Les caractères spéciaux en XML et HTML – la norme Unicode

Andrei Popescu-Belis
TIM / ETI, Université de Genève

Cours n°5 — annexes

Représentation informatique des systèmes d'écriture

- Problèmes informatiques
  - ordre de grandeur très différent entre systèmes alphabétiques (et syllabiques) d'une part, et idéographiques de l'autre
- Au début (premiers ordinateurs de série)
  - alphabet latin, *dans la version anglaise*
  - puis les versions de l'alphabet latin
- Plus récemment
  - solutions pour le système CJK (Chine, Japon et Corée)
    - plusieurs dizaines de milliers de caractères
- Question 1 : et les autres alphabets ?
- Question 2 : et la compatibilité ?

Encodage des caractères dans un ordinateur (1)

- « Information »
  - suite de signaux binaires (' **bi**nary digi**t** ' = bit)
  - traditionnellement symbolisés par 0 / 1
- Octet : série de huit bits
  - il existe 256 valeurs possibles pour un octet
    - 00000000, 00000001, 00000010, 00000011, 00000100, 00000101, ..., 11111111
  - chaque octet se traduit par un nombre décimal entre 0 et 255

Encodage des caractères dans un ordinateur (2)

- Affichage des caractères à l'écran
  - grâce à une interface graphique
  - traduction des codes-machine en lettres affichées (*glyphes*)
  - codes lus (en général) octet par octet
- Jeu de caractères
  - correspondance entre les octets et les caractères
  - par exemple, $01000001_b = 65_d$ = ' A '

Définitions importantes

- Jeu de caractères (*character set*)
  - correspondance théorique entre des caractères (définis conceptuellement) et des nombres
  - valeurs de nombres : 0–127 ou 0–255 ou 0–65535
  - simple convention sur papier
- (En)codage (*encoding*)
  - stockage concret dans la machine des nombres (codes des caractères) sous la forme d'une suite de bits 0/1
- Police (de caractères) (*font*)
  - un jeu de caractères plus la forme des glyphes

En d'autres mots…

- Jeu de caractères (*character set*)
  - correspondance théorique entre des caractères (définis conceptuellement) et des nombres
  - US-ASCII : 0-127
  - ISO-LATIN-1, ISO-8859-2 : 0-255
  - Unicode (ou ISO/IEC 10646) : 0-65535 (au moins)
- (En)codage (*encoding*) des documents textes/balises
  - stockage concret dans la machine des nombres (codes d'un jeu des caractères) sous la forme d'une suite de bits 0/1
  - pour Unicode (2 octets), UTF-8 (longueur variable)
    - ou bien « direct » avec UCS-2, appelé aussi UTF-16
  - pour les autres, encodage « direct » (octet par octet)

Historique des jeux de caractères (1)

- Le standard ASCII - sur « presque un octet »
- Années 1960-1970 et jusqu'à nos jours
  - utilisation des sept premiers bits d'un octet
  - de 0000000 à 1111111 (7 bits) = 2^7 = 128 possibilités
  - donc 128 symboles distincts possibles
- Utilisation
  - une trentaine d'entre eux = caractères de commande
  - le reste : alphabet anglais standard, chiffres, ponctuations

7

Historique des jeux de caractères (2)

- Les standards ISO-LATIN – sur un octet complet
- Utilisation des 256 caractères possibles (2^8)
  - 96 codes (de 160 à 255) affectés au codage des caractères latins accentués dans diverses langues d'Europe de l'Ouest ou Centrale ou de l'Est, ou autre…
- Les standards ISO-LATIN-1 (ISO-8859-1), ISO-LATIN-2, …, ISO-LATIN-15 = diffèrent par les caractères affectés aux codes de 160 à 255
- De plus : variantes introduites par les différents formats propriétaires (Windows, MacOS)

8

Exemple

- Times et Arial, toutes les deux dans le jeu ISO-latin-1
  - code 249 = " u minuscule grave "
  - style différent
    - Times: ù avec sérif (*empattement*) // Arial : ù sans sérif
- Bien distinguer
  - différence entre deux jeux de caractères
  - différence entre deux polices
  - NB :
    - une police est souvent liée à un jeu de caractères
    - les logiciels disposent souvent de la même police dans plusieurs jeux de caractères

    → une certaine confusion

9

Limitation des jeux ISO-LATIN

- On ne peut utiliser simultanément plus de 96 caractères alphabétiques différents (les codes de 160 à 255)
- Exemples de variantes
  - ISO-LATIN-1 ou ISO-8859-1 : français, espagnol, allemand, basque, etc. (mais aussi en Afrique: afrikaans, swahili)
  - ISO-LATIN-2 ou ISO-8859-2: langues à alphabets latins d'Europe Centrale et de l'Est
- Exemple de différences
  - ISO-LATIN-1
    - définition de 217: " latin capital letter u grave " = " Ù "
    - définition de 249: " latin small letter u grave " = " ù "
  - ISO-LATIN-2
    - définition de 217: " latin capital letter u ring " = " Ů "
    - définition de 249: " latin small letter u ring " = " ů "

10

Limitation des jeux sur un octet

- Source
  - un caractère = un octet
- Problème
  - caractères mal affichés d'un ordinateur à l'autre
- Solution
  1. Utiliser un jeu de caractères standard et le préciser
     - mais que faire s'il n'y a pas de standard ou s'il faut en utiliser plusieurs dans un même document ?
  2. Transmettre avec un document toutes les polices spéciales utilisées (donc implicitement le jeu de caractères)
     - difficile : grande variété, beaucoup de documents (ex.: emails)

11

Utilisation de deux octets

- Utiliser deux octets pour coder chaque caractère
  - jeux de caractères sur deux octets (pour commencer)
- Résultat
  - on dispose de 65'536 codes (entre 0 et 65'535)
- Contraintes
  - les documents prennent plus de place
  - comment être compatible avec les documents précédents?
- Il faut fixer un standard, pour éviter les problèmes d'incompatibilité

12

Unicode : 0 - 65’535

- Jeu de caractères standard sur deux octets
- On peut faire tenir dans cette gamme
  - la plupart des caractères alphabétiques existants (dans tous les alphabets)
  - de nombreux caractères symboliques (par exemple mathématiques)
  - des dizaines de milliers d’idéogrammes CJK
  - … et il reste encore des codes non affectés

13

Unicode : 0 - 65’535 (suite)

- Définition d’un standard
  - Le Consortium Unicode et l’Organisation Internationale de Normalisation (ISO)
  - Standard Unicode ou ISO/IEC 10646
  - **http://www.unicode.org/charts**
  - application « Table des caractères » sous Windows
- Extension à quatre octets : d’autres alphabets prennent place
  - Unicode 5.0 → environ 100’000 caractères définis

14

Codage d’Unicode : UTF-8

- Unicode
  - correspondance « sur papier » entre les caractères et les codes
  - comment le représenter en machine ?
    - idée : simplement en utilisant les deux octets à chaque fois (UCS-2)
    - autre idée : à l’aide d’un « code » (UTF-8, etc.)
- UTF-8 : Unicode Transformation Format (8 bits minimum)
  - stockage concret des codes Unicode en machine (dans un fichier)
  - avantages
    - compatible avec ASCII (zones 0 – 127)
    - les caractères ASCII (‘0’ + 7bits) restent les mêmes en Unicode
  - codage de longueur variable
    - les caractères fréquents prennent moins de place (8 bits pour les caractères ASCII), les caractères rares un peu plus… (24 bits pour les caractères CJK)
  - il existe également d’autres codages d’Unicode en machine

15

Problèmes d’encodage (1)

- Que dire des standards ISO-8859 ?
  - tous sur un octet
  - difficile de savoir quel est la version du ISO-LATIN d’un fichier texte ou d’un email sans entête
- Les standards ISO-LATIN-1 (ISO-8859-1), ISO-LATIN-2, …, ISO-LATIN-15 diffèrent seulement par les caractères affectés aux codes de 160 à 255
  - les caractères ASCII (0-127) sont préservés si on se trompe de version
  - les caractères accentués (160-255) peuvent être erronés si on se trompe de version

16

Problèmes d’encodage (2)

- Observation
  - texte en UTF-8 lu avec un outil adapté à UTF-8
    “ un plug-in s'installe grâce à un fichier exécutable ”
  - même texte lu avec un outil utilisant ISO-LATIN-1
    “ un plug-in s'installe grÃ¢ce Ã un fichier exÃ©cutable ”
- Explication
  - les caractères ASCII restent bien invariables
  - les caractères non-ASCII (ici ISO-LATIN-1) sont codés en UTF-8 chacun sur deux caractères bien différents

17

Comment deviner l’encodage réel d’un fichier texte/balises – ou d’un email ?

- Pas de méthode universelle
- Chercher tout de même une déclaration d’encodage
  - en général, Windows distingue 7-bit *vs.* 8-bit (1 octet) *vs.* 2 octets
- Si déclaration absente ou erronée
  - ouvrir le document dans un navigateur et changer à la main l’affichage jusqu’à ce que le texte s’affiche correctement
    - si c’est une langue inconnue… impossible !
  - ouvrir le document avec Notepad/Wordpad et faire ‘Enregistrer sous…’
    - regarder l’encodage proposé par défaut
  - ouvrir le document avec Word en demandant une conversion explicite depuis un document « texte codé » (comment ?)
    - tester l’affichage avec les différents encodages proposés

18

Exemple de rédaction manuelle

- Comment insérer un « epsilon » en HTML ou XML

1. chercher un jeu de caractères contenant « epsilon » = ε
2. Unicode, bien sûr !
3. déclarer l'encodage UTF-8 pour le document (par défaut en XML)
4. chercher le code Unicode de ε : 603_d ou $025B_h$
5. écrire ɛ ou bien ɛ
6. vérifier l'affichage: navigateur correctement configuré

- Ou bien on utilise un programme d'édition (traitement de texte) base sur Unicode qui le fait directement

19

Outils de conversion de l'encodage

- Utiles pour partager des fichiers existants, dans différents formats « transparents »
  - (pour les formats propriétaires, choisir les options d'exportation du programme d'édition)
- Utilitaires pratiques
  - « recode » : http://recode.progiciels-bpi.ca/
  - « native2ascii » dans le Java Development Kit
  - « RXP » le fait aussi (lire manuel, option –c)

20