Chemla nous pilote :

```
Si j'ai suivi, il y a deux trucs qui font la même chose, d'où embrouille, les
locales et un truc appelé nls.

Les locales, sous Linux, ça va par défaut dans /usr/lib/locale,
et ça contient de quoi préciser entre autres le format des
nombres (LC_NUMERIC), les retours des fonctions ctype (isalpha etc) (LC_CTYPE),
le format des sommes d'argent (LC_MONETARY), le format des dates (LC_TIME) et
encore d'autres détails. Sur certains Unix, il y a aussi LC_MESSAGES,
qui contient les messages de l'OS dans la langue choisie par la variable
d'environnement LC_MESSAGES. Pas sous Linux.  Sous Linux, c'est nls qui fait
ça, et de fait, nls gère un fichier qui est dans /etc/locale/.
(NDG : De nos jours, ils se trouvent plutôt dans /usr/lib/locale)
D'où l'embrouille. Ils auraient mis ça dans /usr/lib/nls, comme
tout le monde, ça aurait évite la confusion. Mais bon.

Donc, si l'on veut se tenir à jour d'nls, il suffit d'aller dans les
sources de libc/nls, faire « make french » et copier libc.dat dans
/etc/locale/C/libc.dat et toc, tous les messages sont en français. Y compris
ceux des programmes qui n'utilisent pas setlocale(): c'est perror() qui fait le
boulot (du moins je suppose).

Si l'on veut en plus que les programmes qui utilisent setlocale() utilisent nos
règles françaises, là il faut aller dans les sources de
libc/locale et dans chacun des sous−répertoires jeter un Sil sur le
fichier exemple, le modifier pour le français (c'est des tout petits
fichiers), et faire « make », avant de copier le résultat dans
/usr/lib/local/fr/ et de faire un « export LC_ALL=fr ».
```

Autre astuce pour ménager les vieux logiciels :

```
cd /usr/lib
ln −sf /usr/share/locale .
```

Ne pas oublier de télécharger sur le site du lip6 :

- Les pages de manuel en français, archive man−fr−*, adaptées par C. Blaess. Il suffit d'extraire le contenu de l'archive dans le répertoire /usr/man/fr.
- Une intéressante documentation technique : locale.fr, par P. D'Cruze, adaptée par É. Dumas

Pour en apprendre davantage lire la page de manuel de « locale » : bibliothèque de pages de manuel 5 ou 7 :

```
        bash# man 5 locale
        bash# man 7 locale
```

6.5. Les variables

Ces variables doivent être exportées à chaque login : mettez les donc dans *.bashrc* et *.bash_login*, *.tcshrc* ou /etc/profile :

```
# pour tcsh, utiliser setenv au lieu de export, par exemple «setenv LANG fr»

# Les locales
LC_CTYPE=ISO−8859−1
LANGUAGE=fr
LC_MESSAGES=fr
LC_ALL=fr
LANG=fr
LESSCHARSET=latin1
export LC_CTYPE LANGUAGE LC_MESSAGES LC_ALL LANG LESSCHARSET

# Les raccourcis : sans ll ou d, impossible de voir les fichiers comprenant des
# caractères 8 bits ...
alias q="cd .."
alias ll="ls −−color=auto −a −N −l"
alias d="ls −−color=auto −a −N"
alias indent="indent −kr"
alias netscape="export LC_ALL=en_EN ; netscape $* ; LC_ALL=fr_FR"

# Au cas où Del/BackSpace ne fonctionne pas, essayez l'un des deux :
#stty erase ^?
#stty erase ^H
```

N'oubliez pas : tous les fichiers ici cités dont le nom commence par « . » doivent se trouver dans votre répertoire maison /

6.6. Les caractères 8 bits dans les programmes utilisant readline tels bash ou ncftp...

Readline est une librairie de saisie de caractères avec historique et complétion ; elle est utilisée par de nombreux programmes comme bash, ncftp, gnuplot...

Par défaut, le 8e bit sert à coder la touche Meta.

Elle est utilisée pour coder certaines commandes mais la touche « Escape » ou « Échap » peut la remplacer.

Si vous souhaitez voir et saisir des accents avec les applications compilées avec readline, comme bash pour les versions supérieures à 1.13, ajoutez ceci dans votre fichier $INPUTRC, par défaut *Ü/.inputrc*) :

```
# Permettre de rentrer & recevoir des caractères accentués
set meta-flag on
set convert-meta off
set input-meta on
set output-meta on

# Pas de bip audible mais visible
# set bell-style visible

# Pour faire marcher Home, End, Delete, Esc + Delete & Esc + BackSpace
"\e[1Ü":beginning-of-line
"\e[3Ü":delete-char
"\e[4Ü":end-of-line
"\e\C-h": backward-kill-word
"\e\e[3Ü": kill-word
```

Cela vous permettra d'effacer des caractères vers l'avant avec « Delete » et vers l'arrière avec « BackSpace », ceci n'étant pas prévu sur les tables de clavier par défaut « mimant » le comportement d'un clavier de console VT100 .

Les deux dernières options vous permettront d'utiliser « Home » et « End » pour aller au début et à la fin de la ligne en cours d'édition.

Pour pouvoir employer une touche il suffit d'obtenir son code et l'associer à une fonction grâce à ce fichier ; pour obtenir le code d'une touche invoquer cat puis appuyer sur la touche : son code apparaît.

Le code physique :

```
        ^[
```

correspond à la séquence d'échappement.

Par exemple, si le code de « Home » est « ^[[7 » il suffit d'ajouter une ligne :

```
        "\e[7~": beginning-of-line
```

au *.inputrc*.

Plus de documentation est disponible sur la page de manuel de bash, section *READLINE*.

On peut aussi, grâce à la variable d'environnement *INPUTRC*, utiliser un fichier commun à tous les utilisateurs.

Enfin, je vous recommande d'utiliser ce fichier *.bashrc* à placer dans votre répertoire maison :

```
# ~/.bashrc : exécuté par bash(1) pour les shells lancés après le login.

# Le path
PATH="/usr/sbin:/usr/bin:/sbin:/bin:/usr/X11R6/bin:/usr/local/bin"

# Le prompt
PS1="(\u@\h:\W)$ "

# L'utilisateur
ENV=$HOME/.bashrc
USERNAME="votre nom"

# Les locales
_CTYPE=ISO-8859-1
LANGUAGE=fr
LC_MESSAGES=fr
LC_ALL=fr
LANG=fr
LESSCHARSET=latin1
export LC_CTYPE LANGUAGE LC_MESSAGES LC_ALL LANG LESSCHARSET

# Les raccourcis
alias q="cd .."
alias ll="ls --color=auto -a -N -l"
alias d="ls --color=auto -a -N"
alias indent="indent -kr"
alias elm-box="elm -f ~/Mail/Inbox"
alias netscape="export LC_ALL=en_EN ; netscape $* ; LC_ALL=fr_FR"

# Au cas où Del/BackSpace ne fonctionne pas, essayez l'un des deux :
#stty erase
#stty erase ^H

# Refuser le talk & le write
#mesg n

# Permettre les coredumps
#ulimit -c nombre_max_de_Ko

#Fixer le masque de création de fichiers
#umask 022
```

Le fichier .bash_profile ou .bash_login, exécuté par bash(1) lors des logins, y est strictement identique mais peut, si vous le voulez, présenter des différences pour afficher par exemple une citation en laçant le programme « fortune ».

6.7. zsh

Celui−ci devrait se contenter d'un *stty pass8*.

Si cela ne suffisait pas, rajouter dans le *.zshrc* ou dans */etc/zshrc*:

```
        bindkey "\e[2~"  yank
        bindkey "\e[3~"  delete-char
        bindkey "\e[1~"  beginning-of-line
        bindkey "\e[4~"  end-of-line
        bindkey "\e[5~"  up-line-or-history
        bindkey "\e[6~"  down-line-or-history
```

6.8. tcsh

Pour tcsh, placer, dans le fichier */etc/csh.login* ou dans les fichiers *.tcshrc* des utilisateurs concernés, les variables indiquées plus haut en remplaçant export par setenv:

```
        setenv LC_CTYPE=ISO-8859-1
        setenv LANGUAGE=fr
        setenv LC_MESSAGES=fr
        setenv LC_ALL=fr
        setenv LANG=fr
        setenv LESSCHARSET=latin1
```

6.9. Pour tous les shells

Remplacer 'iso_8859_1' par 'iso−8859−1' si cela ne fonctionne pas, voire french.iso88591.fr_FR ou fr_BR, fr_CA, fr_CH, fr_LU...

À propos de LANG, il vaut mieux éviter fr_FR et lui préférer fr car certains programmes ne passent pas vraiment par la resolution de locales mais lisent LANG directement...

Si nls est installé, les routines correspondantes sont utilisées, sinon tcsh agit en iso_8859_1, quelle que soit les valeurs données à LANG et LC_CTYPE : cf. la section NATIVE LANGUAGE SYSTEM de tcsh(1).

Selon le Danish−HOWTO, utiliser :

```
        (guylhem@victis:tcsh)# setenv LC_CTYPE ISO-8859-1
        (guylhem@victis:tcsh)# stty pass8
```

6.10. L'heure

Rien n'est plus facile que de laisser à Linux le soin de passer de l'heure d'été à l'heure d'hiver et réciproquement...

a) D'abord, quelle heure est−il :−) ?

Après avoir regardé sur la pendule la plus proche de vous, tapez :

clock

puis

date

La première heure est celle du bios, la seconde celle calculée par Linux à partir de */etc/zoneconfig*.

Pour peu que vous ayez installé timezone, votre machine peut jongler aisément entre les différents fuseaux horaires... ce que les DOS ou Windows ne permettent pas !

b) L'heure du choix !

De là, deux possibilitées : soit vous décidez de laisser l'heure système à l'heure locale, peu pratique, excepté si vous hébergez aussi sur votre disque dur ces systèmes d'exploitation peu évolué, ne sachant même pas exploiter les fuseaux horaires, soit vous passez à l'heure de Greenwich, Linux se chargeant de l'adapter à votre fuseau horaire.

Dans ce cas, les grands voyages à l'étranger avec un portable tout neuf équipé de Linux se feront en toute simplicité : le dernier paragraphe vous expliquera comment changer de fuseau horaire facilement...

c) Le réglage

Pour procéder, regardez d'abord si l'heure système retournée par clock est l'heure locale ou l'heure de Greenwich.

Vous avez bien regardé une pendule comme je vous le conseille plus haut, n'est−ce pas ?

Si clock vous retourne l'heure locale, il va falloir jongler un peu avec les fuseaux horaires, sinon passez directement à l'étape « réglage du fuseau ».

- Pour les français, belges wallons et suisses romands :

Vous êtes normalement dans une zone horaire MET (Medium European Time, soit GMT+1).

La zone « MET DST » correspond elle à l'heure d'été active (GMT + 2).

Selon les décisions futures de la cour européenne, l'heure d'été « daylight savings » risque d'être abandonné... ce qui compliquera cette explication !

- Pour les canadiens français :

Vous êtes dans une zone horaire GMT−4 si vous habitez autour de Montréal .

Pour les autres provinces, consulter un dictionnaire !

Premièrement, effacez le fichier /etc/localtime ou /usr/lib/zoneinfo/localtime (l'emplacement varie selon les distributions ; si vous avez les deux fichiers, effacez /etc/localtime : l'autre fichier doit certainement être un lien vers /etc/localtime).

Dès lors, clock et date doivent vous retourner la même heure...

- Pour les canadiens français : utilisez GMT+4

- Pour les français, belges wallons et suisses romands :

Si vous lisez ce document en été, le décalage de l'heure locale par rapport à l'heure de Greenwich est de deux heures, on va donc mettre temporairement le fuseau en GMT−2, soit vous lisez ce document en hiver et il convient de remplacer tous les « 2 » par des « 1 » dans les exemples suivants...

GMT−2 ?!!? Alors que le fuseau est GMT+2 ?

Ceci va vous retourner l'heure de Greenwich dont on a besoin pour régler le système : tapez :

ln −sf /usr/lib/zoneinfo/Etc/GMT−2 /etc/zoneconfig

clock

date

L'heure calculée correspond donc à l'heure de Greenwich, l'heure système à l'heure locale... soit l'inverse de ce que l'on veut.

Un simple : *clock −w*

Va alors mettre l'heure du système en heure de Greenwich, utilisant pour cela l'heure locale inversée volontairement avec l'heure système.

d) Régler le fuseau horaire :

Tapez alors : *ln −sf /usr/lib/zoneinfo/Europe/Votre−ville /etc/zoneconfig*

Et voilà ! Le système est à l'heure de Greenwich, la conversion vers le fuseau horaire local et l'heure d'été se faisant automatiquement.

Si vous habitez hors Europe, procédez de même en remplaçant les « −2 » par « X », X représentant l'opposé de votre décalage horaire (par exemple −4 au Québec, donc X=+4...)

e) Modifier le fuseau horaire :

Si vous partez à l'étranger, il est alors simple de modifier l'heure locale de votre portable : par exemple si vous partez pour Moscou :

ln −sf /usr/lib/zoneinfo/Europe/Moscow /etc/zoneconfig

Très simple et très utile : plus besoin de se souvenir du décalage horaire (plus ou moins X heures) puisque Linux se charge désormais de tout !

6.11. vi

Normalement, aucune modification n'est nécessaire à part l'utilisation de loadkeys (détaillée plus haut).

Ce fichier *.vimrc* pourrait toutefois vous être utile si vous utilisez *vim* au lieu d'*elvis* :

```
" Les guillemets anglais « " » signifient « mis en commentaire »

" Éviter un avertissement « version incorrecte » :
version 4.0

" Utiliser les défauts de vim, bien mieux
set nocompatible

" Autoriser le « BackSpace » sur tout :
set bs=2

" Police à utiliser sous X11 :
"set guifont=-misc-fixed-medium-r-normal--14-130-75-75-c-70-iso8859-1

" Fixer la césure automatique de la ligne à N caractères :
set tw=72

" Faire apparaître les tabulations et les espaces
set list

" Changer les défauts pour voir les espaces inutiles et les tabulations
set listchars=tab:»·,trail:·
hi nontext ctermfg=red
hi nontext ctermbg=blue

" Indentation (pour les programmeurs) :
" set cindent

" Réglages souris :
"set mouse=a

" Ligne
set nowrapscan

" Montrer les correspondances :
set showmatch
```

```
" Montrer le mode
set showmode

" Indentation intelligente (pour les programmeurs) :
" set uc=0

" Faire fonctionner BackSpace :
set t_kD=^?

" Doit se trouver après
map ^H X

" Faire fonctionner Delete  :
map \e[3~ x

" Cacher la souris lorsqu'on tape :
set mousehide

" Faire voir les correspondances lors de recherches :
" set hlsearch

" Colorer à l'intérieur des commentaires en C :
let c_comment_strings=1

" Couleur pour xiterm, rxvt, nxterm, color-xterm :
if has("terminfo")
    set t_Co=8
    set t_Sf=\e[3%p1%dm
    set t_Sb=\e[4%p1%dm
 else
    set t_Co=8
    set t_Sf=\e[3%dm
    set t_Sb=\e[4%dm
endif

" Coloration syntaxique :
if ett_Co > 1
   syntax on
endif

" Autoriser l'édition de fichiers gzippés

augroup gzip
  " Supprimer toutes les autocommandes
  au!
  autocmd BufReadPre,FileReadPre        *.gz set bin
  autocmd BufReadPost,FileReadPost      *.gz '[,']!gunzip
  autocmd BufReadPost,FileReadPost      *.gz set nobin
  autocmd BufReadPost,FileReadPost      *.gz execute ":doautocmd BufReadPost " .  expand("%:r")

  autocmd BufWritePost,FileWritePost    *.gz !mv <afile> <afile>:r
  autocmd BufWritePost,FileWritePost    *.gz !gzip <afile>:r

  autocmd FileAppendPre                 *.gz !gunzip <afile>
  autocmd FileAppendPre                 *.gz !mv <afile>:r <afile>
  autocmd FileAppendPost                *.gz !mv <afile> <afile>:r
  autocmd FileAppendPost                *.gz !gzip <afile>:r
augroup END

augroup bzip
  au!
  autocmd BufReadPre,FileReadPre        *.bz2 set bin
  autocmd BufReadPost,FileReadPost      *.bz2 '[,']!bunzip2
  autocmd BufReadPost,FileReadPost      *.bz2 set nobin
```

```
  autocmd BufReadPost,FileReadPost      *.bz2 execute ":doautocmd BufReadPost " .  expand("%:r")
  autocmd BufWritePost,FileWritePost    *.bz2 !mv <afile> <afile>:r
  autocmd BufWritePost,FileWritePost    *.bz2 !bzip2 <afile>:r

  autocmd FileAppendPre                 *.bz2 !bunzip2 <afile>
  autocmd FileAppendPre                 *.bz2 !mv <afile>:r <afile>
  autocmd FileAppendPost                *.bz2 !mv <afile> <afile>:r
  autocmd FileAppendPost                *.bz2 !bzip2 <afile>:r
augroup END

augroup cprog
  au!
  autocmd BufRead *       set formatoptions=tcql nocindent comments&
  autocmd BufRead *.c,*.h set formatoptions=croql cindent comments=sr:/*,mb:*,el:*/,://
augroup END
```

6.12. Emacs

Voici un fichier de configuration de base pour employer les accents.

```
  ;;
  ;; Fichier .emacs: initialisation d'emacs
  ;; Tiré du Guide du Rootard
  ;;

  (display-time)                     ;; Pour avoir l'heure dans la barre d'état
  (setq display-time-24hr-format t)  ;; Format 24 heures

  ;; Nouveaux modes
  (autoload 'c++-mode   "cplus-md" "C++ Editing Mode" t)
  (autoload 'perl-mode   "perl-mode" "Perl Editing Mode" t)
  (autoload 'c-mode "c-mode" "C Editing Mode" t)
  ; mieux vaudrait utiliser le "cc-mode"

  (autoload 'jargon-mode "jargon" "Jargon Mode" t)

  ;; Auto-Mode Settings : positionne le mode selon l'extension
  (setq auto-mode-alist
  (append '(("\.c$"  . c-mode)    ;; utilise le mode C++ même pour C
  ("\.h$"  . c-mode)
  ("\.C$"  . c++-mode)
  ("\.H$"  . c++-mode)
  ("\.cc$" . c++-mode)
  ("\.C$"  . c++-mode)
  ("\.pl$" . perl-mode)           ;; Perl
  ("/tmp/snd\.[0-9]*" . text-mode);; Text (pour le courriel)
  ("[Rr][Ee][0-9]*" . text-mode)
  ("\.ada$"  . ada-mode)          ;; Ada
  ("\.spec$"  . ada-mode)
  ("\.body$"  . ada-mode)
  ("makefile$" . makefile-mode)  ;; Makefile
  ("Makefile$" . makefile-mode)
  ("Imakefile$" . makefile-mode))
  auto-mode-alist))

  ; Remappages variés à mettre ici
  (global-set-key "\eg" 'goto-line)       ;; ESC G = Aller à une ligne

  (put 'eval-expression 'disabled nil)
```

```
;; Accents...
(standard-display-european 1)
(load-library "iso-syntax")
(set-input-mode (car (current-input-mode))
              (nth 1 (current-input-mode))
              0)

;; Sous X-Window, texte en couleurs (C/C++/Shell/Makefile,etc)
(cond (window-system
(setq hilit-mode-enable-list  '(not text-mode)
hilit-background-mode   'light
hilit-inhibit-hooks     nil
hilit-inhibit-rebinding nil)
(require 'hilit19)
))
(if (not (equal window-system ""))
(global-set-key "\C-?" 'delete-char))
))
```

Le Keyboard−HOWTO conseille quand à lui :

```
          (standard-display-european t)
          (set-input-mode nil nil 1)
          (require 'iso-syntax)
          (load-library "iso-insert.el")
          (define-key global-map [?\C-.] 8859-1-map)
```

Utilisez une version au moins égale à 19.27, modifiez le fichier global (commun à tous les utilisateurs) */usr/lib/emacs/site−lisp/site−start.el* ou le *./emacs* de chaque utilisateur concerné pour y ajouter :

```
        (standard-display-european t)
        (set-input-mode (car (current-input-mode))
          (nth 1 (current-input-mode))
          0)
        (global-set-key [delete]   'delete-char)
        (global-set-key [home]     'beginning-of-line)
        (global-set-key [end]      'end-of-line)
        (global-set-key [prior]    'scroll-down)
        (global-set-key [next]     'scroll-up)
        (global-set-key [C-right]  'forward-word)
        (global-set-key [C-left]   'backward-word)
        (global-set-key [C-prior]  'beginning-of-buffer)
        (global-set-key [C-next]   'end-of-buffer)
        (global-set-key "\033[A"  'previous-line)
        (global-set-key "\033[B"  'next-line)
        (global-set-key "\033[C"  'forward-char)
        (global-set-key "\033[D"  'backward-char)
        (global-set-key "\033[H"  'beginning-of-line)
        (global-set-key "\033[Y"  'end-of-line)
        (global-set-key "\033[1~" 'beginning-of-line)
        (global-set-key "\033[2~" 'overwrite-mode)
        ;; le delete-char peut correspondre à \004 ou à \033[3~
        (global-set-key "\033[3~" 'delete-char)
```

```
        (global-set-key "\033[4~" 'end-of-line)
        (global-set-key "\033[5~" 'scroll-down)
        (global-set-key "\033[6~" 'scroll-up)
```

Pour les autres versions :

- 19.19 :

  ```
          (standard-display-european 1)
          (set-input-mode (car (current-input-mode))
            (nth 1 (current-input-mode))
            0)
  ```

  Lire à ce propos le fichier emacs.info.
- 19.22 :

  ```
          (load-library "iso-transl")
          (standard-display-european t)
  ```

- Autres :

  ```
          (standard-display-european 1)
          (load-library "iso-transl")
  ```

XEmacs les accepte lui par défaut sans broncher mais peut des fois rencontrer des problèmes avec les BackSpaces dans le mini−buffer en console.

Ajouter dans le *.emacs* :

```
(if (eq window-system 'x)
    (global-set-key (read-kbd-macro "DEL") 'delete-char)
  (or (global-set-key "^[[3~" 'delete-char))
)
```

Le mode ispell d'emacs a un certain nombre de dictionnaires prédéfinis qui ne sont pas forcement ceux qui sont en place sur le système.

Si vous constatez des problèmes, vous devez redéfinir la liste ispell−dictionary−alist des dictionnaires.

Cette liste contient :

- le nom du dictionnaire
- la liste des caractères composant un mot
- la lise opposée des caractères ne composant pas un mot
- les caractères de liaison à l'intérieur des mots ("−", ""...)

- vrai ou faux si ces caractères peuvent être présents en plusieurs exemplaire dans le mot
- une liste d'arguments pour ispell
- le mode d'ispell (tex, nroff..)

Par exemple, dans le fichier *.emacs* :

```
(setq ispell-dictionary-alist
 '((nil  ; francais.aff
  "[A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü]" "[^A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü]"
  "[---']" nil ("-n") "~nroff")
 ("english"  ; rosbif
  "[A-Za-z]" "[^A-Za-z]" "[---']" nil ("-B") nil)
 ("american" ; yankee
  "[A-Za-z]" "[^A-Za-z]" "[---']" nil nil nil)
 ("francais" ; français
  "[A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü]" "[^A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü]"
  "[---']" nil ("-n") "~nroff")
 ("francais-TeX" ; français
  "[A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü\\]" "[^A-Za-zÀÂÇ-ËÎÏÔÙÛÜàâç-ëîïôùûü\\]"
  "[---'^`\"]" t nil "~tex")
 ("espanol" ; espa~nol.aff
  "[A-Za-záéíóúüñÁÉÍÓÚÜÑ]" "[^A-Za-záéíóúüñÁÉÍÓÚÜÑ]"
  "[---'^`\"]" t nil "~nroff")
 ("dansk" ; danois et norvégien
  "[A-Za-zåÅøØæÆéÉèÈ]" "[^A-Za-zåÅøØæÆéÉèÈ]"
  "[---'^`\"]" t nil "~nroff")
 ))
```

Le dictionnaire "francais−TeX" est un lien symbolique sur "francais" qui permet d'ajouter une entrée pour les accents à la TeX (ex: \'e).

Si vous utilisez X11, vous voudrez peut−être reconstruire les menus et il vous faudra recharger une partie de lisp/loaddefs.el dans votre *.emacs* :

```
(setq ispell-menu-map nil)

  (if ispell-menu-map-needed
  (let ((dicts (reverse (cons (cons "default" nil) ispell-dictionary-alist)))
    name)
    ;; Can put in defvar when external defines are removed.
    (setq ispell-menu-map (make-sparse-keymap "Spell"))
    (while dicts
  (setq name (car (car dicts))
    dicts (cdr dicts))
  (if (stringp name)
  (define-key ispell-menu-map (vector (intern name))
    (cons (concat "Select " (capitalize name))
  (list 'lambda () '(interactive)
    (list 'ispell-change-dictionary name))))))))

  (if ispell-menu-map-needed
  (progn
    ;; Define commands in opposite order you want them to appear in menu.
```

```
    (define-key ispell-menu-map [ispell-change-dictionary]
  '("Change Dictionary" . ispell-change-dictionary))
    (define-key ispell-menu-map [ispell-kill-ispell]
  '("Kill Process" . ispell-kill-ispell))
    (define-key ispell-menu-map [ispell-pdict-save]
  '("Save Dictionary" . (lambda () (interactive) (ispell-pdict-save t))))
    (define-key ispell-menu-map [ispell-complete-word]
  '("Complete Word" . ispell-complete-word))
    (define-key ispell-menu-map [ispell-complete-word-interior-frag]
  '("Complete Word Frag" . ispell-complete-word-interior-frag))
    (define-key ispell-menu-map [ispell-continue]
  '("Continue Check" . ispell-continue))
    (define-key ispell-menu-map [ispell-region]
  '("Check Region" . ispell-region))
    (define-key ispell-menu-map [ispell-word]
  '("Check Word" . ispell-word))
    (define-key ispell-menu-map [ispell-buffer]
  '("Check Buffer" . ispell-buffer))
    (define-key ispell-menu-map [ispell-message]
  '("Check Message" . ispell-message))
    (define-key ispell-menu-map [ispell-help]
  '("Help" . (lambda () (interactive) (describe-function 'ispell-help))))
    (put 'ispell-region 'menu-enable 'mark-active)))

  (fset 'ispell-menu-map (symbol-value 'ispell-menu-map))
```

6.13. less

/etc/profile ou les fichiers de démarrage du shell (comme *.bashrc* et *.bash_profile* dans votre répertoire maison) doivent contenir :

```
export LESSCHARSET=latin1
```

Comme fichier *.lessrc* je vous recommande :

```
\e[B    forw-line
\e[A    back-line
\e[6~   forw-scroll
\e[5~   back-scroll
\e[1~   goto-line
\e[4~   goto-end
\e[C    next-file
\e[D    prev-file
\eOA    back-line
\eOB    forw-line
```

Pour rentrer un Esc (parfois nommé Échap) taper « \e » ou, sous vi, mettez vous en mode insertion avec i puis tapez « ctrl+v » et enfin « Esc ».

Ce fichier permet d'utiliser les touches « Home », « End », « Page_Up » & « Page_Down ».

Pour s'en servir, taper :

```
lesskey -o .less .lessrc
```

N'oubliez pas de mettre le fichier .less dans votre répertoire maison : il permet d'employer les touches fléchées, « Page_Up », « Page_Down », « Home » et « End ».

Ne pas négliger la page de manuel de less car il peut être intéressant de changer la valeur de sa variable d'environnement de configuration ("LESS").

J'utilise : '−C −M −i −x2'

6.14. ls

Utilisez les options :

```
-N --color=auto
```

ou, plus simplement, définissez dans les fichiers de démarrage de votre shell les alias suivants permettant de voir les fichiers dont le nom comprend des caractères 8 bits :

```
alias ll="ls --color=auto -N -l -a"
alias d="ls --color=auto -N"
```

Un simple *d* remplacera le classique *ls*, en y ajoutant la couleur, alors que *ll* vous offrira une liste complète de tous les fichiers présents dans le répertoire, avec leurs attributs, leurs tailles...

−−color=auto est préférable car −−color=yes fera avoir des codes escape lorsqu'on redirige vers un fichier ou un programme où si le terminal ne supporte pas la couleur.

6.15. flex

Donnez l'option :

```
-8
```

si l'analyseur généré doit accepter les entrées 8−bits (bien sûr qu'il doit le faire !).

6.16. elm

Rajoutez ces trois lignes dans votre *.elm/elmrc* :

```
charset = iso-8859-1
displaycharset = iso-8859-1
textencoding = 8bit
```

6.17. col

Assurez−vous :

1) qu'il a été corrigé et fasse un

```
setlocale(LC_CTYPE,"");
```

2) de définir

```
LC_CTYPE=ISO-8859-1
```

dans l'environment.

6.18. rlogin

Utilisez l'option :

```
-8
```

6.19. joe

Utiliser joe 2.8 ou plus récent qui devraient fonctionner après édition du fichier de configuration : placer l'option −asis dans */usr/lib/joerc* en première colonne.

6.20. TeX et LaTeX

Le plus simple consiste à employer GuTemberg, une distribution de LaTeX avec tous les défauts réglés pour des francophones par des francophones !

Mais si vous préférez une version *standard* non modifiée, il suffit d'ajouter au début de chaque fichier :

- Pour LaTeX:

  ```
  \documentstyle[isolatin]{article}
  ```

- Pour LaTeX2e:

  ```
  \documentclass[12pt,a4paper]{letter}
  \usepackage{isolatin1}
  \usepackage[french]{babel}
  \usepackage{t1enc}
  ```

  ou alors :

  ```
  \usepackage[latin1]{inputenc}
  \usepackage[T1]{fontenc}
  ```

Au cas où votre distribution soit trop vieille pour l'inclure, *isolatin.sty* est disponible séparément

Pour gs et xdvi, il faut utiliser des options spéciales (sur ma machine, ce sont des alias). En effet, ils sont souvent configurés pour un papier états−unien dit « légal », de taille proche, mais non exactement égale, à celle du standard « A4 ».

Ces options sont les suivantes :

gs −sPAPERSIZE=a4 xdvi −paper a4 ghostview −a4

Pour que dvips convertisse les documents dans un format papier a4, il faut spécifier dans le fichier config.ps (le chemin varie en fonction des versions de LaTeX) : /usr/lib/texmf/dvips/config.ps ou *.dvipsrc*

@ a4 210mm 297mm @+ ! %%DocumentPaperSizes: a4 @+ %%PaperSize: a4 @+ %%BeginPaperSize: a4 @+ a4

Sinon ajoutez à votre *.Xresources* :

```
XDvi.paper:             a4
Ghostview.pageMedia:    A4
```

6.21. Manuel

Tout d'abord, il faut vous procurer les pages de manuel Linux traduites en français

Si vous utilisez une distribution RedHat ou Debian, celles−ci sont présentes sur le cédérom : il suffit de les sélectionner lors de l'installation.

De là, deux possibilités :

- soit vous ne voulez que des manuels en Français, auquel cas il faut faire un :

  ```
  export MANPATH=/usr/man/fr_FR
  ```

  NB: fr_FR se remplace par fr_BE, fr_CA, fr_CH ou fr_LU selon le pays concerné.
- soit vous préférez par défaut des manuels en Français et si la page n'existe pas, vous vous rabattez vers l'Anglais, il suffira alors de taper :

  ```
  export LANG=fr
  ```

man ira chercher dans le « MANPATH » des pages en « LANG » par défaut et se rabattra sinon sur des pages en anglais.

Si vous désirez installer la page de manuel expliquant la norme ISO 8859−15 latin 9, il faut installer le fichier iso_8859_15.7.gz :

```
cp standard/iso_8859_15.7.gz /usr/local/man/man7/
```

6.22. Correcteur orthographique ispell

Vous pouvez vous le procurer sur le site original d'Hydro Québec ou sur le mirroir du lip6

Un fichier LISEZMOI explique pas à pas et en Français l'installation.

6.23. Les claviers 105 touches

Aussi appelés « claviers Microsoft (tm) », ils ont trois touches de plus que les claviers normaux.

Comment s'en servir ?

En mode console, éditer *french.map* et décommenter les lignes en parlant !

De même pour *french* (pour Xwindow).

6.24. PGP

Attention, dans certains pays totalitaires, l'utilisation de logiciels de cryptographie est considérée comme usage d'armes de guerre, ce qui est punissable par la loi !

Sautez donc ce paragraphe si votre législation locale ne permet pas d'utiliser des logiciels de cryptographie.

Premièrement, il faut se procurer pgp−international (à cause de problèmes de patentes et de législation sur l'export d'armes de guerre, il existe pgp−us et pgp−international, ce dernire étant doté d'un algorithme encore plus puissant) sur le site norvégien de PGP

Ensuite, mettre dans votre répertoire maison, dans un sous répertoire .pgp les fichiers :

```
config.txt
language.idx
pgpdoc1.txt
pgpdoc2.txt
language.txt
fr.hlp
en.hlp
pgp.hlp
```

Enfin, éditer le fichier *.pgp/config.txt* pour y rajouter :

```
Language = fr
CharSet = latin1
```

Lisez ensuite *.pgp/fr.hlp* pour apprendre à vous servir de pgp !

6.25. Impression en mode ascii

Pour l'impression de texte simple, beaucoup d'imprimantes attendent des lettres en codage IBM (cp850) comme sous MS DOS.

Si le fichier est codé en latin1, on doit utiliser « recode latin1:ibmpc » (à placer dans les filtres lpr de /etc/ par exemple), sous peine de voir imprimer des « alpha » à la place de « à », et autres choses du même genre.

Il existe d'autres utilisation au programme *recode* : « ibmpc:lat1 » ou « lat1:ibmpc » (de ou vers MS−DOS) ou « applemac » pour Macintosh (tm).

6.26. Unicode/latin/cp... je n'ai pas compris ce dernier paragraphe !

Voici donc plus d'explications grâce à Pablo Sartxaga (srtxg à chanae.alphanet.ch) dans un message explicant la différence Unicode/latin/cp :

```
 DB>   quand on lance « setfont » sans argument
 DB>   il semble qu'on obtienne le même effet que setfont −u none

Normal et logique.

 DB>   --> en un mot la table de conversion « Unicode » est désactivée
 DB>       ce qui transforme le « e accent aigü (dec 130) » en « théta (dec 233) »

Non. Il ne transforme rien du tout, il affiche tout simplement sans
*aucune* transformation au contraire, à toi de t'assurer que la police
que tu charges possèdes les glyphes à la bonne place par rapport au codage
de caractères que tu utilises (iso−8859−1).

Autrement dit si ta police n'est pas iso−8859−1 tu dois dire à setfont de
faire des mappings; en particulier s ic'est une police qui utilise l'encodage
DOS cp437 tu fais « setfont −u cp437 nomdelapolice ». Moi par exemple j'aime
beaucoup la police « t.fnt » de la Slackware, et j'ai donc quelque part
dans /etc/rc.d/* une ligne « setfont −u cp437 t.fnt »

Ou alors tu modifies la police (il y a un logiciel qui fait cela avec
interface svgalib, et un autre nommé « chedit » qui permet de créer des
images an ascii−art, un fichier par caractère (256 fichiers donc), il te
suffit alors de jongler avec les fichiers pour les changer de place et
régénerer la police).  Note cependant que le codage cp437 n'inclut pas
tous les caractères de iso−8859−1, notamment manquent quasi toutes les
majuscules accentuées, et les lettres ãõ nécessaires en portugais. Tu
peux toujours modifier la police pour créer ces caractères manquants.

 DB>   sur ma Slackware 3.2, à aucun moment setfont n'apparaît dans les scripts
 DB>   de démarrage ni dans les ~/profiles et autres ~/.bashrc

Donc c'est la police dans la ROM de la carte video qui est utilisée.

 DB>   et pourtant cette
 DB>   conversion est effectuée correctement puisque j'ai les accents à l'écran.

Je suppose que le kernel présuppose (à juste titre) que la police en ROM vidéo
est au codage cp437.

 DB>        une fois que je lance « setfont −u none » je perds donc les accents.

Normal, car dès lors que tu charges une police tu est supposé savoir ce que
tu veux (au contraire de la police en ROM où tu n'as pas le choix).

Autrefois (du temps des 1.2.* et avant) le défaut de setfont était cp437
(et il n'utilisait pas encore des tables Unicode, mais on pouvait
changer par \e(B et \e(K entre codage cp437 et iso (on peut toujours
d'ailleurs)), mais maintenant le défaut est « straight to font », pas de
conversion.

 DB>            j'en conclue que Linux utilise la police résidente par défaut de
 DB>            la carte EGA/VGA

 DB>        [Q] --> comment retrouver cette police ?

Il y a un programme nommé « restoretextmode » qui est assez populaire
auprès des possesseurs de certaines cartes graphiques chez qui le
passage X11 -> console bousille les polices; il permet de faire un dump
sur un fichier; il faudrait donc que tu le lances avant de modifier la
police avec setfont. Le paquetage s'appelle quelque chose comme
SVGATextMode ou quelque chose d'approchant.

 DB>     [Q] --> quelle table appeler pour la remapper et ravoir les accents ?

setfont −u cp437 policedos
```

```
Ou sinon, si tu ne veux pas lancer setfont et tu veux donc garder la
police de la carte graphique; essaye un « echo -e \033(B » (l'ennuyant
c'est que tu dois le faire dans chaque console)

 DB>                  (en général setfont -u def.uni (par exemple) renvoie une
 DB>                   erreur : PIO_UNIMAPCLR: Invalid argument et je reste
 DB>                   en caracteres « graphiques »)

Ah ? Chez moi ça marche; quels kernel et kbd as-tu ?

Sinon je te conseille « cp437 » plutôt que « def », ils font _presque_ la même
chose, mais « cp437 » est visuellement plus agréable car il remplace les
caractères non présents dans la police par les lettres non accentuées si c'est
des lettres, C pour © et R pour ® et par un carré blanc pour le reste, ce qui
permet de garder le formatage et la lecture agréables; « def » lui ne mets
totu simplement rien pour les caractères manquants, si bien que tu te
trouves avec des trous, ce qui est l'horreur si tu édites un texte par
exemple avec vi, puisque le curseur est affiché plusieurs caractères
à gauche à cause des caractères de largeur nulle :)

Essaye aussi la police « t.fnt », elle est assez jolie, toute en rondeurs, je la
trouve réposante, et quand je dois utiliser un PC avec les polices style
courier par défaut je trouve ça très agressif je me demande comment ils
peuvent travailler avec :)
```

Lire aussi les articles de news : « FAQ − les accents français et Usenet» de F. Yergeau dans les groupes fr.usenet.reponses et fr.usenet.8bits » ISO 8859−1 National Character Set FAQ» de mike (à) vlsivie.tuwien.ac.at dans comp.answers .

6.27. ncurses

Rappelons que ncurses est un librairie C qui permet de manipuler facilement l'affichage en mode console texte : fenêtrage, lignes de saisie, édition de formulaires, couleurs, caractères « ALT » ascii semi−graphique ibm, comme sous dos...

6.27.1. Introduction

La version standard de curses utilise le huitième bit pour la vidéo inversée (voir le flag _STANDOUT défini dans /usr/include/curses.h). Cependant, ncurses semble fonctionner en 8−bits et affiche le iso−latin−8859−1 correctement.

6.27.2. Les accents

Remarque: les distributions de Linux sont livrées en général avec la version 1.9.9e de ncurses, or il se trouve que la librairie form de cette version est BUGGÉE (problèmes de rafraîchissement des fenêtres).

Ne pas utiliser la 1.9.9e donc si on veut travailler avec « form » : ce bug est d'ailleurs clairement énoncé sur le site de ncurses, et je m'étonne que ce soit justement cette version qui soit encore livrée sur la plupart des distributions (RedHat 4.2, 5.0, 5.1, Slackware 3.2, Debian 1.9) en tout cas.

Symptôme : la fonction wgetch() de ncurses renvoie les codes suivants :

- je tape 'é' et wgetch retourne « meta−i » code 233 (decimal)

- " " 'è' " " " « meta−h » code 234 (decimal)
- etc

C'est normal avec les polices iso−8859−1 !

Il faut installer les « locales » ou mettre à jour sa version de ncurses.

6.28. Perl

Si tout d'un coup après une mise à jour il raconte :

```
(guylhem@victis:guylhem)$ perl
perl: warning: Setting locale failed for the categories:
        LC_CTYPE LC_COLLATE
perl: warning: Please check that your locale settings:
        LC_ALL = "fr_FR",
        LC_CTYPE = "ISO-8859-1",
        LC_COLLATE = (unset),
        LANG = "fr"
    are supported and installed on your system.
perl: warning: Falling back to the "C" locale.
(guylhem@victis:guylhem)$
```

En libc6 le format des fichiers binaires de définition des locales a (encore) changé : il faut les régénérer à partir des sources sources WG15collection.tar.gz par exemple et des programmes « locale » et « localedef » qui sont fournis dans les sources de la libc.

Sinon on peut également se passer des locales :

```
     unset LANG
     unset LC_CTYPE
     unset LC_COLLATE
     export LC_ALL=fr_FR.ISO-8859-1
```

6.29. Installer les locales

Les « locales » sont des fichiers qui vont modifier le comportement de certains programmes pour qu'ils s'adaptent aux « spécificités culturelles du pays ».

Cela sert à résoudre les erreurs rapportées par perl ou une mauvaise gestion des accents par ncurses.

- récupérer pour la libc5 la collection de locales POSIX
- ou pour la libc6, elle est dans l'ajout « glibc−localedata »
- copier dans le répertoire /usr/share/locale les fichiers fr_FR et en_DK (qui est inclus dans fr_FR)

- créer le répertoire /usr/share/i18n/charmap et copier le fichier ISO_8859−1:1987
- faire man localedef et lancer la commande :

```
          localedef −f ISO_8859−1:1987 −i fr_FR fr
```

  Ceci créera l'entrée fr dans /usr/share/locale soit :

```
        /usr/share/locale/fr
```

  Ce répertoire devrait maintenant contenir les fichiers :

```
         LC_COLLATE
         LC_CTYPE
         LC_MESSAGES
         LC_MONETARY
         LC_NUMERIC
         LC_TIME
```

- lire le fichier « locale.fr » que l'on peut trouver sur le serveur ftp.lip6.fr (chercher sous doc, linux...) et dont la traduction française a été faite par Éric DUMAS (hélas ce doc ne traite principalement que de LC_MESSAGES. À quand la suite Éric ? ;−))
- compléter éventuellement l'initialisation des variables du shell comme décrit dans la section sur bash ou tcsh
- dans un programme, utilisez la fonction setlocale() (man setlocale.3) pour fixer les paramètres locaux CAR le noyau linux initialise toujours cette fonction avec l'argument une locale « C » (POSIX) donc sans accents !

```
          setlocale(LC_CTYPE,"fr_FR");
```

- Ça y est ! Les accents sont disponibles dans ncurses.

6.29.1. Midnight Commander (mc)

En changeant de police, il peut arriver que les lignes de cadre soient remplacées par divers caractères.

2 possibilitées :

- La police chargée ne dispose pas de caractères dits « semi graphiques », qui permettent de dessiner des cadres
- Il y a discordance entre la police et sa table de mappage

Pour le premier cas, la seule solution est de changer de police, mais dans le second cas, il suffit en général de charger la table avec loadunimap par exemple :

```
          loadunimap lat1u.uni
          loadunimap lat5u.uni
          loadunimap lat9u.uni
```

6.30. Kernel

Pour mettre un support pour le clavier français directement dans le kernel, ce qui est pratique pour les systèmes embarqués ou ne démarrant que sur disquettes, afin de gagner le maximum de place :

```
/usr/bin/loadkeys loadkeys --mktable votre-table-de-clavier.map > /usr/src/linux/drivers/char/defk
```

6.31. Lilo

Depuis la version 20, on peut spécifier un clavier (français, par exemple) au démarrage de LILO.

Bien sur, cela ne fait qu'échanger quelques touches (a/q,m/:,... pour un clavier fr) mais cela est assez utile lorsque l'on veut taper « win ».

Je vous conseille toutefois de ne pas faire de label utilisant des touches qui diffèrent entre l'AZERTY et le QWERTY, ne serait−ce que si vous devez avoir un jour un autre utilisateur sur votre ordinateur...

Il est plus simple de faire des labels courts et d'utiliser des programmes comme GAG ou GRUB, remplaçant fort avantageusement les lignes de lilo par un magnifique menu graphique dans le cas de GAG.

6.32. Groff (man)

Sur certaines distributions, le programme man n'est pas configuré pour afficher les accents.

Si vous rencontrez ce problème, éditez son fichier de configuration (/etc/man.config sur les distributions actuelles) et localisez une ligne ressemblant à cela :

```
NROFF           /usr/bin/groff −Tascii −mandoc
```

Remplacez−là par :

```
NROFF           /usr/bin/groff −Tlatin1 −mandoc
```

Les vieilles versions de man utilisent aussi col, et le point suivant s'applique aussi.

6.33. Divers

Une belle discussion sur le thème de l'ISO−8859−1 et sur « comment manipuler les caractères 8−bits » est disponible

7. Réglage du clavier pour les applications X

7.1. Les xterminaux (xterm, nxterm, rxvt...)

Les faire fonctionner avec toutes les touches standard du clavier 102 touches relève du miracle, quant aux accents, seule une recompilation vous permettra de les utiliser !

Pour xterm, essayez de mettre dans votre *.Xdefaults* :

```
XTerm*eightBitInput: true
XTerm*eightBitOutput: true
*customization: −color
XTerm*VT100*Translations: #override\n\
XTerm*pointerColor: red
None<Key>Begin: string(0x1b) string("[1~")\n\
None<Key>End: string(0x1b) string("[4~")\n\
None<Key>Prior: scroll−back(1,halfpage)\n\
None<Key>Next: scroll−forw(1,halfpage)\n\
Shift<Key>Prior: scroll−back(1,page)\n\
Shift<Key>Next: scroll−forw(1,page)
XTerm*fontMenu*fontdefault*Label: Default
XTerm*VT100*font: 9x15
XTerm*fontMenu*font1*Label: Illisible
XTerm*VT100*font1: nil2
XTerm*fontMenu*font2*Label: Minuscule
XTerm*VT100*font2: 5x7
XTerm*fontMenu*font3*Label: Petit
XTerm*VT100*font3: 6x10
XTerm*fontMenu*font4*Label: Normal
XTerm*VT100*font4: fixed
XTerm*fontMenu*font5*Label: Moyen
XTerm*VT100*font5: 7x13
XTerm*fontMenu*font6*Label: Tres grand
XTerm*VT100*font6: 10x20
```

Dans *.inputrc* :

```
# pour rxvt
"\e[7Ü":beginning−of−line
"\e[8Ü":end−of−line
# pour xterm
"\e[H": beginning−of−line
"\e[F": end−of−line
```

Mais ces deux solutions ne vous garantissent pas de résoudre tous les problèmes de rxvt et de xterm ...

Pour les résoudre sans créer de fichiers dans */usr/X11R6/lib/X11/app−defaults*, je tiens sinon à votre disposition un xiterm, un terminal X supportant *TOUTES* les touches des claviers standard, avec Home, End, Page_Up (...) et même le pavé numérique, fonctionnant comme en mode console, i.e. Shift−Page_Up permettant de faire défiler une page, voire mieux : le pavé numérique reconnaissant enfin la différence entre Verr. Num allumé et Verr. Num éteint.

X international terminal (xiterm pour les intimes) est bien sûr sous GPL et mis à jour pour suivre parallèlement l'évolution de rxvt.

Sur par exemple mon site vous trouverez donc xiterm, qui remplace votre xterm classique en y apportant la couleurs, les accents et les touches étendues.

Pour l'installer, compilez−le, devenez root et tapez :

```
rm /usr/X11R6/bin/xterm
rm /usr/X11R6/lib/X11/app-defaults/xterm
rm /usr/X11R6/lib/X11/app-defaults/rxvt
rm /usr/X11R6/lib/X11/app-defaults/Xterm
rm /usr/X11R6/lib/X11/app-defaults/XTerm
rm /usr/X11R6/lib/X11/app-defaults/Rxvt
rm /usr/X11R6/lib/X11/app-defaults/RXvt
cp xiterm /usr/local/bin

ln -sf /usr/local/bin/xiterm /usr/X11R6/bin/xterm
ln -sf /usr/local/bin/xiterm /usr/X11R6/bin/rxvt
```

7.2. Les applications Motif

Là, je n'ai malheureusement aucun remède miracle !

Il faudrait modifier le code source, code indiqué plus haut, pour remplacer XLookupString par XmLookupString.

Utilisez les possibilités du fichier french au lieu des touches mortes (AltGr (voyelle) : voyelle accent circonflexe...) !

7.3. Le manuel

Pablo Saratxaga tient à votre disposition un xman modifié pour qu'il supporte la variable « LANG » et puisse afficher les pages en d'autres langues qu'anglais.

Il est disponible sur le lip6 en sources ou en binaires

7.4. NumLock

Pour l'activer sous X, mettre en commentaire ServerNumLock dans la section keyboard de XF86Config, puis rajouter au dessous Xleds 2, ce qui allumera la led.

Attention, ce conseil ne s'applique que lorsque l'on n'utilise pas xkb, i.e. qu'on met XkbDisable dans XF86Config...

7.5. XDM

/etc/X11/xdm/Xresources sert à ne rencontrer aucun problème avec les touches « spéciales » (Home, End ...) sous X.

```
xlogin*login.translations: #override\
      Ctrl<Key>R: abort-display() \n\
      <Key>BackSpace: delete-previous-character() \n\
      <Key>Home: move-to-begining() \n\
      <Key>Delete: delete-character() \n\
      <Key>End: move-to-end() \n\
      <Key>Left: move-backward-character() \n\
      <Key>Return: set-session-argument() finish-field() \n\
      <Key>Right: move-forward-character()
```

8. Remerciements

Ce document ne serait pas ce qu'il est sans l'aide de Nat Makarévitch (nat à nataa.fr.eu.org) qui l'a relu et m'a aidé à le modifier de nombreuses fois.

Un excellent document plus générique, traitant de l'installation par ex, est le Guide du Rootard, disponible sur freenix ou le lip6

Remercions enfin par ordre alphabétique ces contributeurs :

```
Jean−Michel Antoine
Michel Billaud
Chmouel Boudjnah
Stephane Bortzmeyer
Denis Braussen
Rémy Card
Julien Cassaigne
Xavier Cazin
Laurent Chemla
René Cougnenc
Bruno Cornec
Yann Dirson
Éric Dubreuil
Éric Dumas
Arnaud Gomes−do−Vale
Pierre Ficheux
Laurent Frigault
Hugolino
Antoine Leca
Frederic Lepied
Jeannot Langlois
Stephane Marzloff
Marc Merlin
Thomas Parmelan
Frédéric Petit
Thomas Quinot
David Robert
Olivier Robert
Benjamin Ryzman
Pablo Saratxaga
Julien Simon
Loïc Tortay
J.M. Vansteene
```

Surtout, si je vous oublie, signalez−le moi ! De même si vous ne voulez plus figurer dans cette liste...

Envoyez moi vos suggestions et modifications (guylhem à metalab.unc.edu), je me ferai un plaisir de les intégrer dans les prochaines versions !

Signalez−moi aussi les erreurs : cet HOWTO est en perpétuel changement, avec des nouveautés incluses à chaque version ; des erreurs peuvent facilement s'y glisser !