

今回は、文字データの表現方法について学ぶ。

1. 原理

文字をコンピュータで表現するためには、1年生で学んだ ASCII コード表のような表を用いる。ASCII コード表は英字、数字、記号しか含んでいなかったため、日本語や世界各国の文字を全て表すには不十分である。そこで、ASCII コード表の他に様々な文字コード表が用いられてきた。

次の手順で文字コード表を決める。

(a) 扱う文字の範囲を決める

文字コード表に入れる文字の範囲を決める必要がある。決めた範囲に含まれる文字の集合を**文字集合 (character set)**と呼ぶ。

(b) 文字に番号を付ける

文字集合に含まれる全ての文字に番号を付ける。番号付けされた文字の集合を**符号化文字集合**と呼ぶ。

(c) 番号をビット列に変換する方法を決める。

文字に付けた番号と実際のビット (バイト) 列との変換方法を決める。この変換方法のことを**文字符号化方式 (character encoding scheme)**と呼ぶ。よく使う文字が短いビット列になるように符号化したりする。複数の文字集合を同時に使用する時は、どの文字集合に属するか分かるように符号化方式を決める。

どのコンピュータでも同じ文字データが扱えるように、符号化文字集合、符号化方式は ISO¹、JIS² 等で規格化されている。しかし、国により、コンピュータの OS により、採用している規格が完全に同じではないため、どの規格か判定する際にミスがおこることがある。**文字化け**は、判定ミスが起きた結果である。

Web ブラウザが文字符号化方式の自動判定に成功した例と失敗した例を図 1 に示す。

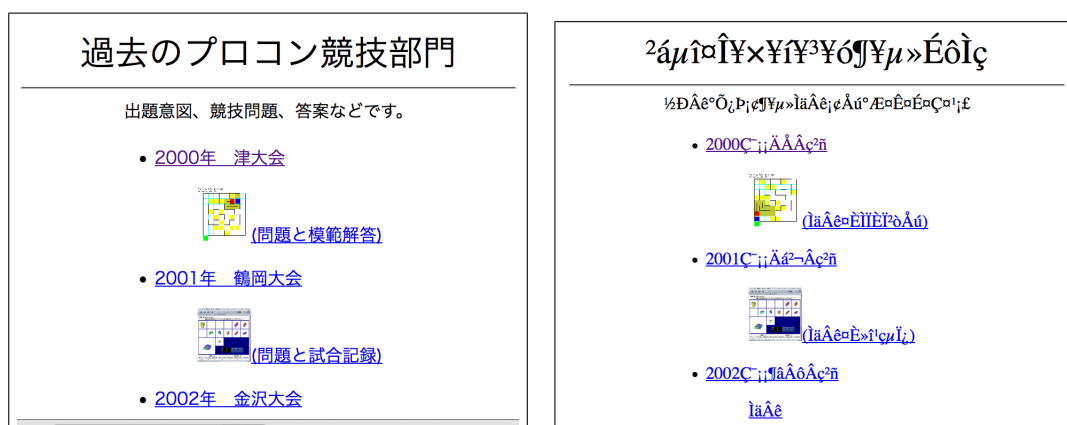


図 1: 文字化けの例

¹ISO : International Organization for Standardization(国際標準化機構)

²JIS : Japan Industrial Standards(日本工業規格)

2. 日本で使用される符号化文字集合

(a) ASCII(American Standard Code for Infomation Interchange) コード表

図2に示す文字コード表、1963年にアメリカ規格協会(ASCII)が定めた。7ビットで、制御文字、記号、数字、英字大文字、英字小文字を表現できる。制御文字と印刷可能文字あわせて128文字が含まれる。

		(上位3ビット)							
		0	1	2	3	4	5	6	7
0	NUL DLE			0	@	P	^	p	
1	SOH DC1	!	1	A	Q	a	q		
2	STX DC2	"	2	B	R	b	r		
3	ETX DC3	#	3	C	S	c	s		
4	EOT DC4	\$	4	D	T	d	t		
5	ENQ NAK	%	5	E	U	e	u		
6	ACK SYN	&	6	F	V	f	v		
7	BEL ETB	'	7	G	W	g	w		
8	BS CAN	(8	H	X	h	x		
9	HT EM)	9	I	Y	i	y		
A	LF SUB	*	:	J	Z	j	z		
B	VT ESC	+	;	K	[k	{		
C	FF FS	,	<	L	\	l			
D	CR GS	-	=	M]	m	}		
E	SO RS	.	>	N	^	n	~		
F	SI US	/	?	O	_	o	DEL		

(下位4ビット)

*TAB: Tabulator(表作成) **機種やソフトによる

	意味	C言語	キー入力
BEL(07H)	ベル(Bell)	'\007'	^G
BS(08H)	バックスペース(Back Space)	'\b'	^H [BackSpace]
HT(09H)	タブ(Horizontal TAB)*	'\t'	^I [Tab]
LF(0AH)	改行(Line Feed)	'\n'	^J [Enter]**
FF(0CH)	改ページ(Form Feed)	'\f'	^L
CR(0DH)	復帰(Carriage Return)	'\r'	^M [Enter]**
ESC(1BH)	エスケープ(Escape)	'\033'	[Esc]
(20H)	空白(White Space)	' '	[]

代表的な制御文字

ファイル	通信回線: CR LF
Windows : CR LF	プログラム中 : LF('\n')
UNIX : LF	
MacOS X : LF	

改行の表現方法

印刷可能文字

図2: ASCII 文字コード表

(b) JIS X 0201(通称 JIS 8 ビットコード)

図3に示す文字コード表、1969年に日本工業規格(JIS)に定められた。7ビットで、制御文字、記号、数字、英字大文字、英字小文字、カタカナを表現できる。制御文字と印刷可能文字あわせて191文字が含まれる。制御文字、記号、数字、英字大文字、英字小文字の範囲は、ASCIIと、ほぼ、同じになっている。

		(上位4ビット)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL DLE			0	@	P	^	p									
1	SOH DC1	!	1	A	Q	a	q										
2	STX DC2	"	2	B	R	b	r										
3	ETX DC3	#	3	C	S	c	s										
4	EOT DC4	\$	4	D	T	d	t										
5	ENQ NAK	%	5	E	U	e	u										
6	ACK SYN	&	6	F	V	f	v										
7	BEL ETB	'	7	G	W	g	w										
8	BS CAN	(8	H	X	h	x										
9	HT EM)	9	I	Y	i	y										
A	LF SUB	*	:	J	Z	j	z										
B	VT ESC	+	;	K	[k	{										
C	FF FS	,	<	L	\	l											
D	CR GS	-	=	M]	m	}										
E	SO RS	.	>	N	^	n	~										
F	SI US	/	?	O	_	o	DEL										

(下位4ビット)

はASCIIコード表と異なる部分

図3: JIS 8 ビット文字コード表

(c) JIS X 0208(通称 JIS 漢字)

図4に示す符号化文字のこと。1978年に日本工業規格(JIS)に定められた。16ビットで、記号、英数字、かな、漢字などを表現できる。**全角文字** 6879文字が登録されている。

(区：上位8ビット)

94区×94点＝最大8,836文字

	21	22	23	24	25	26	27	28	...	30	31	737E
21		◆		あ	ア	A	A	ー	亜	院	鵠
22	、	□		あ	ア	B	B	丨	唾	陰	鷺
23	。	■		い	イ	Γ	B	┐	娃	隠	鵠
24	、	△		い	イ	Δ	Γ	┐	阿	韻	鵠
25	、	▲		う	ウ	E	Δ	┐	哀	時	鵠
...
7D	◎								胤	往	龜
7E	◇	○							藤	応	龠

(点：下位8ビット)

JIS第一水準漢字：2,965文字
JIS第二水準漢字：3,390文字
記号など：524文字

図4: JIS 漢字コード表

(d) JIS X 0213(通称 JIS 拡張漢字)

図5に示す符号化文字集合(character set)のこと。2000年に日本工業規格(JIS)に定められた。その後、2004年、2012年に改正されている。コード表を2枚にすることで、より多くの文字を格納する。**全角文字** 11,223文字が登録されている。JIS X 0208を包含する上位規格(superset, 上位集合)である。

(区：上位8ビット)

	21	22	23	24	25	26	27	28	...	30	31	7E
21		◆	▷	あ	ア	A	A	ー	亜	院
22	、	□	▶	あ	ア	B	B	丨	唾	陰
23	。	■	◁	い	イ	Γ	B	┐	娃	隠
24	、	△	◀	い	イ	Δ	Γ	┐	阿	韻
25	、	▲	↗	う	ウ	E	Δ	┐	哀	時
...
7D	◎	♪	胤	往
7E	◇	○	藤	応

(点：下位8ビット)

第1面
JIS第一水準漢字：2,965文字
JIS第二水準漢字：3,390文字
記号など：524文字
JIS第三水準漢字：1,908文字
合計：8787文字

第2面
JIS第四水準漢字：2,436文字

図5: JIS 拡張漢字コード表

3. 日本で使用される文字符号化方式 (character encoding scheme)

文字データをファイルに格納したり通信路に送信したりスルとき、どんなバイト列に対応付けるか考える必要がある。

(a) ISO-2022-JP(通称 JIS コード)

電子メール等で用いられることが多い。エスケープシーケンスを用いて、使用する符号化文字集合 (文字コード表) を切替える (初期状態は ASCII)。7bit の範囲にエンコーディングされる。次にエスケープシーケンスの一部を示す。

ESC	(B	:	ASCII に切替える。	
ESC	(J	:	JIS X 0201 半角英数に切替える。	
ESC	\$	B	:	JIS X 0208 に切替える。	
ESC	\$	(Q	:	JIS X 0213 の第1面に切替える。
ESC	\$	(P	:	JIS X 0213 の第2面に切替える。

例: “A 亜 a ¥” を ISO-2022-JP にエンコーディングした状態

41H	1BH	24H	42H	30H	21H	1BH	28H	42H	61H	5CH	1BH	28H	4AH	5CH	1BH	28H	42H
'A'	ESC	'\$'	'B'	' '	'亜'	ESC	'('	'B'	'a'	'\'	ESC	'('	'J'	'¥'	ESC	'('	'B'
<JIS X0208 へ>				<ASCII へ>				<ローマ字へ>				<ASCII へ>					

(b) EUC-JP(Extended UNIX Code for Japanese, 日本語 EUC)

1980 年代に UNIX 上で日本語を扱うために考案された。現在では UTF-8 が普及し使用されることが少なくなっている。エスケープシーケンスを用いることなく、ASCII と JIS X 0208 を切替える。符号化文字集合として JIS X 0201 を指定できないので、半角の ¥ を表現することができない。JIS X 0208 の文字は JIS 漢字コードに 8080H を加えた値で表現する。

「ISO-2022-JP より短くエンコーディングできる」、「変換規則が単純」、「バイト列のどの部分が全角文字か直に分かる」、「漢字を構成するバイトが ASCII と重ならない」等のメリットがある。

例: “A 亜 a \” を EUC-JP にエンコーディングした状態

41H	BOH	A1H	61H	5CH	5CH
'A'	' '	'亜'	'a'	'\'	'\'

5CH は ¥ を表現する。

(c) Shift_JIS(Shift JIS code, SJIS)

1980 年代に PC で日本語を扱うために考案され現在も Windows で使用されている。エスケープシーケンスを用いることなく、JIS X 0201 と JIS X 0208 を切替える。符号化文字集合として ASCII を指定できないので、半角の ¥ を表現することができない³。JIS X 0208 の文字は、JIS 漢字コードを 8140H - 9FFCH と E040H - FCFCh に変換して表現する。計算で変換できるが若干複雑である。

「ISO-2022-JP より短くエンコーディングできる」、「半角カナを表現できる」等のメリットがある。

例: “A 亜 a ¥ ¥” を Shift_JIS にエンコーディングした状態

41H	88H	9FH	61H	5CH	5CH
'A'	' '	'亜'	'a'	'¥'	'¥'

5CH は ¥ を表現する。

³日本の Java 言語や C 言語の教科書で ¥n を ¥¥n と表記しているのは Windows 使用前提だからか？

4. Unicode(ユニコード)

符号化文字集合と文字符号化方式の両方を定めた文字コードの規格である。単一 (“Uni”) の巨大な文字コード表に、世界中の全ての文字を収めようとしている。符号化文字集合と文字符号化方式の両方を定めているので、符号化に関する混乱も少ない。

当初、中国・日本・韓国の漢字を統一 (CJK 統合漢字) することで、16bit で全ての文字にコードを割り振る計画だったが、世界中の文字は予想以上に多かったため途中で 21bit に変更された。

(a) 歴史

1980 年代後半から検討され 1990 年代前半に実用化された。現在も世界各地の文字、古い文字、新しい絵文字等が追加され続けている。

Unicode1.0.1	1992	16bit	28,359 文字	JIS X 0201,0208,0212 の文字収録
Unicode2.0.0	1996	21bit	38,950 文字	Unicode1.x.x との互換性を失う
Unicode3.0.0	2002	21bit	95,221 文字	JIS X 0213 へ対応
Unicode4.0.0	2003	21bit	96,447 文字	
Unicode5.0.0	2006	21bit	99,089 文字	楔文字や象形文字を追加
Unicode6.0.0	2012	21bit	109,449 文字	携帯絵文字追加
Unicode7.0.0	2014	21bit	113,021 文字	
Unicode8.0.0	2015	21bit	120,737 文字	
Unicode9.0.0	2016	21bit	128,172 文字	

(b) 符号化文字集合

図 6 に示すように現在の Unicode は、文字コード表を 17 面 (5bit)、256 区 (8bit)、256 点 (8bit) で構成している。この表に最大 111 万文字を収容可能である。半角 ASCII 文字から始まり、半角文字も全角文字もこの表に収録されている。

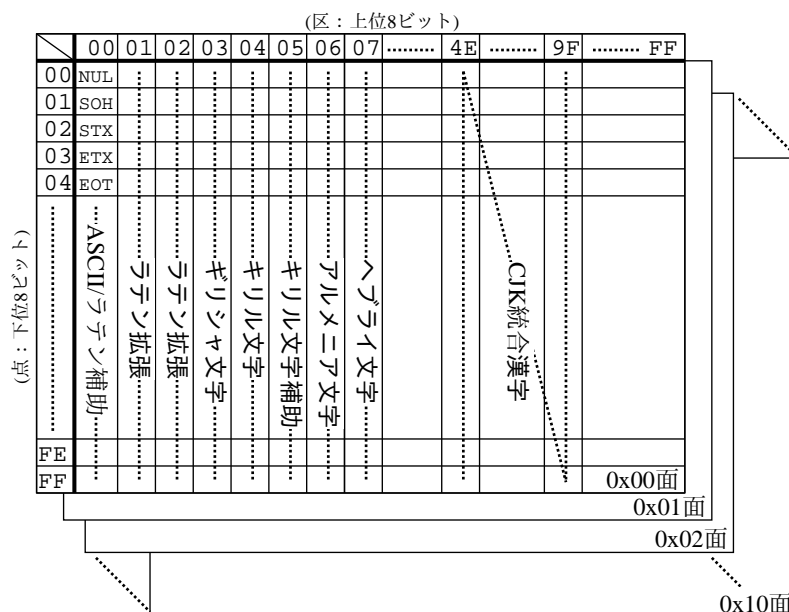


図 6: Unicode 一覧

表の1文字を指定するためには、21bit(5+8+8)のコード(Unicode スカラ値)を用いる。Unicode スカラ値は、"U+1234"のように先頭にU+を付加した16進数で表現する。

00面(U+0000~U+FFFF)はBMP(Basic Multilingual Plane, 基本多言語面)と呼ばれる。当初はBMPだけで全ての文字を格納する予定であった。U+0000~U+007FはASCIIコードと同じ配列になっている。Java言語のchar型が16bitなのも、16bitのUnicodeを納することを前提に設計されたためである。

なお、完全なコード一覧は、<https://ja.wikipedia.org/wiki/Unicode> で見ることができる。面白い文字が見つかるので、一度、見てほしい。

(c) 文字符号化方式

ユニコード用の文字符号化方式(UTF:Unicode Transformation Format)が定められている。多くの方式が定められているが、代表的な方式は以下の3つである。

UTF-8 : 半角英数字が小さく符号化できる。

最もよく使用される方式である。

UTF-16 : 00面を16bitに、その他は32bitに符号化する。

Unicodeが21bitになってから、Javaのchar型に使用されている。

UTF-32 : Unicodeの上位に0を付加し32bitに符号化する。

固定長なので扱いやすい。

(d) UTF-8による符号化手順

21bitのUnicodeのコードポイントを2進表記したものを、表中のy、xに格納する。yの部分には最低1つの1のビットがあるはずである。符号化の結果は1~4バイトになる。

Unicode	UTF-8(ビット列で表記)	バイト数
U+0000~U+007F	0xxxxxxx	1
U+0080~U+07FF	110yyyyx 10xxxxxx	2
U+0800~U+FFFF	1110yyyy 10yxxxxx 10xxxxxx	3
U+10000~U+1FFFFF	11110yyy 10yyxxxx 10xxxxxx 10xxxxxx	4