L2/04-082

A User's Guide to the Unihan Database

This is a preliminary draft of a document to replace the header within the Unihan.txt file that I think should be turned into a Unicode Standard Annex or other appropriate document type. The document still needs a lot of work; this is mostly to let the UTC get a sense of what I'm aiming towards. As such, it contains some information which is aimed more at the UTC as an audience than the general public; such information will be edited appropriately before it's publicly released.

甲 *Introduction*

The Unihan database is the repository for the collective Unicode Consortium's collective knowledge regarding the CJK Unified Ideographs[1] contained in the Unicode Standard. It contains mapping data to allow conversion to and from other coded character sets and additional information to help implement support for the various languages which use the Han ideographic script.

Formally, ideographs are defined within the Unicode Standard via their mappings. That is, the Unicode Standard does not formally define what the ideograph U+4E00 *is*; rather, it defines it as being the equivalent of, say, 0x523B in GB 2312, 0x14421 in CNS 11643, 0x306C in JIS X 0208, and so on.

In practice, implementation of ideographs requires large amounts of ancillary data. Input methods require information such as pronunciations, as do collation algorithms. Data in character sets not included in the world of international standards bodies needs to be converted. Relationships between ideographs need to be defined to allow for fuzzy string matching. And, in line with Becker's Law that for every expert there is an equal and opposite expert, it's important to track not only what properties a given ideograph has, but who claims it has those properties.

Unlike characters in Western scripts such as Latin and Greek, whose basic property is their sound, which stays largely constant across languages, the basic property for Han ideographs is their *meaning*. This isn't to say that ideographs are truly ideographic, in that represent abstract ideas. Most ideographs are divided into a *radical*, which gives a vague sense of

1. We use without apology the term *ideograph*, despite its being a notorious misnomer. Sinologists are not in complete agreement as to what term would be better; *ideograph* has the singular advantage of being equally repugnant to all parties.

meaning, and a *phonetic*, which gives a vague sense of pronunciation.[2] The Unihan database therefore includes structural analyses and definitions for ideographs.

The Unihan database is available to the public in two forms: one, as a text file, Unihan.txt, which is distributed as part of the Unicode Standard; and two, via the World Wide Web on the Unicode Web site. The text file is truly massive, twenty-five megabytes of data with over one million lines, covering over 70,000 ideographs.

This document is a guide to that data, describing the mechanics of the Unihan database, the nature of its contents, and the status of the various fields. The Unihan database is truly a work in progress, with new data (and even new fields) being added on a regular basis.

If the Unihan database has a weakness, however, it is that it is maintained by volunteers. Nobody is paid to work on it. There is a great deal of useful information which would be added if only someone would provide it. Despite this, the Unihan database provides solid, useful data for everyday implementation needs and beyond.

乙 *Mechanics*

The Unihan database is currently maintained using MySQL running on the server, `kwanyin.unicode.org`. There are no public accounts for the database as yet. When Unicode 4.0.1 is stabilized, we'll create a clone of the working database which can also be stored on `kwanyin.unicode.org` to which public access will be granted. We will then shift the Unihan Web pages over to using MySQL instead of grep to get data on characters.

The database is backed up daily by running a shell script under `cron`. The script uses `mysqldump` to create a copy on a second machine and invokes a Perl script to generate a new Unihan.txt file (without the header). The resulting Unihan.txt file is archived weekly on a third machine, and further archives are made on an irregular basis when its content changes significantly. For us to lose more than a week's worth of changes would therefore require a failure of all three machines. Moreover, all three are located where Unicode officers have physical access to them, providing further control.

The database has a single table, Unihan, which has a large number of columns. All but four are included in the public Unihan.txt file. The four exceptions are:

`kNotes`. As the name implies, this is a place for us to store any notes we want to keep track of about a character. For example, if we depart from our sources for a reading for a given character, we can store information on the rationale behind that decision in the `kNotes` field.

---

2. There are exceptions. Some ideographs (such as 上 and 三) are abstractions, others (such as 馬 and 井) originated historically as pictograms. Nor do all ideographs have any real semantic content; some are purely phonetic, used mainly to transcribe foreign sounds.

There is no intention of ever making this public.

kCheungBauer. This is an index into Cheung Kwan-hin and Robert S. Bauer, *The Representation of Cantonese with Chinese Characters*, Journal of Chinese Linguistics, Monograph Series Number 18, 2002, and is described more fully in the section on Dicitonaries below. I would like the UTC's permission to make this data public when the field is fully populated. (There will be roughly 1,000 to 1,500 entries.)

kDefaultSortKey. This is a 32-bit integer which provides a default radical-stroke ordering for the characters in the database. The 32 bits are actually used as a bitfield as follows:

Bits 0-16 are a representation of the character's code point:

> U+4E00 through U+9FFF are mapped to 0x00000 through 0x051FF; that is, 0x4E00 is subtracted from the Unicode Scalar Value.
>
> U+3400 through U+4DBF are mapped to 0x05200 through 0x06BBF; that is, 0x1E00 is added to the Unicode Scalar Value.
>
> U+20000 through U+2A6DF are mapped to 0x06C00 through 0x112DF; that is, 0x19400 is subtracted from the Unicode Scalar Value.
>
> U+F900 through U+FAFF are mapped to 0x1F600 through 0x1F7FF; that is, 0xFD00 is added to the Unicode Scalar Value.
>
> U+2F800 through U+2FA1F are mapped to 0x1F800 through 0x1FA1F; that is, 0x10000 is subtracted from the Unicode Scalar Value.
>
> The net result of these remappings is to reorder the blocks (main CJK Unified Ideographs, Extension A, Extension B, Compatibility Ideographs, Compatibility Extension), and to leave a gap of over 58,000 code points between the end of Extension B and the first Compatibility Ideographs block, and over 12,000 after the Compatibility Ideographs Extension.

Bits 17-22 are the character's residual stroke count (0 through 63). The residual stroke count taken is from the first value in the character's kRSUnicode field.

Bits 23-30 are the character's KangXi radical number used (1 through 214). The radical number used is that of the first value in the character's kRSUnicode field. The difference between simplified and traditional radical is ignored.

Note that bit 31 is unused, so it makes no difference whether the sort key is treated as signed or unsigned.

The kDefaultSortKey field thus defines a consistent way of ordering all the characters in Unihan, first by radical-stroke, then by Unicode block (with the compatibility blocks coming last), and finally by code point. It is not the most efficient sorting key possible, but it has the advantage of being easily generated and does not require existing keys to be regenerated when

new ideographs or compatibility ideographs are added to the standard.

The UTC may want to consider making the kDefaultSortKey available in the public database.

Finally, there is the UTF8 field, which is (as one might expect) the character's UTF-8 encoding. This field is useful for UTF-8-savvy MySQL clients in that it allows them to display a glyph for the character without having to be customized to understand how to use the Unicode Scalar Value. There is no intention to make this field public.

Each character in Unihan occupies one row in the database, with the Unicode Scalar Value as the primary key.

The remaining eighty-plus columns constitute the public contents of the database. They are discussed below.

All data in the Unihan database is stored in UTF-8. MySQL's UTF-8 support is still spotty, but for our purposes we don't need extensive support. We just need to be able to store and acess UTF-8 data. We rely on clients to handle everything else.

There are two forms of the Unihan database made available to the public. One is the on-line Unihan database on the Unicode Web site, <http://www.unicode.org/charts/unihan.html>. The other is the Unihan.txt file. The on-line database is updated when a new version of Unicode is made official (although we need to be a bit more consistent about this). The Unihan.txt file is also made available in beta versions of Unicode releases.

The on-line version of the database currently uses a copy of the Unihan.txt file; it uses the grep command to get data out of the file. This is really inefficient and is using up a lot of the processor power on the Web server. We plan to rewrite the CGI scripts which constitute the on-line version of the database to use a second, read-only MySQL database.

The released version of the Unihan.txt file consists of a header followed by data. Unix line breaks are used. Each line in the data section is one entry with three, tab-separated fields: the Unicode Scalar Value, the database field tag, and the value for the database field for the given Unicode Scalar Value. For most of the fields, if multiple values are possible, the values are separated by spaces. No character may have more than one instance of a given field associated with it, and no empty fields are included in the Unihan.txt file.

The tags consist entirely of ASCII letters and digits with no spaces or other puncutation except for underscore. For historical reasons, they all start with a lower-case "k."

There is no formal limit on the lengths of any of the field values. Any Unicode characters may be used in the field values except for double quotes and control characters (especially tab, newline, and carriage return). Most fields have more a more restricted syntax, such as the kKangXi field which consists of multiple, space-separated entries, with each entry consisting

of four digits 0 through 9, followed by a period, followed by three more digits.

The data lines are sorted by Unicode Scalar Value and field-type as primary and secondary keys, respectively.

丙 *Field Types*

The data in the Unihan database serves a multitude of purposes, and the fields are most conveniently grouped into categories according to the purpose they fulfil. We provide here a general discussion of the various categories, followed by a detailed description of the individual fields, alphabetically arranged.

Again, it is important to remember that all data in the Unihan database has been donated to the Unicode Consortium. Unicode currently has no staff with the responsibility to maintain or update the Unihan database. This means that, for example, the data is more complete for Chinese than for other languages simply because more data has been donated for Chinese for than other languages.

IRG Mappings

The only normative parts of the Unihan database, and the most exhaustively checked fields, are the eight IRG source fields: `kIRG_GSource` (mainland China and Singapore), `kIRG_HSource` (Hong Kong SAR), `kIRG_JSource` (Japan), `kIRG_KPSource` (North Korea), `kIRG_KSource` (South Korea), `kIRG_TSource` (Taiwan), `kIRG_USource` (Unicode/USA), and `kIRG_VSource` (Vietnam).

These represent the official mappings between Unihan and the various encoded character sets or collections which have been submitted by IRG members. The versions of these standards may differ from the published versions generally available, particularly for PRC standards. This is because in the early days of Unicode, the PRC would occasionally add characters to their standards on an ad hoc basis in order to make sure they were included. The various procedures involved in submitting characters to the IRG for consideration no longer make this necessary.

At the moment, the U-source consists only of the Unicode Standard itself, and the field value is always equal to the character's Unicode Scalar Value. This will change when Extension C1 is formally encoded, because Extension C1 contains a small number of characters submitted by the Unicode Technical Committee which use a different indexing system.

Note that we do not include the four IRG dictionary fields in this category, largely because they are not normative parts of the standard.

Other Mappings

There are twenty-four fields in this category. They consist of mapping tables between the ideographic portions of Unicode and those of other encoded character sets or character

collections. Some of the character sets covered mirror official IRG sources. For example, we have data for mapping GB 12345, which is a part of the IRG's G-source. The difference between the two is that the kGB1 field maps all of GB 12345 to Unicode, and not just that portion included in the G-source, and it doesn't map any of the informal extension to GB 12345.

Dictionary Indices

There are three main reasons for providing indices into standard dictionaries.

One, standard dictionaries provide a "paper trail" for fields such as the English gloss (kDefinition) and the various pronunciations or readings, as well as variant data.

Two, standard dictionaries provide a reference for scholars or students who wish more information about a character.

Third, standard dictionaries are a source for unencoded characters. This is particularly important for Cantonese, where the Cantonese lexicon is not standardized and has been neglected by the authors and architects of previous character set encodings other than HK SCS.

As elsewhere, the set of dictionaries covered represent data that has been volunteered. There are important dictionaries (e.g., the *Hanyu Da Cidian*, the *Shuowen*) for which formal indices should be provided. And as elsewhere, the data which has been volunteered is weighted heavily in favor of Chinese.

Four of the dictionary fields represent official IRG indices for the dictionaries used in the four-dictionary sorting algorithm. Two (kIRGHanyuDaZidian and kIRGKangXi) are still being used by the IRG, but the other two (kIRGDaeJaweon and kIRGDaiKanwaZiten) are not. We have, nonetheless, retained their data for reference purposes.

For all four, there are clone fields to hold Unicode indices into the same four dictionaries. By and large, the data in the IRG fields and their Unicode counterparts is the same—but not always.

There are also two obsolete dictionary fields, kAlternateKangXi and kAlternateMorohashi.

The remaining dictionaries can be grouped into three categories: general-purpose Chinese (including classical Chinese and Mandarin), Cantonese, and other.

The general-purpose Chinese dictionary fields are: kCihaiT, kGSR, kKarlgren, kMatthews, and kSBGY. These represent large, standard Chinese-Chinese, Chinese-English dictionaries, or definitive sinological studies.

The Cantonese dictionary fields are kCheungBauer, kCowles, kLau, and kMeyerWempe. All but Cheung-Bauer are large character-based Cantonese-English dictionaries.

At present, the only other dictionary field is `kNelson`, the character's index in the first edition of Andrew N. Nelson's excellent and popular *Modern Reader's Japanese-English Character Dictionary*.

In selecting dictionaries for inclusion—outside of the general consideration of who is willing to volunteer what data—we aim for including large dictionaries rather than small ones, and standard dictionaries such as serious students might have on their shelves.

Readings

We include in this category the pronunciations for a given character in Mandarin, Cantonese, Tang-dynasty Chinese, Japanese, Sino-Japanese, Korean, and Vietnamese. We also include here the English gloss for a given character.

Any attempt at providing a reading or set of readings for a character is bound to be fraught with difficulty, because the readings will vary over time and from place to place, even within a language. Mandarin is the official language of both the PRC and Taiwan (with some differences between the two) and is the primary language over much of northern and central China, with vast differences from place to place. Even Cantonese, the modern language covered by the Unihan database with the least range, is spoken throughout Guangdong Province and in much of neighboring Guangxi, and covers two large urban centers (Guangzhou and Hong Kong), with Guangzhou Cantonese somewhat infected by Mandarin and Hong Kong Cantonese more than a little infected by English.

Indeed, even the same speaker will pronounce the same word differently depending on the *social* context. For example, in Cantonese, the *-ing* and *-eng* finals are fairly interchangeable, with the former preferred in more formal settings, and the latter having a distinct colloquial feel.

Add to this the fact that in none of these languages—the various forms of Chinese, Japanese, Korean, Vietnamese—is the *syllable* the fundamental unit of the language. As in the West, it's the *word*, and the pronunciation of a character is tied to the word of which it is a part. In Chinese (followed by Vietnamese and Korean), the rule is one ideograph/one syllable, with most words written using multiple ideographs. In most cases, the word has only one reading (or only one important reading), but there are numerous exceptions.

In Japanese, the situation is enormously more complex. Japanese has two pronunciation systems, one derived from Chinese (the *on* pronunciation, or Sino-Japanese), and the other from Japanese (the *kun* pronunciation). The *kun* pronunciation for a single kanji can easily be polysyllablic (e.g., *ichi* for 一). In essence, the *on* pronunciation is the Japanese way of pronouncing the Chinese word, whereas the *kun* pronunciation is the Japanese translation of the word.

As if that weren't bad enough, some characters have rare pronunciations known only to a

minority of even native speakers, or are so rare themselves that most native speakers don't know how to pronounce them (e.g., U+40DF 磼, used in a Hong Kong place name). In many cases, the pronunciations given by professional lexicographers are little more than educated guesses.

The net effect here is a confession: Unlike mappings between Unicode and other character sets, providing *definitive* data on pronunciations or, similarly, providing a *definitive* English gloss is impossible, and not something we've achieved. While we make every effort to use our sources judiciously, we are aware of the fact that this data can always be improved and extended. Users should not naïvely assume that learning to pronounce an East Asian language is all about learning to pronounce the individual ideographs, or that reading is done by parsing the ideographs, one at a time.

Despite this, the reading and definition data is very useful both for the student attempting to learn these languages, and for the professional attempting to use them, and so the data is included in the Unihan database.

Dictionary-like Data

This category is something of a hodge-podge, consisting of various fields including information one might find in a dictionary (such as a characters *cangjie* input code), or data useful in determining levels of support (such as frequency), or structural analyses which can be helpful in lookup systems (such as the characters' phonetic).

As with the readings and English gloss, this data does not cover as much of Unihan as is theoretically possible, although it does cover the bulk of what is used day-to-day.

The fields included in this category are `kCangjie`, `kFenn`, `kFrequency`, `kGradeLevel`, `kHanyuPinlu`, `kHKGlyph`, and `kPhonetic`. Note that in the case of `kFenn` and `kHanyuPinlu`, the data is named for the dictionary from which the data is derived, not for the type of data it is.

Radical-Stroke Counts

We include five radical-stroke counts for Unihan, although only two are actively used at the moment. Three of the four are based on IRG standard dictionaries: the *Hanyu Da Zidian*, which uses a slightly different radical system from the others, is not included.

All the radical-stroke fields are based on the radical-system introduced by the 17th century *KangXi* dictionary. Each ideograph is assigned one of 214 radicals. In most cases, the radical assigned is the natural radical, giving a clue as to the character's meaning; in the rest, the radical is arbitrary, based on the character's structure. One also counts the character's residual strokes, that is, the number of brush strokes required to write everything in the character except the radical.

To find a character using the radical-stroke system, one determines its radical and the number of residual strokes, then looks through the list of characters with those characteristics. This is

a clumsy system compared to alphabetical lookup, but is one of the most widespread systems throughout East Asia.

Unfortunately, it's also ambiguous.

First of all, if a character does not have a natural radical, it can sometimes be hard to tell what the radical ought to be (e.g., 井 being assigned the radical 二). Even if the character naturally falls into radical-like pieces, it can be hard to tell which is the radical and which the phonetic (e.g., 和, which looks like it belongs to the radical 禾, actually belongs to the radical 口). Moreover, since Unicode encodes characters, not glyphs, two different glyphs for the same character may have different residual strokes (such as 者, which can be written either with or without a dot, altering its stroke count between nine and eight, respectively).

We include multiple radical-stroke systems to allow for this. Three of the radical-stroke fields represent the character's radical-stroke count as determined by its position within a standard IRG dictionary. The remaining two (`kRSJapanese` and `kRSUnicode`) are intended to cover a "typical" Japanese radical-stroke count, and everything else, respectively.

The primary use for the `kRSUnicode` field is to cover the form of the character as drawn in the Unicode Standard. However, it is also used for cases where there is sufficient ambiguity that a reasonable person might look for a character in multiple places, particularly where one of our source dictionaries categorizes a character under a different radical or with a different stroke count.

The `kRSUnicode` field also uses an apostrophe after the radical number to indicate that the character uses a standard simplification. In simplified Chinese, many radicals have standard, simplified forms, such as 讠, which is the simplified form of the radical 言.

There is, by the way, no standard way of ordering characters *within* a given radical-stroke group. Unicode's radical-stroke charts order characters with the same radical-stroke count by the Unicode block in which they occur. If looking for a character with radical 64 (手) and ten residual strokes, one knows that of the 173 candidates in Unicode 4.0.1, the most common ones come towards the head of the list and the less common ones later.

The IRG is in the process of adopting a common system of assigning the first stroke of the phonetic element to one of five categories, and sorting by those categories. When this "first-stroke" data is available for all of Unihan, it will be added to the Unihan database.

Variants

Although Unicode encodes characters and not glyphs, the line between the two can sometimes be hard to draw, particularly in East Asia. There, thousands of years worth of writing have produced thousands of pairs which can be used more-or-less interchangeably.

To deal with this situation, the Unicode Standard has adopted a three-dimensional model for determining the relationship between ideographs, and has formal rules for when two forms

may be unified. Both are described in some detail in the Unicode Standard. Briefly, however, the three-dimensional model uses the *x*-axis to represent meaning, and the *y*-axis to represent abstract shape. The *z*-axis is used for stylistic variations.

To illustrate, 說 and 貓 have different positions along the *x*-axis, since they mean two entirely different things (*to speak* and *cat*, respectively). 貓 and 猫 mean the same thing and are pronounced the same way but have different abstract shapes, so they have the same position on the *x*-axis (semantics) but different positions on the *y*-axis (abstract shape). They are said to be *y*-variants of one another. On the other hand, 說 and 説 have the same meaning and pronunciation and the *same* abstract shape, and so have the same positions on both the *x*- and *y*-axes but different positions on the *z*-axis. They are *z*-variants of one another.

Ideally, there would be no pairs of *z*-variants in the Unicode Standard; however, the need to provide for round-trip compatibility with earlier standards, and some out-and-out mistakes along the way mean that there are some. These are marked using the `kZVariant` field.

The Unihan database also includes the `kCompatibilityVariant` field, which marks compatibility variants as defined by the Unicode Standard.

The remaining variant fields are used to mark different types of *y*-variation.

The `kSimplifiedVariant` and `kTraditionalVariant` fields are used to aid in the process of going between simplified and traditional Chinese. The People's Republic of China, beginning in the 1950's, undertook a series of language reforms aimed at boosting literacy by making Chinese easier to read and write, largely by reducing the number of strokes needed to write a number of characters. These reforms have also been adopted in Singapore. The traditional forms, however, are predominant in Taiwan, overseas Chinese communities, and even in China's two Special Administrative Regions, Hong Kong and Macao.

The mapping between simplified and traditional Chinese is really quite complex. In many cases, the official simplification is an acceptable alternative even within traditional Chinese, as with our two cats above: both 猫 and 貓 are acceptable in traditional Chinese, but only 猫 is used in simplified Chinese. In a few cases, a single simplified form corresponds to multiple traditional forms, such as 台, which is not only a traditional character in its own right, but also the simplification for 檯, 臺, and 颱. And a character-by-character conversion isn't sufficient to convert between simplified and traditional Chinese because of lexical differences. Mandarin, for example, is called 普通话 (the Common Tongue) in Mainland China, and 國語 (the National Language) in Taiwan.

The remaining two variation fields, `kSemanticVariant` and `kSpecializedSemanticVariant`, are used to mark cases where two characters have identical and overlapping meanings, respectively.

Thus U+514E 兎 and U+5154 兔 are *y*-variants of one another; both mean *rabbit*. U+4E3C 丼 and U+4E95 井 are not pure *y*-variants of one another. 井 means *a well*, and although 丼 can also

mean *a well* and be used for 井, it can also mean *a bowl of food*. We use kSemanticVariant, then, for the former pair, and kSpecializedSemanticVariant for the latter.

Numeric Values

Finally, we have three fields, kAccountingNumeric, kOtherNumeric, and kPrimaryNumeric to indicate the numerical values an ideograph may have. Traditionally, ideographs were used both for numbers and words, and so many ideographs have (or can have) numeric values. The various kinds of numeric values are specified by these three fields.

丁 *The Fields*

We now give an alphabetical listing of the eighty-plus fields included in the Unihan database.

For each field we give the following information: its tag, its Unicode status, its category as defined above, its level of completion, its separator, its syntax, and its description.

The tag is the tag used in the Unihan.txt file to mark instances of this field.

The Unicode status is either *normative*, *informative*, or blank, depending on whether it is a normative part of the standard, an informative part of the standard, or neither. We also include *deprecated* as a Unicode status if the field is no longer to be used.

We use three values for the level of completion.

A *complete* field is one which could not cannot contain more data. The kMatthews field is complete, for example, because there are only so many characters in Mathews' dictionary, and all of them which are encoded have their indices in the Unihan database.

We also include as complete fields such as kKangXi which are Unicode counterparts to IRG data fields which are truly complete.

An *extendable* field is one which is complete as far as it goes. Fields such as the readings and definition fields are extendable, because they could theoretically be extended to cover all of Unihan, but they are sufficiently complete for most needs and we are unlikely to need to go back and revise existing data in the field.

An *incomplete* field is one which has known gaps and needs more data before it can be truly useable.

Fields which allow multiple values have a separator defined, usually a space. Fields which do not need or cannot have a separator do not have this defined, such as the IRG source fields.

The syntax is a Perl-like regular expression describing the formal structure of an individual entry in the field. The syntax for the kKangXi field is "[0-9]{4}\.[0-9]{2}[01]", which means four

decimal digits followed by a period, followed by two more decimal digits, followed by a zero or a one. The syntax can be used to validate the contents of a field. *(Note: We may have to do more than just regexps for some fields. We'll see when we get to them.)* Of course, just complying with the formal syntax is no guarantee that the data is correct: a `kKangXi` value of 9999.990 is syntactically correct but wrong anyway, since there is no page 9999 in the *KangXi* dictionary.

Finally, the description contains not only a description of what the field contains, but also source information, known limitations, methodology used in deriving the data, and so on.

Tag	kAccountingNumeric
Status	Informative
Category	Numeric Value
Level of Completion	Complete
Separator	None
Syntax	[0-9]+
Description	The value of the character when used in the writing of accounting numerals. Accounting numerals are used in East Asia to prevent fraud. Because a number like ten (十) is easily turned into one thousand (千) with a stroke of a brush, monetary documents will often use an accounting form of the numeral ten (such as 拾) in their place.

Tag	kAlternateKangXi
Status	Deprecated
Category	Dictionary Indices
Level of Completion	Complete
Separator	space
Syntax	[0-9]{4}\.[0-9]{3}

Description	An alternate possible position for the character in the KangXi dictionary.
Tag	kAlternateMorohashi
Status	Deprecated
Category	Dictionary Indices
Level of Completion	Complete
Separator	space
Syntax	[0-9]{5}\'?
Description	An alternate possible position for the character in the Morohashi dictionary
Tag	kBigFive
Status	Informative
Category	Other Mappings
Level of Completion	Complete
Separator	N/A
Syntax	[0-9A-F]{4}
Description	The Big Five mapping for this character in hex; note that this does *not* cover any of the Big Five extensions in common use, including the ETEN extensions.
Tag	kCCCII

Status	Informative
Category	Other Mappings
Level of Completion	Complete
Separator	N/A
Syntax	[0-9A-F]{6}
Description	The CCCII mapping for this character in hex

Tag	kCNS1986
Status	Informative
Category	Other Mappings
Level of Completion	Complete
Separator	N/A
Syntax	[12E]-[0-9A-F]{4}
Description	The CNS 11643-1986 mapping for this character in hex

Tag	kCNS1992
Status	Informative
Category	Other Mappings
Level of Completion	Complete
Separator	N/A

Syntax	[123]-[0-9A-F]{4}
Description	The CNS 11643-1992 mapping for this character in hex

Tag	kCangjie
Status	Informative
Category	Dictionary-like Data
Level of Completion	Complete
Separator	N/A
Syntax	[A-Z]+
Description	The *cangjie* input code for the character. This incorporates data from the file `cangjie-table.b5` by Christian Wittern.

Tag	kCantonese
Status	Informative
Category	Readings
Level of Completion	Extendable
Separator	space
Syntax	[A-Z]+[1-6]

Description

The Cantonese pronunciation(s) for this character
The romanization used is a modified version of the Yale romanization, modified as follows:

(1) No effort is made to distinguish between Yale's "high level" and "high falling" tones, which are not universally reflected in all Cantonese romanizations and which appear to be no longer distinctive in Hong Kong Cantonese. As a general rule, syllables which end with a stop (p, t, or k) have the "high level" tone; but there are numerous exceptions.

(2) Digits 1-6 are used to indicate the tones --

1 == High level/high falling

2 == High rising

3 == Middle level

4 == Low falling

5 == Low rising

6 == Low level

(3) Accordingly, the letter "H" is *not* used as a tone indicator
Cantonese pronunciations are sorted alphabetically, not in order of frequency

N.B., the Hong Kong dialect of Cantonese is in the process of dropping initial NG- before non-null finals. Any word with an initial NG- may actually be pronounced without it, depending on the speaker and circumstances. Many words with a null initial may similarly be pronounced with an initial NG-. Similarly, many speakers use an initial L- for words previously pronounced with an initial N-.

Cantonese data are derived from the following sources:

Cheung Kwan-hin and Robert S. Bauer, *The Representation of Cantonese with Chinese Characters*, *Journal of Chinese Linguistics* Monograph Series Number 18, 2002 (kCheungBauer)

Roy T. Cowles, *A Pocket Dictionary of Cantonese*, Hong Kong: University Press, 1999 (kCowles)

Sidney Lau, *A Practical Cantonese-English Dictionary*, Hong Kong: Government Printer, 1977 (kLau)

Bernard F. Meyer and Theodore F. Wempe, Student's *Cantonese-English Dictionary*, Maryknoll, New York: Catholic Foreign Mission Society of America, 1947 (kMeyerWempe)

饒秉才, ed. 廣州音字典, Hong Kong: Joint Publishing (H.K.) Co., Ltd., 1989

中華新字典, Hong Kong:中華書局, 1987

黃港生, ed. 商務新詞典, Hong Kong: The Commercial Press, 1991

朗文初級中文詞典, Hong Kong: Longman, 2001 (kGradeLevel)

Tag	kCihaiT
Status	Informative
Category	Dictionary Indices
Level of Completion	Complete
Separator	N/A
Syntax	[0-9]{3}\.[0-9]{3}
Description	The position of this character in the Cihai (辭海) dictionary, single volume edition, published in Hong Kong by the Zhonghua Bookstore, 1983 (reprint of the 1947 edition), ISBN 962-231-005-2. The position is indicated by a decimal number. The digits to the left of the decimal are the page number. The first digit after the decimal is the row on the page, and the remaining two digits after the decimal are the position on the row.