Pages Web mutlilingues avec UTF-8

Exemples turcs et indiens

1. Mécanismes fondamentaux (l'informatique sous-jacente)
2. Le turc
3. La devanagari

A. Mécanismes

1. Philosophie de la mécanique

Rappel : il n'y a pas de caractères dans une machine, rien que des octets !

Glose : ce qu'on trouve dans les mots de la mémoire centrale ou sur les pistes d'un disque, ce qui circule sur le réseau, ce ne sont que des octets...
Ce que nous prenons pour des caractères est produit par une interprétation de ces octets.

Les octets demeurent, les interprétations varient...
Quand on transmet des octets d'une machine à l'autre,
l'interprétation par le récepteur est en général différente de celle de l'émetteur.

Entrer & sortir :
les machines modernes avec écran et clavier nous laissent croire que lire et écrire ne font qu'un
c'est faux !
L'écriture (fabrication d'un fichier) et la lecture (interprétation) sont deux processus *dissymétriques*.
Cette dissymétrie est masquée par la puissance et la bonne adéquation des outils (clavier - écran - logiciels)
Mais la puissance est relative et l'adéquation dépend du but poursuivi !

Il n'en a pas toujours été ainsi (à l'origine, cartes perforées et imprimantes...)
et actuellement l'exigence nouvelle de multilinguisme rend les outils inadéquats.
Nous sommes donc ramenés (provisoirement) à une situation analogue à celle des débuts de l'informatique.

2. Moyens d'observation

Comment observer la réalité des processus d'écriture et de lecture ?

Principe : **On ne voit jamais directement les octets,**
ils sont toujours interprétés par le logiciel de visualisation.

> Cherchons donc l'interprète le plus fruste, celui qui "en fera le moins"
> et qui nous révélera le fichier "tel qu'en lui-même".

Sous Unix (Mac OS X.3)

- la commande `cat` interprète complètement le format UTF-8 !
  elle nous masque donc la réalité des octets.
- en revanche la commande `more` n'interprète que l'ASCII 7bits
  et reproduit fidèlement tous les autres octets.

On peut donc utiliser `more` comme instrument d'observation de la matérialité des octets.

Observations préliminaires

La notation hexadécimale

On n'écrit jamais les chaînes de bits, elles sont illisibles.
En hexadécimal chaque chiffre représente 4 bits, un octet est donc représenté par deux chiffres.
Exemples : `09` = `0000 1001`, `2F` = `0010 1111`

Le code ASCII 7 bits - 128 caractères : **un caractère = un octet** (avec le premier bit à 0)

Les extensions iso-8859-1 (iso-latin-1), Mac Roman et autres - 256 caractères :

- *idem*, **un caractère = un octet** (complet)
- mais différences incompatibles entre plateformes et entre langues

Voici le même texte en deux codages Mac Roman et iso-8859-1 (Windows) :

- `Ça, c'est mon frère René et ça c'est ma sœur Iñès.`
- `<82>a, c'est mon fr<8F>re Ren<8E> et <8D>a c'est ma s<CF>ur I<96><8F>s.`
- `<C7>a, c'est mon fr<E8>re Ren<E9> et <E7>a c'est ma s<9C>ur I<F1><E8>s.`

Enfin Unicode vint...

et **un caractère n'est plus un octet !!!**

Bien distinguer

- **la désignation** du caractère (nommer le caractère)
  - par son n° (en décimal ou en hexa)
    - en HTML/XML : `&#lenumérodec;` ou `&#xlenumérohex;`
    - en Java : `\ulenumérohex`
  - par son nom officiel (en Perl)
  - Exemple : Unicode n° 304 = `x0130 LATIN CAPITAL LETTER I WITH DOT ABOVE`
    peut être désigné en HTML/XML par `&#304;` ou par `&#x0130;` et en Java par `\u0130`

- **sa réalisation,**
  laquelle prend deux aspects (dissymétriques):
  - codage en octets (UTF-8)
    Exemple : `LATIN CAPITAL LETTER I WITH DOT ABOVE` ==> sur 2 octets `C4B0`

  - affichage via une police (présente sur la machine !)
    Exemple : `LATIN CAPITAL LETTER I WITH DOT ABOVE` ==> ***İ***

Exemple : le texte ci-dessus en UTF-8 :

```
<C3><87>a, c'est mon fr<C3><A8>re Ren<C3><A9> et <C3><A7>a c'est ma
s<C5><93>ur
I<C3><B1><C3><A8>s.
```

Mise en œuvre :

- Côté entrée :
  1. Écriture dans un outil d'édition par un moyen quelconque (clavier, palette)
  2. Sauvegarde dans un fichier **en spécifiant le codage UTF-8**
     par exemple, dans une fenêtre de dialogue.

- Côté sortie : avertir le logiciel de traitement qu'il doit "lire" en UTF-8
  - à l'ouverture (*préférences* de l'outil)
  - par un message spécifique avec un type MIME comme `text/plain` ou `text/xml`
    assorti de la mention `charset="UTF-8".`
    - Pour un courrier envoyé par SMTP, on dira dans un en-tête (*header*) :
      `Content-Type: text/xml; charset="UTF-8"`
    - Dans un fichier HTML, on va ajouter comme premier élément de la partie `<head>` un élément ainsi conçu:
      `<meta http-equiv="Content-Type" content="text/html; charset="UTF-8" />`
      en observant que tous les caractères précédents, à savoir
      `<DOCTYPE....> <html xmlns:...><head>`
      sont de l'ASCII 7 bits, donc déchiffrables sans ambiguïté.

B. Un exemple de bilinguisme franco-turc : problème & solution

1. Problème à traiter

Le turc s'écrit avec un alphabet latin étendu dont 4 caractères sont étrangers au français :

1. le "i sans point" minuscule : **ı** (voyelle d'arrière)
   Unicode n° 305 = `x0131 LATIN SMALL LETTER DOTLESS I`

2. le "i majuscule avec point" **İ** (pour le distinguer du i sans point majuscule)
   Unicode n° 304 = `x0130 LATIN CAPITAL LETTER I WITH DOT ABOVE`

3. le g "mou" (variante phonologique du g entre voyelles)
   minuscule **ğ** Unicode n° 287 = `x011F LATIN SMALL LETTER G WITH BREVE`

   majuscule **Ğ** Unicode n° 286 = `x011E LATIN CAPITAL LETTER G WITH BREVE`

4. le s cédille (note la chuintante sourde, fr. "ch")
   minuscule **ş** Unicode n° 351 = `x015F LATIN SMALL LETTER S WITH CEDILLA`

   majuscule **Ş** Unicode n° 350 = `x015E LATIN CAPITAL LETTER S WITH CEDILLA`

J'ai trouvé il y a plusieurs années une police de caractères dite `international` permettant d'obtenir ces caractères sous Word par le procédé suivant :

1. Le "i" sans point minuscule : taper ^ puis espace : *garı*

2. Le "i" majuscule avec point : taper opt-shift-x après le "I" : *İzmir*

3. Le "g yumuşak" : taper opt-u après le "g" : *oğlan*
   pour la majuscule, taper opt-$ après le "G" : *OĞLAN*

4. Le "c" et le "s" cédille : taper opt-< après la lettre : *çicek*, *Çicek*, *paşa*, *PAŞA*

5. Le tréma : en minuscule, opt-shift-t après la lettre : *Ben yürürüm yana yana*
   en majuscule, opt-shift-r après la lettre : *Ödemiş, Turgut Özal*

J'ai donc dans mes archives une collection de documents réalisés avec cette technique, contenant des textes turcs annotés en français.
Les lettres turques y sont "codées" comme expliqué ci-dessus, avec la police `international`
et le logiciel de visualisation (en l'occurrence Word) interprète ce codage avec cette police de manière à restituer les glyphes attendus.

Le problème est de transformer ces fichiers en pages Web bilingues, uniformément codées en UTF-8.
Exemple : une anecdote de Nasreddin Hodja

2. **Solution**

Principe : (en Java) lire le fichier caractère par caractère et
laisser intacts les caractères français (sous réserve qu'ils aient été lus correctement)
décoder les constructions turques ci-dessus en les traduisant en Unicode.
Exemple : `OG€LAN` ==> *OĞLAN*

Réalisation : 3 étapes à partir du fichier Word (contenant les balises HTML de mise en forme)
1. Obtention du texte en Unicode UTF-16 (dialogue avec Word)
2. Recodage en Mac Roman (pour assurer la bonne lecture par Java sur Macintosh) via TextEdit
3. Traduction par Java.

Démonstration...

C. Mise en œuvre de la devanagari : traitement des ligatures

1. **Le problème**

L'écriture devanagari considère que la voyelle a (bref) est inhérente à chaque voyelle.
Elle ne l'écrit donc pas : क = *ka*

Les autres voyelles en revanche sont écrites : = *ku*, = *ko*, etc.

Mais comment faire si on ne veut pas de voyelle, pour avoir un groupe de 2 ou 3 consonnes ?
On a alors recours à une **ligature**, qui peut prendre diverses formes (glyphes) :
क + व = क्व , क + म = क्म , क + ष = क्ष , क + र = क्र

Mais **les caractères correspondants ne figurent pas dans le catalogue Unicode !**

Comment va-t-on faire ?

2. La solution

On écrira sous forme normalisée,
avec une lettre spéciale (le *virama*, n° 2381 = `x094D`) indiquant l'absence de voyelle

et **la réalisation de la ligature sera confiée au dispositif d'affichage.**

3. Démonstration...