ユニコード（Unicode）

tbasic.org [*1]

[2013 年 10 月版]

これから，コンピュータを使う上で，日本語に限らず，外国語も使うことが多くなってくるかもしれません。また，一文書に多国語が混在することもあるでしょう。そのようなことは，ユニコードを使うことで容易に可能となります。そこで，ここではその，ユニコードについて簡単にまとめます。

1　ユニコードとは

現在，私たちはコンピューターでワープロ等を使用する場合，入力の問題を別にすれば，日本語を特に意識しないで使うことができます。しかし実は日本語に限らず，アルファベットや数字などもコンピューターで使うためには，コンピューター内部ではいくつかの仕組みが必要です。ただ日本で販売されているコンピューターは日本語を使えるような仕様で作られていますから，日本語だけを使用している場合は，そのことを特に意識する必要はありません。

しかし，いくつかの言語を同時に使う場合は，それらの仕組みを意識する必要があります。例えば，中国語やハングルなどを使った文書を取り扱う場合，それに対応した処理を考えることになります。元々は，それぞれの言語に従って，それぞれの方式があり，それぞれを切り替えながら利用することになります。しかしそのように利用するにしても，各ソフトウエアがそれに対応する仕組みを持たなければそれを利用することはできません。例えば，Tiny Basic for Windows でも，Ver. 1.17 までは，

Print "你好"

と入力して実行しても，画面には

?好

と表示されます。これは Tiny Basic for Windows が日本語対応ソフトで，日本語の漢字でない "?" の文字を正しく扱えないことに依ります。

現在世界は国境を越えての情報交換が頻繁に行われています。これから益々複数の国の文字を扱う必要性が高まってきています。この問題に対応するために世界中の文字を同時に扱う仕組みが求められていました。これの一つの解決策がユニコードでした。ユニコードは世界中の文字を一つの仕組みの中で表す試みです。この方式を利用すれは一つのプログラムの中に色々な言語が混在して利用できるようになります。

例えば，ユニコード対応になった Tiny Basic for Windows では，

Print "你好! 안녕하십니까 세계！Hello World こんにちは "

と入力して実行すると，画面には

你好! 안녕하십니까 세계！Hello World こんにちは

[*1] http://www.tbasic.org

と正しく表示されます。これは Tiny Basic for Windows がユニコードに対応していることに依ります。
まとめると，

- コンピューターで文字を扱うには対応する仕組みが必要
- 元々は言語ごとその仕組みが考えられていた。
- それらを統一して利用できる方法として考えられたのがユニコード

と言えるでしょう。ここでは日本語とユニコードについて簡単にまとめます。

ユニコードについての詳細で正確な理解には，ユニコード規格の本家

The Unicode Consortium
http://www.unicode.org/

を参照を薦めます。

2 符号化

2.1 符号化文字集合

コンピューターで文字を使用する場合，その文字集合を確定する必要があります。また文字集合を決めたとき，その集合に含まれる各々の文字を対応させる方法を決める必要もあります。このように，文字とその対応方法を決めた集合を **符号化文字集合**と言います。

日本語での符号化文字集合としては，JIS X 0201 や JIS X 0208（JIS 基本漢字），JIS X 0213（JIS 拡張漢字）などがあります。現在の PC の最新 OS ではこの JIS X 0213 がサポートされています。

ユニコードの符号化文字集合は世界中の国々の符号化文字集合を集めたものと言えますが，その対応方法（符号化，番号付け）は元々の各国のものと異なります。現在の最新のユニコード規格では，JIS X 0213 の文字を含んでいますが，文字の対応方法はユニコードでのものが採用されており，JIS の方法とは異なります。まとめると，

(1) 日本語の符号化文字集合は JIS X 0213 が最新。
(2) ユニコードでも，その文字を使うことができる。

となります。

2.2 日本語の文字符号化方式

符号化文字集合に対して，実際のコンピューターでそれらの文字を表示する方法は一通りではありません。日本語の場合，JIS 符号化，シフト JIS 符号化，EUC 符号化の 3 方法があります。JIS X 0213 でもそれぞれの符号化方式について規定しています。符号化方式は，コンピューターでの具体的な取扱い方法ですので，コンピューター内部のみでなく，ファイル形式も規定します。ですから，例えば，同じ文字でも，JIS 符号化で表現されたファイルと，シフト JIS 符号化で表現されたファイルは，実際にフィルに記録される内容が異なります。ですから，日本語を正しく取り扱うためには，それぞれその符号化法に対応した処理ができなければなりません。そうでないと正しく読み取ることや，処理することもできません。文字化けの現象は符号化方式の不

対応から起きます。

まとめると，日本語の文字符号化方式は

JIS，シフト JIS，EUC の 3 種類

となります。

3 ユニコード

3.1 ユニコード

ユニコードは世界中の文字を一つの体系で表そうとしていますから，ユニコードの符号化文字集合は世界中の文字を含んでいます。その結果，かなり大きなものです。欧米で使われているラテン文字はもちろん，日本語の他，ギリシア語，ロシア語，アラビア語，中国語，ハングルなどが含まれています。1991 年にバージョン 1.0.0 が発表された時は，7161 個の文字が登録されていましたが，その後バージョンが上がるにつれて，文字が増え，2006 年にバージョン 5.0 になった時は，99089 文字が登録されました。2013 年 10 月現在，バージョン 6.3 で，110,122 個の文字が登録されています。携帯で使われている絵文字も登録されています。

ユニコードの符号化文字集合は，16 進数で番号付けされています。文字はその番号に U+ を付け加えて表すことになっています。例えば，672C 番目の文字は，U+672C と表します。

ユニコードは最初は 16 ビットで番号付けされました。16 ビットは 4 桁の 16 進数ですから，U+0000 から U+FFFF と表されます。例えば，"日"という文字は，26085 番目の文字として登録されています。これを 16 進数として表すと，65E5 ですから，U+65E5 となります。

日本語漢字は CJK というグループとしてまとめられています。CJK は中国，日本，韓国を意味していますので，このグループには，日本語以外の漢字も含まれています。

例えば，U+4E00 から CJK 漢字が登録されていて，

U+4E00　一
U+4E01　丁
U+4E02　丂
U+4E03　七
U+4E04　丄
・・・

となっています。上にある文字は日本語にもある文字ですが，例えば，U+4E66 は 书 という文字で日本語にはありません。

ところで，16 ビットで表される数は，$2^{16} = 65536$ です。上の表示法では，

U+0000　～　U+FFFF

と表されますが，これで世界中のすべての文字を表すには明らかに不足です。そこで，1996 年にユニコードがバージョン 2.0 になった時，この文字が拡張され，U+20B9F のような 16 進 5 桁の文字番号も登録されています。U+0000～U+FFFF を基本領域（基本多言語面），U+10000 以上を拡張領域と言われています。現在，そ

の上限は U+10FFFF とされていて，最大 1,114,112 個の文字が登録できます。

まとめると
ユニコードは

U+0000, ・・・, U+10FFFF

のように番号付けされた多国語文字の集まり。

U+0000 ～ U+FFFF　　を基本領域,
U+10000 ～ U+10FFFF　　を拡張領域　と言う。

となります。

3.2　ユニコードの文字符号化方式

ユニコードの文字符号化方式は 3 種類あります。

UTF-32，UTF-16，UTF-8

です。（ここで UTF は Unicode Transformation Format を意味します。）それぞれ，UTF-32 は，32 ビットを使ってユニコード文字を表し，UTF-16 は，16 ビットを使い，UTF-8 は 8 ビットを使います。U+10000 以上の文字は，16 ビットでは表せませんから，UTF-16 では，2 組の 16 ビットを使って一文字を表します。ですから，UTF-16 では，一文字を表すに必要なメモリーは固定されていません。16 ビット 1 個の場合と，16 ビット 2 個の場合があります。UFT-8 は，8 ビット 1 個から 4 個を使って表します。

ユニコードはもともと，16 ビットで世界中の文字を表す試みとして始まり，後にその不足を補うために，それ以上のビットを使うことになりました。ですから，16 ビットの領域に実は殆んど普通に使う文字は含まれています。ですから，UTF-16 で表した場合殆んどの文字が 16 ビット 1 個で表されます。逆に，UTF-32 で表した場合，殆んど 32 ビットの半分 16 ビットで間に合うのに，32 ビット 1 個を使うことになります。UFT-32 が文字の取扱いは便利で簡単ですが，使用するメモリー領域がほぼ倍必要となります。

これに対して，UTF-16 は，多少の面倒な部分はありますが，比較的効率的なメモリー使用になります。

このことから，現在のコンピューターでは内部的には UTF-16 を使う OS もあります。実際，Windows NT 系列では UTF-16 が昔から使われていました。現在普通に使われている Windows 系の OS は，Windows Server，XP，Vista，Windows 7, 8 ですがこれらはすべて NT 系です。

UTF-8 は一文字を表すのに 1 ～ 4 個の 8 ビットを使う煩雑さはありますが，ASCII との上位互換もあり，ネット上ではよく使われています。また，最近の Linux では，UTF-8 を標準符号化方式に採用しているものもあります。

まとめると　ユニコードの文字符号化方式は主として

- UTF-32，UTF-16，UTF-8 の 3 種類。
- 現在の Windows では内部的には UTF-16 が使われている。
- 外部ファイルの形式としては，UTF-8 が多く使われている。

となります。

3.3 結合文字

ユニコードは世界中の文字を表すことを目的としていますので，結合文字という特別な文字の扱いがあります。例えば，

が

と言う字は，「か」に濁点「゛」という風に言います。つまり「が」は「か」に「゛」で合成されると考えられます。「か」は U+304B ですし，実は，「゛」は U+3099 というユニコード文字です。これらを合わせると，「が」が合成されます。つまりこの場合，

が ＝ か ＋ ゛

となります。ここで，「か」を基底文字，「゛」を結合文字，今の場合「が」を合成済み文字といいます。日本語に関連した結合文字は，「゛」や「゜」などがあります。欧米語では，「´」や「^」などがあります。

これはこれで良いのですが，関連して少し困った状況が存在します。実は，例えば，上の「が」は結合文字を使って 2 文字で表すことになりますが，この文字の作り方とは別に「が」という文字がユニコードに登録されています。U+304C は一文字「が」を表します。このように結合文字を使って合成された文字の多くが，一文字として別に登録されています。つまりそれらは，文字としての表し方が，ユニコードとして 2 種類あることになります。文字列を比較して処理をするときは，基本的にはユニコードのコード番号で処理しますから，この状況では正しく処理されないことになります。例えば「が」を検索する場合，U+304B と U+3099 として検索すれば，U+304C で表されている「が」は見つからないことになります。

この状況に対応するため，ユニコードでは，正規化（normalize）という概念があります。これは，文字列を表す標準的な形式を定めておき，その形に変換してから処理をするというものです。この正規化の形式もまず次の 2 種があります。

- D 型正規化形式 (Normalization Form D (NFD))
- C 型正規化形式 (Normalization Form C (NFC))

D 型正規化形式は，結合文字などで表されるものを，結合文字を使って分解します。C 型正規化形式はその文字を結合された等価な文字に合成します。例えば

- 元の文章　　：「に」「っ」「ほ」「゜」「ん」「じ」「ん」
- D 型正規化形式：「に」「っ」「ほ」「゜」「ん」「し」「゛」「ん」
- C 型正規化形式：「に」「っ」「ぽ」「ん」「じ」「ん」

となります。この他の正規化として

- KD 型正規化形式 (Normalization Form KD (NFKD))
- KC 型正規化形式 (Normalization Form KC (NFKC))

というものもあります。これは，等価の文字ではなくで互換文字についての正規化です。互換とは，実質同じ意味の文字という意味で，例えば，日本語では

「１」（全角文字１）と「1」（半角文字１）

などが対応します。まとめると，

- ユニコードには文字によっては分解や合成がある。
- その為の結合文字がある。（濁点やアクサンなど・・・）
- 文字列の標準形として正規化という概念がある。

となります。

3.4 フォント

ユニコードは文字を表す一つの仕組みですが，実際にコンピューターの画面に文字を表示するためには，対応する文字の形が必要です。それがフォントです。コンピューターで文字を表示する場合，文字コードを利用するフォントに対応させます。当然ですが同じ文字でもフォントが違えば，文字の形は微妙に異なります。例えば，「日本」という文字も

明朝：　日本

ゴシック：　日本

のように使うフォントによって形が異なります。上にフォントは日本語フォントですが例えば中国語の漢字を表すには日本語のフォントでは表すことができません。例えば

図

という文字は日本の文字にはありませんので，対応するフォントが無ければ表示はできません。この文字を表示するには，中国語漢字用フォントが必要になります。ワープロなどでは，フォントの判定をソフト側で行い必要なフォントがコンピューターにインストールされているれば，それを利用して表示します。しかし，エディターなどではそれに対応していないのが普通ですので，ユーザー側でとくに指定しなければ

□

と表示されてしまいます。まとめると，

- ユニコードで対応する文字があってもフォントが正しく設定されていなければ表示されない。
- ワープロの場合，フォントの切り替えを自動的に行うので，対応するフォントがインストールされていれば，表示できることが多い。
- エディターなどでは，自動変換は行われないのが普通。

となります。

4　まとめ

現在の私たちは，一つの国に閉じて生活をしている訳ではなく，世界中の多くの国々と交流をして生活をしています。そしてこの傾向は進むことはあれ，後退することはないでしょう。ですから，色々な国々について，各国々の特徴を尊重しながら，それらを統一的な方法で扱うことができれば大変便利で効率的でしょう。コンピューターでの文字の取扱いについてのその試みの一つがユニコードです。

歴史的な状況と，各国々の事情により，ユニコードは種々の問題を含んでいますが，このような試みは歴史の進化に沿ったものであることは確かです。また現在のコンピューターの OS の中にもその仕組みが既に随分以前から取り入れられています。

最近，それが OS だけではなく，応用ソフトも対応が進みつつあります。このような中で，コンピューター利用者にとっても，ユニコードの理解は避けて通れない事柄でしょう。まとめると，

- ユニコードはコンピューター進化の方向に沿った試み。
- 一般のコンピューター利用者もこれからは，ユニコードの理解を避けて通れない。

となります。