Représentation d'un texte en machine

ASCII, ISO-8859-1, Unicode

Caractère	Décimal	Hexadécimal	Binaire
A	65	0x41	01000001
B	66	0x42	01000010
C	67	0x43	01000011
...	...	...	...
Z	90	0x5A	01011010

Présentation

Toute l'information est représentée dans un ordinateur par des nombres encodés sous forme binaire par des 0 et des 1. Se pose alors la question de la représentation des caractères, ne serait-ce que parce que la communication entre les utilisateurs et les ordinateurs s'opère essentiellement sous forme textuelle.

Le Principe de l'encodage

La solution est simple : on associe chaque caractère à un code binaire.

Caractère	Décimal	Hexadécimal	Binaire
A	65	0x41	01000001
B	66	0x42	01000010
C	67	0x43	01000011
...	...	...	...
Z	90	0x5A	01011010

Les « caractères » sur fond bleu sont les caractères non imprimables. Pour bien lire le tableau, il faut construire le code hexadécimal en prenant d'abord le digit de la ligne, puis le digit de la colonne. Par exemple, la lettre « n » a pour code hexadécimal 6E.

Table ASCII

L'*American Standard Code for Information Interchange*, de son petit nom le code ASCII ou Ascii (prononcez aski), est une norme informatique pour le codage des caractères. En adoptant le même codage, les systèmes informatiques conçus par n'importe quel fabricant savent ainsi échanger du texte, des nombres, des signes de ponctuation et bien d'autres symboles.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STH	ETH	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	CD2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	spc	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Les « caractères » sur fond bleu sont les caractères non imprimables. Pour bien lire le tableau, il faut construire le code hexadécimal en prenant d'abord le digit de la ligne, puis le digit de la colonne. Par exemple, la lettre « n » a pour code hexadécimal 6E.

Malgré sa large acceptation, avec ses **7 bits par caractère**, cette table avait pour principal défaut de ne pas prendre en compte les caractères qui n'existent pas dans la langue anglaise, ne serait-ce que les lettres accentuées.

Les premières évolutions

Des tables multiples, mutuellement incompatibles, ont alors émergé : une table pour les européens, une autre pour les Japonais et ainsi de suite.

Progressivement, notamment avec l'émergence du Web au cours des années 1990, l'augmentation de l'interconnexion des ordinateurs personnels a amené au début des années 2000 à la mise en place d'une énorme table intégrant le contenu de toutes les tables existantes, via le standard UTF.

Le standard UTF

Le standard Unicode UTF (Universal Character Set Transformation Format) s'est imposé pour l'échange, car il permet d'agréger sur 8 bits, 16 bits ou 32 bits par caractère la totalité des caractères utilisés dans toutes les langues humaines… et même extraterrestres, puisque le Klingon est également intégré.

Les caractères liés à l'édition des partitions de musique ou les émojis sont également intégrés.

UTF-8

Pour éviter de consommer 32 bits par caractère, des variantes plus compactes ont été mises à disposition.

La plus connue – des européens, puisqu'elle regroupe les caractères qui nous concernent – est la table UTF-8. Elle se concentre sur les premiers 8 bits de la table UTF complète. Par sa nature, UTF-8 est d'un usage très répandu sur internet et dans les systèmes échangeant de l'information. Il s'agit également du codage le plus utilisé dans les systèmes de logiciels libres pour gérer le plus simplement possible des textes et leurs traductions dans tous les systèmes d'écritures et alphabets du monde. Les navigateurs internet d'aujourd'hui utilisent le codage UTF-8 et les concepteurs de sites prenent en compte cette même norme ; c'est pourquoi il y a de moins en moins de problèmes de *compatibilité*. **Toutefois, toutes ces différentes normes et leurs incompatibilités** sont la cause des problèmes que l'on rencontre parfois avec les caractères accentués. Il est donc préférable pour la rédaction de courriels à l'étranger, de n'utiliser que des caractères non accentués.

De la difficultÃ© avÃ©rÃ©e du bon amÃ©nagement des caractÃ¨res dans l'encodage numÃ©rique

UTF-8 est donc un encodage des caractères basé sur UNICODE, de longueur variable qui utilise de 1 à 4 octets par symbole.

Pour éviter de consommer 32 bits par caractère, des variantes plus compactes ont été mises à disposition.

La plus connue – des européens, puisqu'elle regroupe les caractères qui nous concernent – est la table UTF-8. Elle se concentre sur les premiers 8 bits de la table UTF complète. Par sa nature, UTF-8 est d'un usage très répandu sur internet et dans les systèmes échangeant de l'information. Il s'agit également du codage le plus utilisé dans les systèmes de logiciels libres pour gérer le plus simplement possible des textes et leurs traductions dans tous les systèmes d'écritures et alphabets du monde. Les navigateurs internet d'aujourd'hui utilisent le codage UTF-8 et les concepteurs de sites prennent en compte cette même norme ; c'est pourquoi il y a de moins en moins de problèmes de *compatibilité*. Toutefois, toutes ces différentes normes et leurs incompatibilités sont la cause des problèmes que l'on rencontre parfois avec les caractères accentués. Il est donc préférable pour la rédaction de courriels à l'étranger, de n'utiliser que des caractères non accentués.

UTF-8 est donc un encodage des caractères basé sur UNICODE, de longueur variable qui utilise de 1 à 4 octets par symbole.

Utilisation du code ASCII en Python

En Python, il est très facile de travailler avec le code ASCII grâce aux fonctions intégrées.

1) Conversion d'un caractère en code ASCII

Pour obtenir le code ASCII d'un caractère, on utilise la fonction `ord()` *:*

```
# Exemple de conversion d'un caractère en ASCII
print(ord('A'))  # Affiche 65
print(ord('a'))  # Affiche 97
print(ord('0'))  # Affiche 48
```

2) Conversion du code ASCII en caractère :

Inversement, pour obtenir le caractère correspondant à un code ASCII, on utilise la fonction `chr()` *:*

```
# Exemple de conversion d'un code ASCII en caractère
print(chr(65))  # Affiche 'A'
print(chr(97))  # Affiche 'a'
print(chr(48))  # Affiche '0'
```