L2/20-XXX

Informal FAQ: Mongolian script

To: Unicode Technical Committee
From: Liang Hai / 梁海 <lianghai@gmail.com>
Date: 29 July 2020

This document is intended to be a combination of:

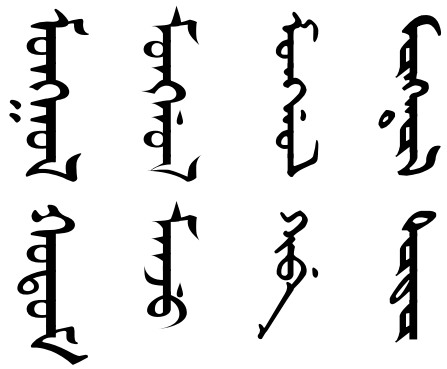
- an FAQ for the general public;
- a summary of our progress so far, so interested parties can more easily get on board to participate;
- and an introduction of the resources hosted in the authors personal GitHub repo.

General

Q1: What writing systems are covered by the encoded Mongolian script?

Three groups of four major writing systems:

- *Hudum* (more commonly referred to as the *traditional Mongolian script*) for the Mongolian language in general
- *Manchu* and *Sibe* for the Manchu–Sibe language group
- *Todo* for the Oirat–Kalmyk Mongolian language group



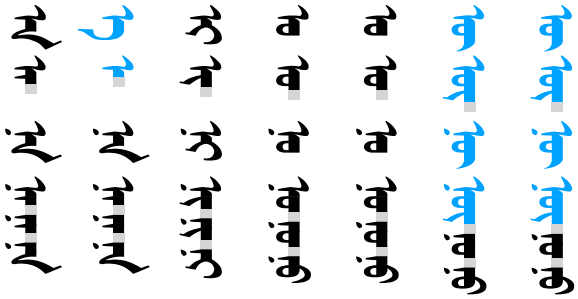
Hudum, Manchu, and Todo writing systems all have their own *Ali Gali* extensions for transcribing Sanskrit and Tibetan texts. Many Ali Gali characters were encoded but they are even more problematic because of a lack of thorough investigation when Mongolian was originally proposed. How early forms of Hudum, Manchu, and Todo should be represented is not clear at all.

The appended Table 1 shows the de facto usage of each character in the four major writing systems. Table 13-4 on page 537 of the *Core Specification* (version 13.0) is essentially a simplified version of Table 1.

Q2: What's wrong with the current encoding?

The current Unicode encoding of the Mongolian script is largely unusable because of multiple levels of critical issues, including:

- Despite common belief, the current Mongolian encoding is *not* merely problematic on the font or rendering level. Instead, it is fundamentally unsuitable for general text exchange, because is was designed to encode underlying phonetic letters (*a e i o u ö ü* ...) that do not have a predictable relation to the written shapes in texts.



- Many phonetic letters are fully or contextually confusable with other letters, such as *o* and *u* (fully indistinguishable in their written forms), or *a* and *e* (distinguishable only in certain contexts, such as the beginning of a word). This over-differentiation means digital texts are unreliable in general exchange, because one cannot assume all the differentiated cases (*a/e, é/w, i/y, o/u/ö/ü/w, h/g, t/d,* ...) are intentionally and correctly made by users when the characters are originally inputted or later edited. Also, in many cases, it is simply controversial to determine a written word's underlying phonetic letters because of orthographical and dialectal disagreements.
- On the other hand, many phonetic letters have more written forms (variants) than what can be predicted in a simple cursive joining model (which is used by the Arabic script and is well-understood), however the essential contextual rules for determining each variant have never been standardized. None of the existing fonts are built with coherent rules. This has caused great incompatibility between fonts, and further worsened the problem that texts are inconsistently encoded.
- Also, NNBSP (U+202F NARROW NO-BREAK SPACE, with property Script = Common), is architecturally unreliable for serving the Mongolian encoding as a critical format control character.

Some documents introducing the issues:

- L2/17-328, *Script Ad Hoc Group Recommendations on Mongolian Text Model*
- L2/18-106, *Current problems in the Mongolian encoding*

The slide deck of a talk by the author at the IUC #42 is also relevant:

- *The Mongolian script: What's going on?!*

Q3: What's the current goal and progress?

The current goal is to improve the existing phonetic encoding model with comprehensive and coherent recommendations for text representation and shaping, based on recognition of the existing model's architectural defects.

The author has been drafting the recommendations in the form of a Unicode Technical Note (UTN), which may then be standardized to certain extent once it is stabilized and receives a consensus:

- L2/19-368, Draft technical note: *Text representation and shaping specification of the Mongolian script*

This draft UTN has not received enough feedback from the community. The author plans to transform the document into a tutorial, with abstract analyses appended, in order to improve readability and attract a greater readership.

In order to understand how the drafted UTN can converge with the EAC (Ethnic Affairs Committee of the Inner Mongolia Autonomous Region)'s published industry standard, the author has also been reviewing the latter.

Note that an earlier proposed migration to a grapheme-based (instead of phonetic letter based) new encoding model (see L2/18-102, *An improved graphetic model for the Mongolian encoding*) did not receive support from interested parties (see L2/18-108, *Mongolian Working Group Meeting 2 Report*).

Q4: Where can I find additional documents and resources?

A topical document list has been maintained to host all the documents relevant to the Mongolian encoding issues, including documents submitted to the UTC document register and the ones submitted to the Mongolian Working Group meetings:

- Topical Document List: Mongolian

The author maintains a personal GitHub repo to host some additional resources.

Q5: How can I contribute to this effort?

Read the documents (especially the draft UTN), understand the existing arguments, and get involved by submitting your feedback or proposal.

Fonts and shaping

Q6: Why have the variants been removed from the names list?

The variant information previously available in the names list is seriously misleading to implementors, as it seems to suggest how Mongolian texts should be encoded and how Mongolian fonts should be built, except it doesn't. Unicode Technical Report #54,

Unicode Mongolian 12.1 Snapshot, has been created to document the rationale behind and capture the last state before the removal.

Q7: Why does NNBSP often fail to trigger the correct shapes of particles (disconnected suffixes)?

When NNBSP (U+202F NARROW NO-BREAK SPACE) is displayed by a font different from the particle's, the change of font causes the required contextual shaping between NNBSP and the particle to fail, because of the OpenType Layout (OTL)'s architectural limit.

NNBSP is often displayed by a different font because it's supported by many Latin fonts for other purposes, and Latin fonts are generally high in the prioritized list of fonts in environments that do not offer font control to users, such as in browsers. The situation is more controllable in word processors such as Word because users can manually format the whole text with a Mongolian font.

Q8: Is NNBSP really supposed to have 1/3 the width of an ordinary space?

Not really.

In a font that does want to have NNBSP narrower than an ordinary word space, as long as it's clearly narrower and still wide enough for clearly separating a particle from the preceding word, it's good.

The widespread 1/3 width idea is a myth caused by misunderstanding of some original description's context.

Table 1. Unification and usage of characters

Notes: "UC" stands for *The Users' Convention*, and the decimal numbers are are how each code point is referred to in the UC. Red cells are specified in the UC but not actually used by the writing system in ordinary texts. The yellow cell is an omission in the UC.

Unicode character	*UC*	*Hudum*	*Todo*	*Sibe*	*Manchu*
U+1820 … A	32	a	a	a	a
U+1821 … E	33	e		~~[UC e]~~	~~[UC e]~~
U+1822 … I	34	i			
U+1823 … O	35	o		o	o
U+1824 … U	36	u			
U+1825 … OE	37	ö			
U+1826 … UE	38	ü			
U+1827 … EE	39	e’			
U+1828 … NA	40	n	n	n	n
U+1829 … ANG	41	ŋ			ŋ
U+182A … BA	42	b		b	b
U+182B … PA	43	p			
U+182C … QA	44	q(a) k(e)			
U+182D … GA	45	ɣ(a) g(e)			
U+182E … MA	46	m		m	m
U+182F … LA	47	l	l	l	l
U+1830 … SA	48	s	s	s	s
U+1831 … SHA	49	sh	sh		
U+1832 … TA	50	t			
U+1833 … DA	51	d			
U+1834 … CHA	52	ch	z	ch	ch
U+1835 … JA	53	j		~~[UC j]~~	j
U+1836 … YA	54	y		y	y
U+1837 … RA	55	r	r	r	~~[UC r]~~
U+1838 … WA	56	w	f	w	w
U+1839 … FA	57	f			
U+183A … KA	58	k’	~~[UC k(a)]~~	k(a)	k(a)
U+183B … KHA	59	k’	~~[UC k(e)]~~		
U+183C … TSA	60	ts			
U+183D … ZA	61	dz			
U+183E … HAA	62	h			
U+183F … ZRA	63	ř			
U+1840 … LHA	64	lh	lh		
U+1841 … ZHI	65	ǰ			
U+1842 … CHI	66	č			
U+1843 … TODO LONG …	67		<long>		
U+1844 … TODO E	68		e		
U+1845 … TODO I	69		i		
U+1846 … TODO O	70		o		
U+1847 … TODO U	71		u		
U+1848 … TODO OE	72		ö		
U+1849 … TODO UE	73		ü		
U+184A … TODO ANG	74		ŋ		
U+184B … TODO BA	75		b		
U+184C … TODO PA	76		p		

Unicode character	*UC*	*Hudum*	*Todo*	*Sibe*	*Manchu*
U+184D … TODO QA	77		x(a) k(e)		
U+184E … TODO GA	78		ɣ(a) g(e)		
U+184F … TODO MA	79		m		
U+1850 … TODO TA	80		t		
U+1851 … TODO DA	81		d		
U+1852 … TODO CHA	82		ch		
U+1853 … TODO JA	83		j		
U+1854 … TODO TSA	84		ts		
U+1855 … TODO YA	85		y		
U+1856 … TODO WA	86		w		
U+1857 … TODO KA	87		k(a)		
U+1858 … TODO GAA	88		g(a)		
U+1859 … TODO HAA	89		h		
U+185A … TODO JIA	90		j’		
U+185B … TODO NIA	91		[Sanskrit ñ]		
U+185C … TODO DZA	92		dz		
U+185D … SIBE E	93			e	e
U+185E … SIBE I	94			i	
U+185F … SIBE IY	95			ɪ	~~[UC none]~~ ɪ
U+1860 … SIBE UE	96			u	u
U+1861 … SIBE U	97			ū	ū
U+1862 … SIBE ANG	98			ŋ	
U+1863 … SIBE KA	99			q(a) k(e)	
U+1864 … SIBE GA	100			ɢ(a) g(e)	ɢ(a) g(e)
U+1865 … SIBE HA	101			χ(a) x(e)	χ(a) x(e)
U+1866 … SIBE PA	102			p	p
U+1867 … SIBE SHA	103			sh	sh
U+1868 … SIBE TA	104			t	t
U+1869 … SIBE DA	105			d	d
U+186A … SIBE JA	106			j	
U+186B … SIBE FA	107			f	
U+186C … SIBE GAA	108			g(a)	g(a)
U+186D … SIBE HAA	109			x(a)	x(a)
U+186E … SIBE TSA	110			ts	ts
U+186F … SIBE ZA	111			dz	dz
U+1870 … SIBE RAA	112			ř	ř
U+1871 … SIBE CHA	113			č	č
U+1872 … SIBE ZHA	114			ǰ	
U+1873 … MANCHU I	115				i
U+1874 … MANCHU KA	116				q(a) k(e)
U+1875 … MANCHU RA	117				r
U+1876 … MANCHU FA	118				f
U+1877 … MANCHU ZHA	119				ǰ

* EOF *