Jeux de caractères codés

Histoire

Structure

Manipulation

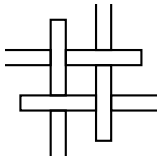
Jean-Marc Bourguet

Jeux de caractères codés

Histoire

Structure

Manipulation



Brouillon de la première édition 4 janvier 2009

Table des matières

Liste des tableaux

1 Introduction

Nous codons. Nous abstrayons. Nous codons les pensées que nous voulons communiquer sous forme de sons et nous appelons cela la parole. Nous abstrayons l'essentiel de la parole et nous appelons cela une langue. Nous codons les langues sous forme de dessins et nous appelons cela l'écriture. Nous abstrayons l'essentiel de ces dessins et nous appelons cela des lettres. Ces codes évoluent et acquièrent des conventions qui permettent d'exprimer certaines choses hors des limites strictes du langage qu'ils sont sensés coder. À mes intonations ma fille de dix-huit mois sais que je ne suis pas content et qu'elle a fait une bêtise. Je change de paragraphe et mon lecteur sait que je vais aborder une nouvelle idée.

Les codes dont il sera question ici sont ceux utilisés dans les ordinateurs et les moyens de communication numériques dans l'objectif de transmettre, conserver et manipuler l'information. L'objectif de dissimulation – dissimulation de l'information, la *cryptographie*, ou la dissimulation du fait même qu'il y a une information, la *stéganographie* – une autre application importante de codes, ne sera pas traité.

Dans les applications de communication, on peut coder les sons – c'est ce qui est fait pour le téléphone – ou analyser ceux-ci et coder la structure phonétique, en supposant ou non connue la langue. Ces codes ne seront pas non plus traités ici ; les codes que nous considérerons sont basés sur l'écriture.

2 Histoire

Les codes utilisés en transmission ont une longue histoire. Au deuxième siècle avant notre ère, le général et historien grec Polybe (Πολύβιος, v. 200 – v. 120 av. J.-C.) propose un système pour coder l'alphabet grec en utilisant deux groupes de cinq torches. Ce système a eu un certain succès : au XX^{e} siècle les prisonniers de guerre américains au Viêt Nam aurait utilisé une variante de ce système, adapté à l'alphabet latin [Wika].

Les marins de la Grèce antique utilisaient déjà des drapeaux pour communiquer entre navires d'une même flotte. Ce moyen de communication était relativement inflexible mais en 1738, Bertrand François Mahé de la Bourdonnais a conçu un code plus complexe. Dix drapeaux différents codaient les dix chiffres, ensuite trois drapeaux permettaient de coder 1000 messages en utilisant un dictionnaire pour faire la correspondance entre les nombres et les messages. Ce système ne fut pas directement employé, mais influança vraisemblablement Lord Richard Howe qui inclu un mécanisme similaire dans son « The Howe Code ». Par la suite, ce code fut complèté par Sir Home Popham dans son « Telegraphic Signals or Marine Vocabulary » qui finit de le perfectionner en incluant l'alphabet, permettant ainsi d'épeller ce qui ne se trouve pas dans le dictionnaire, à temps pour que ce soit utilisé lors de la bataille de Trafalgar en 1805 ([Mü]).

À la fin du dix-huitième siècle, les frères Chappe établirent un système de sémaphores composés de tours équipées de bras articulés. Comme le code de Popham, le code utilisé était un système à dictionnaire disposant d'un moyen d'épeller ce qui n'était pas dans le dictionnaire. Ce système fut utilisé par Napoléon pour gouverner son empire et commander ses armées, lui donnant un avantage sur ses ennemis [Wikb], [Kat].

En 1837 Samuel Morse brevette un télégraphe électrique. Le code qu'il propose alors est apparemment basé uniquement sur un dictionnaire. C'est son assistant, Alfred Vail,

A	·—	M	——	Y	—·——
B	—···	N	—·	Z	——··
C	—·—·	O	———	0	—————
D	—··	P	·——·	1	·————
E	·	Q	——·—	2	··———
F	··—·	R	·—·	3	···——
G	——·	S	···	4	····—
H	····	T	—	5	·····
I	··	U	··—	6	—····
J	·———	V	···—	7	——···
K	—·—	W	·——	8	———··
L	·—··	X	—··—	9	————·

TABLE 1: Le code Morse

qui aurait travaillé sur le code et proposé ce qu'on appelle le code Morse (voir table 1). Ce code est basé sur cinq symboles (deux durées de courant sur la ligne – représentées par « · » et « — » dans la table – et trois durées d'absence de courant – séparant les points et les traits, les lettres et les mots). Dans ce code tous les caractères codés ne sont pas composés du même nombre de symboles, ce qui peut être un avantage dans une application de transmission – après tout, c'est un principe de compression bien connu que de coder avec moins de symboles ce qui est fréquent ; ce n'est pas un hasard si le E, lettre la plus fréquente en anglais comme en français, est codée par « · » – mais est plutôt une nuisance quand il s'agit de traiter automatiquement ces informations.

En 1870 et toujours pour les transmissions télégraphiques, Émile Baudot invente un code utilisant deux symboles (présence ou absence de courant), c'est donc un code binaire, comme tout ceux que nous rencontrerons par la suite. Les caractères étaient codés en utilisant cinq *bits* (*moment* dans la terminologie de l'époque). Longueur fixe, car ce code était utilisé avec un clavier à cinq touches manipulé par un opérateur du coté de l'émetteur. La réception était totalement automatique, le message était imprimé par le système. Le code (voir table 2 [1]) a été conçu de manière à limiter la fatigue de l'opérateur. Pour permettre plus de 32 caractères (ce qui ne permet même pas de coder 26 lettres plus 10 chiffres), ce code a deux modes : certains motifs sont utilisés pour coder deux caractères différents, le choix entre eux étant effectué en fonction du mode courant. Le changement de mode est notifié par l'envoi de motifs donnés.

Vers 1900, Donald Murray développa un système télégraphique inspiré de celui de Baudot, mais remplaça le clavier à cinq touches par un clavier semblable à celui d'une machine à écrire. Il changea aussi le code, cette fois-ci avec pour objectif de faciliter la mécanique de son système. Son code, légèrement modifié, devint le code CCITT#2 (voir sectop 4.1, page 7).

1. Cette table a quelques inconsitances. Et si j'ai généralement pu recouper mes informations, pour cette table ma source est unique, [Jen04], et elle émet elle même des doutes sur la table, concluant : « It is entirely possible my second-hand source(10) (and now this document) is propagating authoritative-seeming but wrong data. » La source citée, [Nor], était inaccessible au moment de la rédaction de ce document.

			Lettres		Chiffres	
HEX			0	1	0	1
	BIN		0	1	0	1
		OCT	*0*	*2*	*0*	*2*
0	0000	*0*	0 (1)	16 LS	0 (1)	16 LS
1	0001	*1*	1 A	17 C	1 1	17 .
2	0010	*2*	2 E	18 X	2 2	18 ,
3	0011	*3*	3 É	19 Z	3 &	19 :
4	0100	*4*	4 I	20 S	4 3	20 ;
5	0101	*5*	5 O	21 T	5 4	21 !
6	0110	*6*	6 U	22 W	6 O	22 ?
7	0111	*7*	7 Y	23 V	7 5	23 ’
8	1000	*0*	8 FS	24 RUB	8 FS	24 RUB
9	1001	*1*	9 J	25 K	9 6	25 (
A	1010	*2*	10 G	26 M	10 7	26)
B	1011	*3*	11 H	27 L	11 H	27 =
C	1100	*4*	12 B	28 R	12 8	28 –
D	1101	*5*	13 C	29 Q	13 9	29 /
E	1110	*6*	14 F	30 Z	14 F	30 N°
F	1111	*7*	15 D	31 P	15 0	31 %
		OCT	*1*	*3*	*1*	*3*

(1) pas utilisé
LS Letter-shift (passage aux lettres)
FS Figure-shift (passage aux chiffres)
RUB Rubout (annulation du caractère précédent)

TABLE 2: Le code Baudot

L'automatisation du traitement des données est lui plus récent. C'est durant les années 1880, Herman Hollerith développe les premières tabulatrices qui seront utilisées pour le traitement du recensement américain de 1890. Il fonde alors une compagnie qui deviendra après quelques fusions IBM. Dans ces systèmes, les informations sont codées par des trous dans des cartes. Les premiers codes pour les cartes perforées avaient 12 caractères : les 10 chiffres et deux caractères de contrôle. Ce code fut étendu à 40 caractères dans les années 30, à 48 caractères dans les années 50 avec les premiers ordinateurs. À la même époque, on ajoute au code Hollerith, défini par des combinaison de trous dans les 12 rangées des cartes, un codage numérique sur 6 bits le code BCDIC (*Binary Coded Decimal Interchange Code*).

Durant les années 60, les insuffisances d'un code sur 6 bits deviennent de moins en moins supportables. Pour les systèmes 360, IBM développe alors EBCDIC (*Extended BCDIC*), un code 8 bits ayant une structure proche du BCDIC et un codage pour les cartes compatible avec lui – le codage 8 bits n'est pas compatible avec le codage 6 bits de BCDIC, mais la transformation de l'un en autre est simple à implémenter en hardware.

À la même époque, l'institut de normalisation américain (qui change de nom une série de fois sur la période) développe le code 7 bits ASCII (*American Standard Code for Information Interchange*). Ce code, et une série de variantes nationales remplaçant quelques caractères, sera normalisé par l'ISO sous le nom ISO 646 (généralement accompagné d'un code de deux lettres désignant la variante). Plus tard, une série de code 8 bits – dont les 128 premiers caractères reprennent ceux de l'ASCII – seront normalisés par l'ISO sous le nom ISO 8851 (généralement accompagnés d'un nombre indiquant précisément le code). Une autre norme, ISO 2022 [ECMA35], décrit une structure pour les codes – structure respectée par ISO 646 et ISO 8851 – permettant le changement d'un répertoire à un autre et l'utilisation de répertoires beaucoup plus grands, nécessaires pour des écritures idéographiques comme le Chinois, Japonais et le Coréen. Naturellement, ces écritures ont aussi des codages ne respectant pas la structure d'ISO 2022.

La multiplicité des codes existants pose naturellement des problèmes. Dans les années 1990, deux projets se sont donnés pour tâche d'établir un répertoire et un code universel – l'ISO et la norme ISO 10646, et le consortium Unicode. Après quelque temps d'une certaine rivalité, ces deux projets ont fini par collaborer et avoir le même répertoire et utiliser les mêmes codets.

3 Structure

Pour décrire les jeux de caractères codés, il faut avoir un modèle permettant d'en donner la structure. En voici un, inspiré de ceux de [ECMA35], [RFC2130] et [UTR17] mais qui n'est identique à aucun d'eux ; les objectifs étant différents.

Le *caractère* est l'unité de base décrivant ce qui sera codé.

Un *jeu de caractères* ou encore *répertoire de caractères* est un ensemble de caractères complet pour un usage donné.

Un *jeu de caractères codé*, c'est un répertoire de caractères auquel on ajoute une association d'un nombre à chaque caractère. Ce nombre est appelé *codet*.

Le *mécanisme de codage* permet de passer d'une suite de caractères, provenant éventuellement de plusieurs jeux de caractères codés, en une suite de *bytes*, l'unité de base

de la représentation codée [2].

Ensuite, il faut parfois tenir des des problèmes qui existent chaque fois qu'on a à manipuler des données.

Le byte du mécanisme de codage peut ne pas avoir de représentation naturelle sur le système cible. Il faut alors les *sérialiser*, en ajoutant du bourrage, en les découpant ou en les regroupant.

Finalement, quand on veut transmettre le résultat, on peut avoir à effectuer un *surcodage* pour respecter les contraintes du protocole utilisé.

3.1 Caractères

Les caractères peuvent être des lettres, des ligatures typographiques (« fi ») ou linguistiques (« œ »), des idéogrammes ou des idéophonogrammes des écritures non alphabétiques, des chiffres, des symboles mathématiques ou autre,... en fait n'importe quoi qui a été jugé digne d'être encodé.

Il ne faut pas confondre caractère et glyphe. Un *glyphe*, c'est un dessin. « $\mathscr{A}$ », « A », « A » sont trois glyphes différents. Le caractère, c'est la notion abstraite. Les trois glyphes précédents sont des représentations du même caractère : la lettre A majuscule.

Et si généralement plusieurs glyphes peuvent représenter un caractère, parfois, deux caractères distincts peuvent être représentés par un même glyphe. Par exemple, le « A » de tout à l'heure pourrait être aussi bien un alpha majuscule.

Les choix faits sur ce qui constitue un caractère varient suivant les codes. Par exemple, un « à » peut être codé comme un caractère, comme deux caractères (un « a » et un accent grave) comme trois caractères (un « a », un espace arrière et un accent grave).

Les *caractères de commande* (souvent appelés *caractères de contrôle*) font partie des caractères qui sont là parce qu'ils ont été jugés digne d'être encodés. C'est dans ces caractères qu'on trouve le mieux la trace des applications envisagées pour le jeu de caractères. On y trouve des caractères servant à gérer le formatage (passage à la ligne, tabulation), des caractères servant à séparer des blocs d'information et à gérer les protocoles de communication, des caractères servant à contrôler des périphériques (retour chariot par exemple), des caractères servant à gérer l'encodage, et d'autres choses choses encore.

3.2 Jeu de caractères codé

On emploie aussi les termes de *page de code* (surtout chez IBM et Microsoft), *charset*, *codeset* pour jeu de caractères codé.

Et codet n'est pas non plus dénué de synonymes : *valeur scalaire*, *élément de code*, *position de code*, *point de code* et souvent même *code* malgré l'ambiguïté possible.

Chaque caractère du répertoire doit avoir un seul codet, mais un codet donné peut correspondre à plusieurs caractères. Les jeux de caractères associant plusieurs caractères

2. Le mot *byte* est aussi utilisé avec au moins deux autres acceptions : la plus petite unité adressable par un ordinateur et une chaîne d'exactement 8 bits. Dans ce dernier sens, il faut préférer le mot *octet* dont c'est l'unique signification.

à un codet sont des jeux modaux : un mode indique quel interprétation doit être utilisée. Le changement de mode s'effectuant souvent à l'aide de caractères de contrôle.

3.3 Mécanisme de codage

Le mécanisme de codage le plus simple, c'est la fonction identité. Il est souvent utilisé implicitement. Certains mécanismes permettent de combiner des jeux de caractères codés différents. Cela peut être très simple – simplement la fonction identité à nouveau en imposant que les codets soient différents – ou plus compliqué (voir 4.5).

Les grands jeux de caractères (nécessaire pour les écritures idéographiques) ont un problème avec l'utilisation de la fonction identité : ils exigeraient alors des bytes de grandes tailles. Ils prévoient donc des moyens d'utiliser des bytes plus petits, mais en utilisant parfois plusieurs bytes pour un caractère.

3.4 Sérialisation

Si la taille du byte ne correspond pas à une unité manipulée naturellement par le système, il faut alors prévoir comment la représentation aura lieu.

On peut alors avoir à regrouper plusieurs bytes en un mot. Par exemple, le PDP-10 a des mots de 36 bits. Il a été utilisé avec des mécanismes de codage utilisant des bytes de 6 bits (on mettait 6 caractères par mot), de 7 bits (on mettait 5 caractères par mot, et un bit était inutilisé) et de 9 bits (4 caractères par mot). Dans le cas du PDP-10, la position des bytes dans le mot a été déterminée par le jeu d'instruction de ce processeur qui a une notion de pointeur vers bytes où les pointeurs en question contienne le nombre de bits du byte, de 1 à 36.

On peut aussi avoir à découper un byte en plusieurs unités, par exemple Unicode définit un mécanisme de codage appelé UTF-16 qui utilise des bytes de 16 bits. Pour les représenter sur un ordinateur manipulant des octets on a le choix : si on place la partie la plus significative du byte en premier, la sérialisation est *gros-boutiste*, si la partie la moins significative est en premier, elle est *petit-boutiste*.

3.5 Surcodage

Le surcodage est généralement transparent pour l'utilisateur ; et le système qui l'effectue n'a pas nécessairement connaissance qu'il est en train de manipuler des caractères.

L'exemple le plus commun de surcodage visible – heureusement de moins en moins – est avec le protocole SMTP [RFC821], utilisé pour transmettre les couriels. Il est définit comme opérant sur des octets, mais sans garantie de conservation du bit de poids fort. Ca fonctionne bien avec l'ASCII – qui est un code sur 7 bits – mais données utilisant un code sur 8 bits peuvent être modifiée dans le transport.

Un protocole annexe, MIME [RFC1521], résouds ce problème, et d'autres, tout en conservant une bonne compatibilité avec les systèmes antérieurs. Bonne compatibilité en ce sens qu'avec un logiciel qui ne comprend pas MIME (ce qui est maintenant rare,

sauf quand les mécanismes de MIME sont utilisés là où ce n'est pas prévu), on se retrouve avec une partie du message dégradée, mais souvent encore compréhensible.

Il définit deux surcodages qui peuvent être utilisés pour transmettre un code sur 8 bits : un (*quoted printable*) à utiliser plutôt quand les données sont principalement des codets avec les 8^{e}bit à 0, seuls ces codets sont transformés ce qui laisse souvent le message surcodé compréhensible (par exemple le « é » d'ISO 8859-1 devient « =E9 » dans les couriels) ; le deuxième (*base64*) est à utiliser dans les autres cas, et le message est alors incompréhensible.

4 Quelques jeux de caractères codés

Ce document ne se veut pas un répertoire de codes (voir [ISO-IR], [IBMa], [IBMb], [IAN] ou [RFC1345] pour des répertoires récents et [Win] pour une collection avec un accent plus historique avec en particulier des codes pour cartes et rubans perforés). Les jeux de caractères qui sont présentés ici ont été choisi pour des raisons de pertinence historique ou pratique.

4.1 CCITT#2

Ce code (voir table 3) a été conçu pour remplacer le code de Baudot lors du développement de terminaux ayant un clavier semblable à celui d'une machine à écrire (les claviers des terminaux utilisant le code de Baudot avaient 5 touches, une pour chaque bit). Le choix des codets fut effectué en tenant compte de la fréquence des caractères avec pour objectif de minimiser les opérations mécaniques.

Références : [ITU, Jen04].

4.2 BCDIC

Si le code CCITT#2 provient du monde des transmission en général et du télégraphe en particulier, le code BCDIC (voir table 4) a pour origine le traitement des données. Les codes pour cartes perforées ne sont pas l'objet de ce document (voir [Mac80, Win, Jona, Jonb] par exemple), mais comprendre la structure du BCDIC et donc de l'EBCDIC est impossible sans rien en connaître.

Les cartes perforées ont quasiment universellement 12 rangées (appelées ici 0 à 9, X et Y – le nom des deux dernières rangées varie suivant les sources), et chaque caractère est codé sur une colonne. Dans la famille des codes pour cartes appelée “Hollerith”, au début, seuls les chiffres étaient codés par un trou dans la rangée correspondante. Les deux autres rangées avaient un usage varié. Pour coder les lettres, on utilisa des combinaisons de deux trous : un des rangées 0, X ou Y (ces rangées sont appelées rangées de zone) et un des rangées des autres chiffres (appelées rangées de chiffres). Par la suite, des combinaisons ajoutant un troisième trou dans la rangée 8 furent utilisées pour coder d'autres caractères.

Le code BCDIC présenté ici (qualifié de final par Mackenzie [Mac80]) est le résultat de cette évolution chez IBM. Le code respecte la structure donnée ci-dessus : les colonnes

			Lettres		Chiffres	
HEX			0	1	0	1
	BIN		0	1	0	1
		OCT	*0*	*2*	*0*	*2*
0	0000	*0*	0 (2)	16 E	0 (2)	16 3
1	0001	*1*	1 T	17 Z	1 5	17 +
2	0010	*2*	2 CR	18 D	2 CR	18 WRU
3	0011	*3*	3 O	19 B	3 9	19 ?
4	0100	*4*	4 SP	20 S	4 SP	20 ’
5	0101	*5*	5 H	21 Y	5 (1)	21 6
6	0110	*6*	6 N	22 F	6 ,	22 (1)
7	0111	*7*	7 M	23 X	7 .	23 /
8	1000	*0*	8 LF	24 A	8 LF	24 -
9	1001	*1*	9 L	25 W	9)	25 2
A	1010	*2*	10 R	26 J	10 4	26 Bell
B	1011	*3*	11 G	27 FS	11 (1)	27 FS
C	1100	*4*	12 I	28 U	12 8	28 7
D	1101	*5*	13 P	29 Q	13 0	29 1
E	1110	*6*	14 C	30 K	14 :	30 (
F	1111	*7*	15 V	31 LS	15 =	31 LS
		OCT	*1*	*3*	*1*	*3*

(1) usage local (lettres accentuées par exemple)
(2) pas utilisé
WRU Who are you
CR Carriage-return (retour chariot)
LF Line-feed (passage à la ligne)
LS Letter-shift (passage aux lettres)
FS Figure-shift (passage aux chiffres)
SP Space (espace)

TABLE 3: Le code CCITT#2 ou alphabet télégraphique international n° 2

	HEX		0	1	2	3
	BIN		00	01	10	11
		OCT	*0*	*2*	*4*	*6*
0	0000	*0*	0 (SP)	16 ƀ	32 -	48 & ou +
1	0001	*1*	1 1	17 /	33 J	49 A
2	0010	*2*	2 2	18 S	34 K	50 B
3	0011	*3*	3 3	19 T	35 L	51 C
4	0100	*4*	4 4	20 U	36 M	52 D
5	0101	*5*	5 5	21 V	37 N	53 E
6	0110	*6*	6 6	22 W	38 O	54 F
7	0111	*7*	7 7	23 X	39 P	55 G
8	1000	*0*	8 8	24 Y	40 Q	56 H
9	1001	*1*	9 9	25 Z	41 R	57 I
A	1010	*2*	10 0	26 ǂ	42 !	58 ?
B	1011	*3*	11 # ou =	27 ,	43 $	59 .
C	1100	*4*	12 @ ou ’	28 % ou (	44 *	60 ¤ ou)
D	1101	*5*	13 :	29 γ	45]	61 [
E	1110	*6*	14 >	30 \	46 ;	62 <
F	1111	*7*	15 ✓	31 ǂǂ	47 Δ	63 ǂǂǂ
		OCT	*1*	*3*	*5*	*7*

SP	Space (espace)
ƀ	Substitute Blank
Δ	Mode Change
γ	Word Separator
ǂ	Record Mark
ǂǂ	Group Mark
ǂǂǂ	Segment Mark
✓	Tape Mark
& # @ % ¤	version commerciale
+ = ’ ( )	version FORTRAN

TABLE 4: Le code BCDIC

correspondent à une combinaison donnée dans les rangées de zones et les rangées correspondent à une combinaison de trous dans les rangées de chiffres [3].

Certains caractères de contrôle étaient interprétés dans tous les contextes par certains lecteur de bandes magnétiques, ce qui fait qu'ils n'étaient en pratique pas stockables sur bandes.

Il y a eu quelques variantes de ce code. Deux sont données ici : la variante commerciale, utilisée pour ce genre d'application, et la variante FORTRAN, utilisée pour programmer dans ce langage.

4.3 EBCDIC

Au début des années 60, les insuffisances des jeux de caractères 6 bits tels que le BCDIC était bien connues : d'une part, ils n'avaient pas assez de caractères graphiques, donnant donc naissance à des variantes, d'autres part ils n'avaient pas assez de caractères de contrôle, donnant donc naissance à des codes particuliers pour communiquer avec les équipements, et donc la nécessité de conversions.

Une fois choisi de baser le System/360 autour d'une unité d'adressage de 8 bits, c'était l'occasion de concevoir un jeu de caractères codé qui s'affranchissait de ces limitations en profitant des 256 points de code disponibles. C'était a peu près au même moment que le développement de l'ASCII commençait, mais avec des contraintes différentes.

Pour les clients d'IBM – et donc pour IBM – la compatibilité avec BCDIC était d'une importance majeure. Ils avaient déjà beaucoup d'information ainsi codée sur des cartes perforées. Avoir une conversion simple était une nécessité. Et respecter la structure globale aussi : une technique habituelle à l'époque était d'utiliser indépendamment les rangées de zones et de chiffres. C'est-à-dire que les caractères – ou plutôt les combinaisons de trous correspondant à ces caractères – ? et A à I étaient utilisés pour les chiffres 0 à 9 combinés avec un drapeau, de même que les caractères ! et J à R. Conserver le même code pour les cartes perforées pour les lettres étaient une des contraintes que les concepteurs d'EBCDIC s'étaient imposé, et conserver une équivalence simple entre les chiffres combinés avec d'autres informations en était une autre. On a l'explication de pourquoi les lettres sont ainsi codées en EBCDIC – mais par rapport au BCDIC, elles sont au moins dans l'ordre alphabétique.

Des considérations du même genre, de compatibilité et d'autres, ont fini par donner le code montré en table 5. Il en existe des variantes, adaptées aux usages nationaux. Dans la variété des codes EBCDIC, il est à remarquer que certains caractères courant n'ont pas le même code dans toutes les versions.

4.4 ASCII

Ce code (voir table 6) est un code 7 bit développé aux États-Unis durant les années 60 (avec des étapes en 63, 65, 67 et 68 ; la version présentée ici est celle de 68 d'après [Mac80] [4]. Ses concepteurs étaient conscients qu'il était peu probable que des systèmes

3. Il y a quelques exceptions causées par le fait que 0 est à la fois une rangée de zone et de chiffres.
4. Voir 5.2 pour l'utilisation de ¦ plutôt que de |.

HEX			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	BIN		00				01				10				11			
			00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
		OCT	*0*	*2*	*4*	*6*	*10*	*12*	*14*	*16*	*20*	*22*	*24*	*26*	*30*	*32*	*34*	*36*
0	0000	*0*	0 NUL	16 DLE	32 PAD	48 DCS	64 SP	80 &	96 -	112	128	144	160	176	192 {	208 }	224 \	240 0
1	0001	*1*	1 SOH	17 DC1	33 HOP	49 PUL	65	81	97 /	113	129 a	145 j	161 ~	177	193 A	209 J	225	241 1
2	0010	*2*	2 STX	18 DC2	34 BHP	50 SYN	66	82	98	114	130 b	146 k	162 s	178	194 B	210 K	226 S	242 2
3	0011	*3*	3 ETX	19 DC3	35 NBH	51 STS	67	83	99	115	131 c	147 l	163 t	179	195 C	211 L	227 T	243 3
4	0100	*4*	4 ST	20 OC	36 IND	52 CCH	68	84	100	116	132 d	148 m	164 u	180	196 D	212 M	228 U	244 4
5	0101	*5*	5 HT	21 NEL	37 LF	53 MW	69	85	101	117	133 e	149 n	165 v	181	197 E	213 N	229 V	245 5
6	0110	*6*	6 SSA	22 BS	38 ETB	54 SPA	70	86	102	118	134 f	150 o	166 w	182	198 F	214 O	230 W	246 6
7	0111	*7*	7 DEL	23 ESA	39 ESC	55 EOT	71	87	103	119	135 g	151 p	167 x	183	199 G	215 P	231 X	247 7
8	1000	*0*	8 EPA	24 CAN	40 HTS	56 SOS	72	88	104	120	136 h	152 q	168 y	184	200 H	216 Q	232 Y	248 8
9	1001	*1*	9 RI	25 EM	41 HTJ	57 SGC	73	89	105	121 `	137 i	153 r	169 z	185	201 I	217 R	233 Z	249 9
A	1010	*2*	10 SS2	26 PU2	42 VTS	58 SCI	74 ¢	90 !	106 ¦	122 :	138	154	170	186	202	218	234	250
B	1011	*3*	11 VT	27 SS3	43 PLD	59 CSI	75 .	91 $	107 ,	123 #	139	155	171	187	203	219	235	251
C	1100	*4*	12 FF	28 IS4	44 PLU	60 DC4	76 <	92 *	108 %	124 @	140	156	172	188	204	220	236	252
D	1101	*5*	13 CR	29 IS3	45 ENQ	61 NAK	77 (	93)	109 _	125 ’	141	157	173	189	205	221	237	253
E	1110	*6*	14 SO	30 IS2	46 ACQ	62 PM	78 +	94 ;	110 >	126 =	142	158	174	190	206	222	238	254
F	1111	*7*	15 SI	31 IS1	47 BEL	63 SUB	79 |	95 ¬	111 ?	127 "	143	159	175	191	207	223	239	255 APC
		OCT	*1*	*3*	*5*	*7*	*11*	*13*	*15*	*17*	*21*	*23*	*25*	*27*	*31*	*33*	*35*	*37*

TABLE 5: Le code EBCDIC

utilisent exactement 7 bits de manière interne [Mac80, P. 217]. Le choix d'un code sur 7 bits a cependant été fait car il satisfaisait les besoins de la plupart des utilisations prévues, que donc un code à 8 bits aurait été perçu comme imposant un coût inutile à la majorité et finalement que l'ajout d'un huitième bit ne permettait plus l'utilisation de certaines techniques comme les rubans perforés (qui permettaient jusqu'à 8 bit mais étaient utilisés avec un bit de parité).

Par la suite, il fut normalisé par l'ISO (sous le nom ISO 646, ECMA a publié une norme techniquement équivalente sous le nom ECMA 6 qui a l'avantage d'être disponible publiquement [ECMA6]). Cette norme prévoit la possibilité pour des variantes nationales de changer certains caractères pour correspondre à des nécessités nationales (par exemple

HEX			0	1	2	3	4	5	6	7
	BIN		000	001	010	011	100	101	110	111
		OCT	*0*	*2*	*4*	*6*	*10*	*12*	*14*	*16*
0	0000	*0*	0 NUL	16 DLE	32 SP	48 0	64 @	80 P	96 `	112 p
1	0001	*1*	1 SOH	17 DC1	33 !	49 1	65 A	81 Q	97 a	113 q
2	0010	*2*	2 STX	18 DC2	34 "	50 2	66 B	82 R	98 b	114 r
3	0011	*3*	3 LTX	19 DC3	35 #	51 3	67 C	83 S	99 c	115 s
4	0100	*4*	4 EOT	20 DC4	36 $	52 4	68 D	84 T	100 d	116 t
5	0101	*5*	5 ENO	21 NAK	37 %	53 5	69 E	85 U	101 e	117 u
6	0110	*6*	6 ACK	22 SYN	38 &	54 6	70 F	86 V	102 f	118 v
7	0111	*7*	7 BEL	23 ETB	39 ’	55 7	71 G	87 W	103 g	119 w
8	1000	*0*	8 BS	24 CAN	40 (	56 8	72 H	88 X	104 h	120 x
9	1001	*1*	9 HT	25 EM	41)	57 9	73 I	89 Y	105 i	121 y
A	1010	*2*	10 LF	26 SUB	42 *	58 :	74 J	90 Z	106 j	122 z
B	1011	*3*	11 VT	27 ESC	43 +	59 ;	75 K	91 [	107 k	123 {
C	1100	*4*	12 FF	28 FS	44 ,	60 <	76 L	92 \	108 l	124 ¦
D	1101	*5*	13 CR	29 GS	45 -	61 =	77 M	93]	109 m	125 }
E	1110	*6*	14 SO	30 RS	46 .	62 >	78 N	94 ^	110 n	126 ~
F	1111	*7*	15 SI	31 US	47 /	63 ?	79 O	95 _	111 o	127 DEL
		OCT	*1*	*3*	*5*	*7*	*11*	*13*	*15*	*17*

TABLE 6: Le code ASCII

l’ajout de lettres accentuées). De même, les variantes nationales peuvent prévoir des séquences combinantes (par exemple la variante française prévoyait l’utilisation de ÂS a pour le caractère â).

Les caractères pouvant avoir une définition nationale sont #, $, @, [, \,], ^, `, {, |, }, ~ et de plus les choix pour # et sont restreints (# peut être remplacé par £, et $ par ¤ ; ce dernier caractère – le symbole monétaire – a d’ailleurs été présent dans certaines versions de référence).

Il y eut plusieurs révisions de cette norme. La plus importante fut vraisemblablement de la mettre en conformité avec la structure de ISO 2022 (voir 4.5) en enlevant la description des caractères de contrôle pour la confier à d’autres normes (ISO 6429 alias ECMA 48 [ECMA48] par exemple). Dans cette forme, ce jeu de caractères ne comporte plus que 94 caractères, ceux de codes 33 à 126 (l’espace et DEL font partie de ISO 2022).

L'existence de variantes nationales – parfois plusieurs variantes par pays – a posé suffisemment de problèmes pratiques que la France a fini par retirer ses variantes et adopter la version de référence.

Ce jeu de caractères a servi de base à beaucoup d'autres qui ont rempli les 128 positions laissées libres si on le code avec une unité faisant 8 bits ; ces jeux sont parfois appelé ASCII étendu.

4.5 ISO 2022

La norme ISO 2022 (disponible plus faciment en tant que norme ECMA 35 [ECMA35]) décrit un mécanisme de codage permettant de coder plusieurs jeux de caractères codé respectant une certaine structure dans un même flux.

Trois types de charset sont codables :

– des jeux de 32 caractères de commandes ayant les codets 0 à 31.
– des jeux de 94 caractères graphiques ayant les codets 33 à 126.
– des jeux de 96 caractères graphiques ayant les codets 32 à 127.

4.6 ISO 8859-1

Dés qu'on utilise une autre langue que l'anglais – en fait, même dès qu'on utilise un anglais cultivé – on a besoin de caractères absents du répertoire de l'ASCII. Les variantes nationales d'ISO 646 étaient une tentative de répondre à ces besoins ; mais étant *nationales*, elles posaient des problèmes dès qu'il fallait communiquer entre nations. Elles posaient aussi d'autres problèmes quand on avait besoin des caractères remplacés, par exemple pour écrire des programmes dans un langage de programmation qui faisait usage des ces caractères.

L'ASCII est un jeu de caractères sur 7 bits. Entre son introduction et le moment où les problèmes évoqués ci-dessus sont devenus criant, les ordinateurs basés sur un byte de 8 bits sont devenus monnaie courante. Des extensions de l'ASCII à 8 bits étaient naturelles.

La normes ISO 8859 en normalise une série, valables pour des ensembles de langues différents.

HEX			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	BIN		00				01				10				11			
			00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
		OCT	*0*	*2*	*4*	*6*	*10*	*12*	*14*	*16*	*20*	*22*	*24*	*26*	*30*	*32*	*34*	*36*
0	0000	*0*	0	16	32 (SP)	48 0	64 @	80 P	96 `	112 p	128	144	160 (NBS)	176 °	192 À	208 Ð	224 à	240 ð
1	0001	*1*	1	17	33 !	49 1	65 A	81 Q	97 a	113 q	129	145	161 ¡	177 ±	193 Á	209 Ñ	225 á	241 ñ
2	0010	*2*	2	18	34 "	50 2	66 B	82 R	98 b	114 r	130	146	162 ¢	178 ²	194 Â	210 Ò	226 â	242 ò
3	0011	*3*	3	19	35 #	51 3	67 C	83 S	99 c	115 s	131	147	163 £	179 ³	195 Ã	211 Ó	227 ã	243 ó
4	0100	*4*	4	20	36 $	52 4	68 D	84 T	100 d	116 t	132	148	164 ¤	180 ´	196 Ä	212 Ô	228 ä	244 ô
5	0101	*5*	5	21	37 %	53 5	69 E	85 U	101 e	117 u	133	149	165 ¥	181 µ	197 Å	213 Õ	229 å	245 õ
6	0110	*6*	6	22	38 &	54 6	70 F	86 V	102 f	118 v	134	150	166 ¦	182 ¶	198 Æ	214 Ö	230 æ	246 ö
7	0111	*7*	7	23	39 ’	55 7	71 G	87 W	103 g	119 w	135	151	167 §	183 ·	199 Ç	215 ×	231 ç	247 ÷
8	1000	*0*	8	24	40 (	56 8	72 H	88 X	104 h	120 x	136	152	168 ¨	184 ¸	200 È	216 Ø	232 è	248 ø
9	1001	*1*	9	25	41)	57 9	73 I	89 Y	105 i	121 y	137	153	169 ©	185 ¹	201 É	217 Ù	233 é	249 ù
A	1010	*2*	10	26	42 *	58 :	74 J	90 Z	106 j	122 z	138	154	170 ª	186 º	202 Ê	218 Ú	234 ê	250 ú
B	1011	*3*	11	27	43 +	59 ;	75 K	91 [	107 k	123 {	139	155	171 «	187 »	203 Ë	219 Û	235 ë	251 û
C	1100	*4*	12	28	44 ,	60 <	76 L	92 \	108 l	124 |	140	156	172 ¬	188 ¼	204 Ì	220 Ü	236 ì	252 ü
D	1101	*5*	13	29	45 -	61 =	77 M	93]	109 m	125 }	141	157	173 (SHY)	189 ½	205 Í	221 Ý	237 í	253 ý
E	1110	*6*	14	30	46 .	62 >	78 N	94 ^	110 n	126 ~	142	158	174 ®	190 ¾	206 Î	222 Þ	238 î	254 þ
F	1111	*7*	15	31	47 /	63 ?	79 O	95 _	111 o	127 (DEL)	143	159	175 ¯	191 ¿	207 Ï	223 ß	239 ï	255 ÿ
		OCT	*1*	*3*	*5*	*7*	*11*	*13*	*15*	*17*	*21*	*23*	*25*	*27*	*31*	*33*	*35*	*37*

TABLE 7: Le code ISO 8859-1 ou Latin 1

HEX			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	BIN		00				01				10				11			
			00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
		OCT	*0*	*2*	*4*	*6*	*10*	*12*	*14*	*16*	*20*	*22*	*24*	*26*	*30*	*32*	*34*	*36*
0	0000	*0*	0	16	32 SP	48 0	64 @	80 P	96 `	112 p	128	144	160 NBS	176 °	192 À	208 Đ	224 à	240 ð
1	0001	*1*	1	17	33 !	49 1	65 A	81 Q	97 a	113 q	129	145	161 ¡	177 ±	193 Á	209 Ñ	225 á	241 ñ
2	0010	*2*	2	18	34 "	50 2	66 B	82 R	98 b	114 r	130	146	162 ¢	178 ²	194 Â	210 Ò	226 â	242 ò
3	0011	*3*	3	19	35 #	51 3	67 C	83 S	99 c	115 s	131	147	163 £	179 ³	195 Ã	211 Ó	227 ã	243 ó
4	0100	*4*	4	20	36 $	52 4	68 D	84 T	100 d	116 t	132	148	164 €	180 Ž	196 Ä	212 Ô	228 ä	244 ô
5	0101	*5*	5	21	37 %	53 5	69 E	85 U	101 e	117 u	133	149	165 ¥	181 µ	197 Å	213 Õ	229 å	245 õ
6	0110	*6*	6	22	38 &	54 6	70 F	86 V	102 f	118 v	134	150	166 Š	182 ¶	198 Æ	214 Ö	230 æ	246 ö
7	0111	*7*	7	23	39 ’	55 7	71 G	87 W	103 g	119 w	135	151	167 §	183 ·	199 Ç	215 ×	231 ç	247 ÷
8	1000	*0*	8	24	40 (	56 8	72 H	88 X	104 h	120 x	136	152	168 š	184 ž	200 È	216 Ø	232 è	248 ø
9	1001	*1*	9	25	41)	57 9	73 I	89 Y	105 i	121 y	137	153	169 ©	185 ¹	201 É	217 Ù	233 é	249 ù
A	1010	*2*	10	26	42 *	58 :	74 J	90 Z	106 j	122 z	138	154	170 ª	186 º	202 Ê	218 Ú	234 ê	250 ú
B	1011	*3*	11	27	43 +	59 ;	75 K	91 [	107 k	123 {	139	155	171 «	187 »	203 Ë	219 Û	235 ë	251 û
C	1100	*4*	12	28	44 ,	60 <	76 L	92 \	108 l	124 |	140	156	172 ¬	188 Œ	204 Ì	220 Ü	236 ì	252 ü
D	1101	*5*	13	29	45 -	61 =	77 M	93]	109 m	125 }	141	157	173 SHY	189 œ	205 Í	221 Ý	237 í	253 ý
E	1110	*6*	14	30	46 .	62 >	78 N	94 ^	110 n	126 ~	142	158	174 ®	190 Ÿ	206 Î	222 Þ	238 î	254 þ
F	1111	*7*	15	31	47 /	63 ?	79 O	95 _	111 o	127 DEL	143	159	175 ¯	191 ¿	207 Ï	223 ß	239 ï	255 ÿ
		OCT	*1*	*3*	*5*	*7*	*11*	*13*	*15*	*17*	*21*	*23*	*25*	*27*	*31*	*33*	*35*	*37*

TABLE 8: Le code ISO 8859-15 ou Latin 9

4.7 ISO 8859-15

4.8 CP 1252

4.9 Unicode

5 Au sujet de quelques caractères

5.1 La barre oblique inversée

5.2 Les barres verticales

Toutes les autres références que j’ai sont postérieures et utilisent | plutôt que ¦ pour le caractère de code 124 ; je ne sais pas si c’est une différence de fontes – mais cet ou-

HEX			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	BIN		00				01				10				11			
			00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11
		OCT	*0*	*2*	*4*	*6*	*10*	*12*	*14*	*16*	*20*	*22*	*24*	*26*	*30*	*32*	*34*	*36*
0	0000	*0*	0	16	32 (SP)	48 0	64 @	80 P	96 `	112 p	128 €	144	160 (NBS)	176 °	192 À	208 Đ	224 à	240 ð
1	0001	*1*	1	17	33 !	49 1	65 A	81 Q	97 a	113 q	129	145 ‘	161 ¡	177 ±	193 Á	209 Ñ	225 á	241 ñ
2	0010	*2*	2	18	34 "	50 2	66 B	82 R	98 b	114 r	130 ‚	146 ’	162 ¢	178 ²	194 Â	210 Ò	226 â	242 ò
3	0011	*3*	3	19	35 #	51 3	67 C	83 S	99 c	115 s	131 *ƒ*	147 “	163 £	179 ³	195 Ã	211 Ó	227 ã	243 ó
4	0100	*4*	4	20	36 $	52 4	68 D	84 T	100 d	116 t	132 „	148 ”	164 ¤	180 ´	196 Ä	212 Ô	228 ä	244 ô
5	0101	*5*	5	21	37 %	53 5	69 E	85 U	101 e	117 u	133 …	149 •	165 ¥	181 µ	197 Å	213 Õ	229 å	245 õ
6	0110	*6*	6	22	38 &	54 6	70 F	86 V	102 f	118 v	134 †	150 –	166 ¦	182 ¶	198 Æ	214 Ö	230 æ	246 ö
7	0111	*7*	7	23	39 ’	55 7	71 G	87 W	103 g	119 w	135 ‡	151 —	167 §	183 ·	199 Ç	215 ×	231 ç	247 ÷
8	1000	*0*	8	24	40 (	56 8	72 H	88 X	104 h	120 x	136 ^	152 ~	168 ¨	184 ¸	200 È	216 Ø	232 è	248 ø
9	1001	*1*	9	25	41)	57 9	73 I	89 Y	105 i	121 y	137 ‰	153 ™	169 ©	185 ¹	201 É	217 Ù	233 é	249 ù
A	1010	*2*	10	26	42 *	58 :	74 J	90 Z	106 j	122 z	138 Š	154 š	170 ª	186 º	202 Ê	218 Ú	234 ê	250 ú
B	1011	*3*	11	27	43 +	59 ;	75 K	91 [	107 k	123 {	139 ‹	155 ›	171 «	187 »	203 Ë	219 Û	235 ë	251 û
C	1100	*4*	12	28	44 ,	60 <	76 L	92 \	108 l	124 |	140 Œ	156 œ	172 ¬	188 ¼	204 Ì	220 Ü	236 ì	252 ü
D	1101	*5*	13	29	45 -	61 =	77 M	93]	109 m	125 }	141	157	173 (SHY)	189 ½	205 Í	221 Ý	237 í	253 ý
E	1110	*6*	14	30	46 .	62 >	78 N	94 ^	110 n	126 ~	142 Ž	158	174 ®	190 ¾	206 Î	222 Þ	238 î	254 þ
F	1111	*7*	15	31	47 /	63 ?	79 O	95 _	111 o	127 (DEL)	143 ž	159	175 ¯	191 ¿	207 Ï	223 ß	239 ï	255 ÿ
		OCT	*1*	*3*	*5*	*7*	*11*	*13*	*15*	*17*	*21*	*23*	*25*	*27*	*31*	*33*	*35*	*37*

TABLE 9: Le code CP 1252 ou MS ANSI

vrage donne des tables EBCDIC où les deux caractères sont présents – ou un changement dans la norme. [RFC20], vraisemblablement une copie de X3.4-1968, décrit le caractère comme *Vertical Line*.

5.3 Le signe monétaire universel

6 C et C++

Glossaire

byte (*byte*) ([ISO2382-4] utilise *multiplet* qui est rarement employé dans l'usage courant)

1/ chaîne composée d'un certain nombre de bits traitée comme un tout et représentant habituellement un caractère ou une partie de caractère [ISO2382-4].

2/ la plus petite unité adressable par un ordinateur. En particulier quand le mot-machine n'est pas la plus petite unité adressable.

3/ chaîne de bits formée de 8 bits (voir *octet*).

caractère (*character*) Élément d'un ensemble employé pour constituer, représenter ou gérer des données [ISO2382-4].

code (*code*, *coding scheme*) ensemble de règles établissant une correspondance entre les éléments d'un premier ensemble et ceux d'un second ensemble [ISO2382-4].

codage 1/ transformation d'une représentation en une autre.

2/ règles utilisée pour effectuer cette transformation (voir *code*).

codet (*code value*, *code element*) résultat de l'application d'un code à un élément d'un jeu codé [ISO2382-4].

décodage opération inverse de l'encodage.

encodage 1/ transformation d'une représentation, considérée comme principale, en une autre.

2/ règles utilisées pour effectuer cette transformation (voir *code*).

jeu de caractères (*character set*) ensemble fini de caractères considéré comme complet pour un usage particulier.

jeu de caractères codé (*coded character set*) ensemble de caractères mis en correspondance avec un autre ensemble d'éléments selon un code [ISO2382-4].

multiplet (*byte*) voir *byte*.

octet chaîne de bits formée de 8 bits.

seizet chaîne de bits formée de 16 bits.

Lexique français-anglais

boutisme endianness
byte byte
caractère character
code code, coding scheme
codet code value, code element, code point
fonte font
gros-boutien big-endian
gros-boutiste big-endian
jeu de caractères character set
jeu de caractères codé coded character set
multiplet byte
numéro de caractère code point
page de code code page
petit-boutien little-endian
petit-boutiste little-endian
point de code code point
police font

Lexique anglais-français

big-endian gros-boutiste, gros-boutien

byte multiplet, byte

character caractère

character set jeu de caractères

code code

coded character set jeu de caractères codé

code element codet

code page page de code

code point numéro de caractère, codet, point de code

code value codet

coding scheme code

endianness boutisme

font fonte, police

little-endian petit-boutiste, petit-boutien

Voir aussi [Anda], un lexique anglais-français du vocabulaire d'Unicode.

Bibliographie

[Anda] P. Andries. Lexique unicode. [Online]. Available : http://hapax.qc.ca/dunod/Unicode_Lexique.pdf

[Andb] ——. Unicode et iso 10646 en français. [Online]. Available : http://hapax.qc.ca

[And08] ——, *Unicode 5.0 en pratique*. Dunod, 2008.

[ECMA6] *7-Bit coded Character Set*, ECMA Std. 6, 1991, technically equivalent to ISO/IEC 646. [Online]. Available : http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-006.pdf

[ECMA7] *Representation of the Standard ECMA-6 (7 bit Code) on Punched Cards*, ECMA Std. 7, 1963. [Online]. Available : http://www.ecma-international.org/publications/files/ECMA-ST-WITHDRAWN/ECMA-7,1stEdition,April1965.pdf

[ECMA35] *Character Code Structure and Extension Techniques*, ECMA Std. 35, 1994, technically equivalent to ISO/IEC 2022. [Online]. Available : http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-035.pdf

[ECMA48] *Control Functions for Coded Character Sets*, ECMA Std. 48, 1991, technically equivalent to ISO/IEC 6429. [Online]. Available : http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-048.pdf

[Har04] Y. Haralambous, *Fontes & codages*. O'Reilly, 2004.

[IAN] Character sets. IANA. [Online]. Available : http://www.iana.org/assignments/character-sets

[IBMa] Code page by GPGID. IBM. [Online]. Available : http://www-01.ibm.com/software/globalization/cp/cp_cpgid.jsp

[IBMb] Coded character set identifier. IBM. [Online]. Available : http://www-01.ibm.com/software/globalization/ccsid/ccsid_registered.jsp

[ISO-IR] International register of coded character sets. ISO. [Online]. Available : http://www.itscj.ipsj.or.jp/ISO-IR

[ISO646] *Information technology – ISO 7 bit coded character set for information interchange*, ISO Std. ISO/IEC 646 :1991, Équivalent techniquement à la norme ECMA 6, plus facile à se procurer.

[ISO2022] *Information technology – Character code structure and extension techniques*, ISO Std. ISO/IEC 2022 :1994, Équivalent techniquement à la norme ECMA 35, plus facile à se procurer.

[ISO2382-4] *Technologies de l'information – Vocabulaire, Partie 4 : Organisation des données*, ISO/IEC Std. 2382-4 :1999(E/F). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO2382-5] *Technologies de l'information – Vocabulaire, Partie 5 : Représentation des données*, ISO/IEC Std. 2382-5 :1999(E/F). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO4873] *Information technology – ISO 8-bit code for information interchange – Structure and rules for implementation*, ISO Std. ISO/IEC 4873 :1991, Équivalent techniquement à la norme ECMA 43, plus facile à se procurer.

[ISO6429] *Information technology – Control functions for coded character sets*, ISO Std. ISO/IEC 6429 :1992, Équivalent techniquement à la norme ECMA 48, plus facile à se procurer.

[ISO8859-1] *Information technology – 8-bit single-byte coded graphic character sets – Part 1 : Latin alphabet No. 1*, ISO Std. ISO/IEC 8859-1 :1998, Équivalent techniquement à une partie de la norme ECMA 94, plus facile à se procurer.

[ISO8859-2] *Information technology – 8-bit single-byte coded graphic character sets – Part 2 : Latin alphabet No. 2*, ISO Std. ISO/IEC 8859-2 :1999, Équivalent techniquement à une partie de la norme ECMA 94, plus facile à se procurer.

[ISO8859-3] *Information technology – 8-bit single-byte coded graphic character sets – Part 3 : Latin alphabet No. 3*, ISO Std. ISO/IEC 8859-3 :1999, Équivalent techniquement à une partie de la norme ECMA 94, plus facile à se procurer.

[ISO8859-4] *Information technology – 8-bit single-byte coded graphic character sets – Part 4 : Latin alphabet No. 4*, ISO Std. ISO/IEC 8859-4 :1998, Équivalent techniquement à une partie de la norme ECMA 94, plus facile à se procurer.

[ISO8859-5] *Information technology – 8-bit single-byte coded graphic character sets – Part 5 : Latin/Cyrillic alphabet*, ISO Std. ISO/IEC 8859-3 :1999.

[ISO8859-6] *Information technology – 8-bit single-byte coded graphic character sets – Part 6 : Latin/Arabic alphabet*, ISO Std. ISO/IEC 8859-6 :1999.

[ISO8859-7] *Information technology – 8-bit single-byte coded graphic character sets – Part 7 : Latin/Greek alphabet*, ISO Std. ISO/IEC 8859-7 :2003.

[ISO8859-8] *Information technology – 8-bit single-byte coded graphic character sets – Part 8 : Latin/Hebrew alphabet*, ISO Std. ISO/IEC 8859-8 :1999.

[ISO8859-9] *Information technology – 8-bit single-byte coded graphic character sets – Part 9 : Latin alphabet No. 5*, ISO Std. ISO/IEC 8859-9 :1999.

[ISO8859-10] *Information technology – 8-bit single-byte coded graphic character sets – Part 10 : Latin alphabet No. 6*, ISO Std. ISO/IEC 8859-10 :1998.

[ISO8859-11] *Information technology – 8-bit single-byte coded graphic character sets – Part 11 : Latin/Thai alphabet*, ISO Std. ISO/IEC 8859-11 :2001.

[ISO8859-13] *Information technology – 8-bit single-byte coded graphic character sets – Part 13 : Latin alphabet No. 7*, ISO Std. ISO/IEC 8859-13 :1998.

[ISO8859-14] *Information technology – 8-bit single-byte coded graphic character sets – Part 14 : Latin alphabet No. 8 (Celtic)*, ISO Std. ISO/IEC 8859-14 :1998.

[ISO8859-15] *Information technology – 8-bit single-byte coded graphic character sets – Part 15 : Latin alphabet No. 9*, ISO Std. ISO/IEC 8859-15 :1999.

[ISO8859-16] *Information technology – 8-bit single-byte coded graphic character sets – Part 15 : Latin alphabet No. 10*, ISO Std. ISO/IEC 8859-16 :2001.

[ISO10646] *Technologies de l'information – Jeu universel de caractères codés sur plusieurs octets (JUC)*, ISO Std. ISO/CEI 10 646 :2003(F). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO10646a1] *Technologies de l'information – Jeu universel de caractères codés sur plusieurs octets (JUC) – Amendement 1 : Glagolitique, copte, géorgien et autres caractères*, ISO Std. ISO/CEI 10 646 :2003/Amd.1 :2005(F). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO10646a2] *Technologies de l'information – Jeu universel de caractères codés sur plusieurs octets (JUC) – Amendement 2 : N'Ko, phags-pa, phénicien et autres caractères*, ISO Std. ISOCEI 10 646 :2003/Amd.2 :2006(F). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO10646a3] *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Amendment 3 : Lepcha, Ol Chiki, Saurashtra, Vai and other characters*, ISO Std. ISO/IEC 10 646 :2003/Amd.3 :2008(E). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO10646a4] *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Amendment 4 : Cham, Game Tiles, and other characters*, ISO Std. ISO/IEC 10 646 :2003/Amd.4 :2008(E). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ISO10646a5] *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Amendment 5 : Tai Tham, Tai Viet, Avestan, Egyptian Hieroglyphs, CJK Unified Ideographs Extension C, and other characters*, ISO Std. ISO/IEC 10 646 :2003/Amd.5 :2008(E). [Online]. Available : http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[ITU] International telegraph alphabet no. 2. ITU. [Online]. Available : http://www.itu.int/rec/T-REC-S.1-199303-I/en

[Jen04] T. Jennings. (2004) An annotated history of some character codes. [Online]. Available : http://wps.com/projects/codes/index.html

[Jona] D. W. Jones. Punched cards. [Online]. Available : http://www.cs.uiowa.edu/~jones/cards/

[Jonb] ——. Punched cards codes. [Online]. Available : http://www.cs.uiowa.edu/~jones/cards/codes.html

[Kat] R. H. Katz. Napoleon's secret weapon. [Online]. Available : http://bnrg.eecs.berkeley.edu/~randy/Courses/CS39C.S97/optical/optical.html

[Mac80] C. E. Mackenzie, *Coded Character Sets, History and Development*, ser. The systems programming series. Addison-Wesley, 1980.

[Mü] D. Müller. Code de Popham. [Online]. Available : http://www.apprendre-en-ligne.net/crypto/codes/popham.html

[Nor] North american data communications museum. North American Data Communications Museum. [Online]. Available : http://www.nadcomm.com/fiveunit/fiveunits.htm

[rab] Technical information. [Online]. Available : http://rabbit.eng.miami.edu/info/index.html

[RFC20] V. Cerf. (1969) ASCII format for network interchange. RFC 20. IETF. Probably a copy of X3.4-1968. [Online]. Available : http://www.ietf.org/rfc/rfc20.txt

[RFC821] J. B. Postel. (1982) Simple mail transfer protocol. RFC 821. IETF. [Online]. Available : http://www.ietf.org/rfc/rfc821.txt

[RFC1345] K. Simonsen. (1992) Character mnemonics and character sets. RFC 1345. IETF. [Online]. Available : http://www.ietf.org/rfc/rfc1345.txt

[RFC1521] N. Borenstein and N. Freed. (1993) Mime (multipurpose internet mail extensions) part one. RFC 1521. IETF. [Online]. Available : http://www.ietf.org/rfc/rfc1521.txt

[RFC2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R. Atkinson, M. Crispin, and P. Svanberg. (1996) The report of the IAB character set workshop. RFC 2130. IETF. [Online]. Available : http://www.ietf.org/rfc/rfc2130.txt

[Sav08] J. Savard. (2008) Re : Vertical bar, broken bar and ASCII code 124. Usenet message on alt.folklore.computers. [Online]. Available : news:a6738911-857a-46a0-a874-f2a9229415f6@d36g2000prf.googlegroups.com

[Sea] S. J. Searle. A brief history of character codes. [Online]. Available : http://tronweb.super-nova.co.jp/characcodehist.html

[Uni] The unicode consortium home page. Unicode Consortium. [Online]. Available : http://www.unicode.org

[UTR17] K. Whistler, M. Davis, and A. Freytag, "Unicode character encoding model," Unicode Consortium, Tech. Rep. 17, 2008. [Online]. Available : http://www.unicode.org/reports/tr17

[Wika] Polybius square. Wikipedia, The Free Encyclopedia. [Online]. Available : http://en.wikipedia.org/w/index.php?title=Polybius_square&oldid=261174254

[Wikb] Sémaphore (communication). Wikipédia, l'encyclopédie libre. [Online]. Available : http://fr.wikipedia.org/w/index.php?title=S%C3%A9maphore_(communication)&oldid=35629796

[Win] D. T. Winter. Codes. [Online]. Available : http://homepages.cwi.nl/~dik/english/codes