Canonical combining class for nukta characters

Norbert Lindenberg
Draft, 2022-10-17

Proposal

This document proposes to end the practice of assigning canonical combining class 7, also known as "Nukta" or "NK", to newly encoded nukta characters, and instead to determine the ccc values for nuktas in the same way as for other combining marks of the same script. For most Brahmic scripts, this means ccc=0; for non-Brahmic scripts likely positional values such as 220 or 230.

The Proposal to encode KAWI SIGN NUKTA already reflects this proposal and uses ccc=0 for the new nukta. There don't appear to be characters with ccc=7 among those already accepted for encoding.

Discussion

The UTC has traditionally assigned ccc=7 to nuktas, in the same way as it assigned ccc=9 to viramas. However, while viramas in their various forms as visible marks, control characters, or in-betweens are easily identified, it's not always clear at the time a character is encoded whether it should be considered a nukta. Once encoded, ccc values can not be changed because of the Unicode Normalization Stability policy. This has led to differences between the set of characters with ccc=7 and the set of characters with Indic syllabic category Nukta:

Code point	Character name	ccc	InSC
0AFD	GUJARATI SIGN THREE-DOT NUKTA ABOVE	0	**Nukta**
0AFE	GUJARATI SIGN CIRCLE NUKTA ABOVE	0	**Nukta**
0AFF	GUJARATI SIGN TWO-CIRCLE NUKTA ABOVE	0	**Nukta**
0F39	TIBETAN MARK TSA -PHRU	216	**Nukta**
10A38	KHAROSHTHI SIGN BAR ABOVE	230	**Nukta**
10A39	KHAROSHTHI SIGN CAUDA	1	**Nukta**
10A3A	KHAROSHTHI SIGN DOT BELOW	220	**Nukta**
1037	MYANMAR SIGN DOT BELOW	**7**	Tone_Mark
1E94A	ADLAM NUKTA	**7**	Other

These differences mean that clients can't rely on ccc=7 to find nuktas.

The assignment of ccc=7 for nuktas and ccc=9 for viramas also means that a character sequence where a nukta immediately follows a virama is canonical equivalent to the character sequence where that nukta immediately precedes that virama. The current documentation of the OpenType Universal shaping engine does not account for this equivalence. Of the major OpenType implementations only HarfBuzz is known to normalize text to be rendered, while others may insert dotted circles into virama-nukta sequences despite their canonical equivalence to corresponding and supported nukta-virama sequences.