Architecture et Système

Stefan Schwoon

Cours L3, 2016/17, ENS Cachan

Codage caractères : Terminologie

Pour connaître les différents standards de codage de caractères, il sera utile de distinguer les termes suivants :

jeu de caractères : l'ensemble (non-ordonné) des caractères (graphèmes) dont on dispose
On note $A \subseteq B$ si le jeu du standard A est un sous-ensemble de celui de B.

jeu de caractères codés : fonction qui affecte à chaque graphème un *code* numérique.
On note $A \prec B$ si $A \subseteq B$ et A affecte tous ses caractères aux mêmes codes que B.

Terminologie

Un chaîne de caractères sera donc représentée par une suite de codes.

Ces codes doivent être représentés en binaire. Selon les besoins, les mêmes codes peuvent être représentés différemment, on distingue donc deux couches de représentation :

codets : découpage du code en une ou plusieurs unités, généralement des valeurs entre 0 et 255

forme de codage : comment représenter un codet

Débuts historiques

Terminal : à l'origine, machine "bête" conçue pour laisser un utilisateur accéder à une machine partagé :

Affichage des caractères saisis par l'utilisateur + transfert vers l'ordinateur

Affichage des données venues de l'ordinateur

À l'époque : un caractère équivaut un octet, chaque caractère à son code

Caractères imprimables et de contrôle) (retour chariot CR, saut de ligne LF, appel BEL, retour arrière BS, . . .)

Les standards les plus importants

ISO-646 in 1963/1972, versoin américaine : ASCII

codes à 7 bits, caractères de contrôle avec codes < 32

Codes à 8 bits :

ISO-8859 (à partir de 1985) : diverse variants, le plus important étant ISO-8859-1 (= Latin-1), avec encore quelques positions inutilisées

Windows-1252 (arrivée avec MS Windows) : extension de Latin-1

$\text{ASCII} \prec \text{Latin-1} \prec \text{Windows-1252}$

Unicode/ISO 10646

(à partir de 1991) effort pour définir un codage *universelle*, c'est à dire pour pouvoir représenter tous les langages du monde.

version actuelle connaît 1 million de caractères différents

Latin-1 $\prec$ Unicode et Windows-1252 $\subseteq$ Unicode

Usage actuel :

plus de 80% des pages web modiales en Unicode, 10% en Latin-1

Découpage en codets

ASCII/Latin-1/Windows : facile, un caractère égale un octet

Unicode : plusieurs standards différents

- Coder chaque caractère avec 4 octets = 32 bits (UTF-32)
- Coder les caractères ‘bas’ avec 16 bits (UTF-16)
- Codage vers des codets ‘imprimables’ en ASCII (UTF-7)
- Le plus important : UTF-8

Découpage UTF-8

Un code représenté par un ou plusieurs *codets*.

Pour les caractères ASCII: un seul codets entre 0..127.

Pour les autres : 2 à 4 codets entre 127..255. Exemple :

> Les codes entre 128 (= 0x80) et 2047 (= 0x7FF) prennent deux octets.
>
> Le code pour “é” égale 233, en binaire avec 11 positions: 00011101001
>
> Le premier octet est composé de 110 suivi par les cinq premiers bits du code, le deuxième de 10 suivi par les six autres bits.
>
> Ça donne 11000011 10101001 = 0xC3 0xA9.

Parenthèse : gestion des touches

Qu'est-ce qui se passe quand l'utilisateur tape un touche ?

Le clavier signale une interruption auprès du processeur central.
(une interruption pour déprimer la clé, une autre pour la relâcher)

Pour chaque type d'interruption, le processeur connaît une adresse mémoire à appeler pour gérer l'interruption. Ce gestionnaire fait partie du système.

Lorsque le processeur est prêt pour traiter l'interruption, il transfert donc le contrôle au gestionnaire qui obtient le code de touche.

Les codes de touche sont spécifiques au clavier. P.ex. un clavier pourrait envoyer 11 pour la touche 'espace', un autre 25.

...

Sous X11, le gestionnaire fait appel a un pilote qui traduit le code de touche vers un nom de touche (Shift, Espace, A, . . .).

Le gestionnaire consulte la configuration de l'utilisateur pour traduire le nom de touche et les *modificateurs* actuels (Shift, Ctrl, Alt) vers un symbole (p.ex, A, a-accent-aigu, . . .).

Finalement, le gestionnaire génère un évènement auprès de l'application du premier plan qui contient le nom de touche et le symbole (`xev` laisse observer ces évènements).

Si l'application est un terminal, elle traduit le symbole selon son codage en vigueur.

Détails sur le terminal

Normalement, le terminal est en mode echo. Dans ce mode, il affiche immédiatement tout symbole reçu par le système. L'echo peut être désactivé, p.ex. pour les mots de passe . . .

En mode canonique, les codages des caractères sont envoyés au processus du premier plan (dans le terminal) lorsque l'utilisateur saisit "nouvelle ligne". En mode non-canonique, ils sont envoyés toute de suite.

En plus, le terminal reçoit des séquences d'octets de la part des processus qui lui sont attachés. Ces séquences seront décodées vers des caractères qui seront affichés sur l'écran.

Séquences de contrôle dans le terminal

Certains caractères non-imprimables ont une interprétation spéciale dans le terminal :

des contrôles simples (nouvelle ligne etc)

des séquences commençant par ‘escape’ (27), p.ex.:
ESC[31m : afficher texte en rouge
ESC[42m : arrière-plan en vert
ESC[0m : arrière-plan en vert
ESC[2J : effacer l’écran
ESC[10;20H : mettre le cursor sur position (10,20)

Le caractère ESC correspond à `\e` en C. Dans le shell, `echo -e "\e..."`

Forme de codage

Les codets sont typiqment sous la forme d'un valeur 0..255.

Typiquement donc, on représente un codet dans sa forme binaire dans un octet.

Mais parfois, on souhaite une représentation "imprimable" :

> P.ex., les protocoles Internet (mail, web) ne permettent que des caractères imprimables dans les entêtes, dans les adresses, ...
>
> Les formes imprimables résistent mieux aux erreurs de traductions car tout codage habituel est compatible avec ASCII.

Exemples de forme de codage non-trivial

Codage ‘pourcent’ dans les adresses web, p.ex. :

`https://fr.wikipedia.org/wiki/Caract%C3%A8re`

Codage ‘quoted-printable’, p.ex. dans les entêtes mail :

`Subject:  =?utf-8?Q?T=C3=A9st?=`

(donne ‘Tést’ comme sujet du mail)

Codage ‘base64’, pour le même contenu :

`Subject:  =?utf-8?B?VMOpc3Q=?=`

Dans base64, on se donne 64 symboles (A-Za-z0-9+/) qui code des séquences de 6 bits ; quatre symboles font trois codets.

Impact sur le programmeur

Dès qu'un programme interprète la valeur des châines de caracteur (longueur, comparaison, formattage), il doit prendre en compte le codagen en vigueur.

P.ex., `strcmp` (en C) ne fait que comparer des séquences d'octets
– non adapté pour trier des noms dans les langages autre qu'anglais.

En Unix, il existe une convention pour la paramétrisation locale, appelé locale (mot anglais).

Dans le shell : variables d'environnement, commande `locale`

Dans C : `setlocale`

Exemple : Trier des chaînes de caractère en C

Avec `strcmp`, les mots commençant avec É apparaîtront après ceux avec Z.

D'abord, appeler `setlocale(LC_COLLATE,"fr_FR.UTF-8")` pour activer le bon triage pour français (ou `setlocale(LC_ALL,"")` pour utiliser les paramètres du shell).

Pour comparer, on utilise `strcoll` plutôt que `strcmp`.