L2/26-070

Make sequences with multiple variation selectors in a row formally non-conformant

Asmus Freytag & Mark Davis & Markus Scherer et al.
Unicode properties & algorithms group
2026-03-16

Recommended UTC actions

1. **Consensus**: Adopt formal core spec definitions for Variation Sequence, Variation Base and Variation Selector, Assigned Variation Sequence, and subsidiary definitions including Misplaced Default Ignorable Code Point, and provide updated guidance on security-relevant abuse of default ignorable characters in general, but Variation Selectors in particular; following the proposal in L2/26-070. For Unicode 18.0. See REPORTREF.
2. **Action Item** for Ken Whistler, Edcom: Add formal core spec definitions for Variation Sequence, Variation Base and Variation Selector, Assigned Variation Sequence, and subsidiary definitions including Misplaced Default Ignorable Code Point: Update the core spec text in 23.4, Variation Selectors to reflect the new definitions, any new behavior for “misplaced” DI, and to correct errors; following the proposal in L2/26-070. For Unicode 18.0. See REPORTREF.

PAG input

From discussion at UTC #186 and Edcom:

Faced with the fact that external specifications are proposing to repurpose Variation Selectors (VS) for unrelated purposes of invisibly adding metadata to “plain-text” data streams while relying on the “default ignorable” properties of the VS, the UTC decided on the need to clarify acceptable uses for VS.

Currently, Chapter 23 defines a variation sequence as a two-character sequence, and definition D1 in Chapter 3 states that the description of variation sequences in Chapter 23 is considered normative, but we do not have a conformance clause/definition in Chapters 3 or 23 that makes conformance requirements for variation sequences unambiguously clear and that suggests treatment for degenerate cases of multiple VS in a row or for a VS without base character.

Proposal

Add new text to the following sections in the Unicode Standard. Note that ***only*** the wording of proposed *definitions* or the *intent* of *conformance relevant statements* are presented for UTC ratification. All other text is suggested as input to the Editorial Working Group which has as its task to come up with final wording. Proposed text is shown as indented below.

Section 3.6 Combination

We are lacking an actual *definition* of ***Variation Sequence***. We currently just have a comment following D56, as well as text in chapter 23 that is definitional, but not actual definitions. Therefore, this proposal suggests to add a formal definition of *Variation Sequences* and its prerequisite and dependent definitions such as "Variation Base" or "Variation Selector" as a new subsection "Variation Sequences" between the existing subsections 3.6.1 *Combining Character Sequences* and 3.6.2 *Grapheme Clusters*.

> ***3.6.x Variation Sequences***

Start with an actual definition for Variation Selector, followed by a definition of Variation Base, because it's different from the base of a Combining Sequence.

> D57xx: *Variation Selector (VS):* A character with the property Variation_Selector.
>
> - A variation selector is also a default-ignorable combining character with combining class 0.
>
> D57xx: *Variation Base*: A Graphic Character (gc=L, M, N, P, S, Zs) that satisfies the following four constraints:
>
> 1. Not a Variation Selector
> 2. Canonical Combining Class = 0 (does not reorder)
> 3. NFD_QC=Yes (does not decompose)
> 4. NFC_QC=Yes (does not get consumed in composition)
>
> - Note the differences in the definitions of Variation Base and Base Character.

Next we need a formal definition of Variation Sequence:

> D57xx: *Variation Sequence (VSeq)*: A two-character sequence consisting of an initial *Variation Base* character followed by a *Variation Selector*.

Currently, the third note (bullet) under D56 starts with a sentence that refers to chapter 23 for the definition of a variation sequence. That sentence can be deleted and the remainder of the note can be placed below the new definition, like this:

> Because any variation selector is a combining character, a variation sequence is either a combining character sequence, or it is a subsequence of a longer combining character sequence. For example, the sequence <0030, FE00, 20E3> represents a variant of the digit zero, followed by an enclosing keycap. A variation sequence can also be a non-initial subsequence within a combining character sequence. For example, the sequence <1000, FE00, 1031, FE00> is a single combining character sequence with two variation sequences representing variants of the base character MYANMAR LETTER KA and the combining mark MYANMAR VOWEL SIGN E.> See Section 23.4, Variation Selectors.

(This includes a small wording improvement, and a terminology fix: “combining mark sequence” → “combining character sequence”)

While we are at it, we need to capture the definitions for three types of Variation Sequences here.

> Variation sequences are not intended as a general extension mechanism for the character encoding. The combination of a particular initial character plus a particular variation selector has no effect on display unless it occurs in pre-defined lists maintained by the Unicode Consortium.
>
> D57xx: *Assigned Variation Sequence:* A Variation Sequence that is explicitly listed with its intended effect on displaying the Variation Base character. Interpretation of a character sequence by a conformant implementation as a VSeq affecting the display of the Variation Base Character is limited to assigned VSeqs.
>
> The three types of *Assigned Variation Sequences* are *standardized*, *emoji*, and *ideographic*, as set forth below.:
>
> D57xx: *Standardized Variation Sequence:* A Variation Sequence that is listed in the file StandardizedVariants.txt in the Unicode Character Database.
>
> - The intended effect of a standardized variation sequence is described in the file and representative glyphs may be provided in the code charts.
>
> D57xx: *Emoji Variation Sequence:* A Variation Sequence for emoji listed in the file emoji-variation-sequences.txt in the Unicode Character Database.
>
> - The intended effect of an emoji variation sequence is described in UTS #51 and the associated data file.
>
> D57xx: *Ideographic Variation Sequence (IVS):* A Variation Sequence for ideographs that has been accepted for registration via the process defined in Unicode Technical Standard #37, “Unicode

Ideographic Variation Database," and is listed in the IVD_Sequences.txt file in the Ideographic Variation Database (IVD).

- The intended glyph for an ideographic variation sequence is registered in the IVD.

An implementation might not support a Variation Sequence as a request for a modified or limited range of appearance for the Variation Base character: The sequence might be an *unassigned* Variation Sequence (one that is not listed as any of the three types above), the implementation might not be able to support it, or it might choose not to support it. The implementation then displays the Variation Base character as if the Variation Selector was not present and, in normal rendering, does not display the Variation Selector. (See *Section 5.21.6, Characters Ignored for Display*.)

Note: A Variation Sequence that is unassigned may *become* assigned in the future. Because different programs or modules exchanging text may be using different versions of Unicode, each implementation needs to treat such a Variation Sequence like an unsupported one. This ensures that any sequence that could be assigned in the future is treated with a reasonable fallback appearance by a downlevel implementation.

Any Variation Selector that is not part of a Variation Sequence is treated like any other misplaced Default Ignorable character (see "Misplaced Default Ignorable"). In particular, a conformant implementation does not interpret such a Variation Selector as a request to modify the glyph shape of a preceding character. Instead, the Variation Selector is displayed like a missing glyph.

NOTE: If the description of variation selectors / variation sequences is consolidated and/or expanded on in Chapter 23 as suggested in the NOTE below, the above paragraph or something similar should be retained here so that it isn't overlooked. (This may result in a slight duplication).

In the following examples, the VS1 is not part of a variation sequence. In the last case, the VS2 *is* part of a variation sequence, but the VS1 is not.

- <CR><VS1>
- <VS1>A
- A<VS2><VS1>

This does not preclude implementations from providing specialized displays, such as a "Show Hidden" option that displays invisible characters (eg, VS or gc=Cf) or confusable whitespace (gc=Zs) with special symbols or stylized acronyms.

Aside from appearance, a variation sequence generally has the same behavior as its base character. For example, it would line-break or word-break like its base. However, this does not preclude an implementation from distinguishing the two depending on the type of processing. For example, a word processor providing searching for emoji in text could distinguish ‼ from ‼️.

NOTE: There is existing text on Variation Selectors in chapter 23. It may be useful to move (and/or copy) some of the description of how to handle variation selectors / sequences into a revised section there rather than overburdening the definitions with a full explanation of how to handle variation selectors. The focus on that other section would be to give consolidated advice. The details are left to the editorial committee. This is no different from other instances where some other part of the standard, e.g. a UAX, cites some definitions and then provides a comprehensive coverage of an implementation topic.

Add a definition of Default Ignorable Code Points and Misplaced Default Ignorable Code Points in Chapter 3.

> Dxxx. *Default Ignorable Code Point*. A character with the property Default_Ignorable_Code_Point.
>
> - These are characters that ordinarily do not have visible glyphs on their own, although they may have an effect on the display of other characters.
>
> Dxxx. *Misplaced Default Ignorable Code Point*. A Default Ignorable Code Point that is placed in a context in which it is not possible to reasonably interpret the character, satisfying one or more of the criteria below.
>
> 1. A variation selector outside of a Variation Sequence (See Dxxx)
> 2. A tag character that is not part of a valid emoji tag sequence as defined in UTS #51: Unicode Emoji, Annex C: Valid Emoji Tag Sequences.
> 3. A Default Ignorable Code Point that is the same character as the preceding one is almost always misplaced; for example: a sequence of two ZWJ.
>   a. Exception: Sequences of tag characters (U+E00xx) can include repeated tag characters.
> 4. A Hangul character (filler) that is not adjacent to a non-filler Hangul script character
> 5. A Musical begin or end character that is not adjacent to a non-default ignorable character in the Musical Symbols block
> 6. A Shorthand character that is not adjacent to a Duployan script character
> 7. A COMBINING GRAPHEME JOINER that is not between letters and/or combining marks
> 8. An unassigned character or deprecated character (regardless of context)
>
> For security reasons, a conformant implementation should not interpret any Misplaced Default Ignorable Code Point, nor ignore its presence, but instead flag it to the user by displaying it as a missing glyph.
>
> For more information about Default Ignorable Code Points, see Section 5.3 *Unknown and Missing Characters* and Section 5.20 *Unicode Security*.
>
> [Ed. Note: Here or in Section 5.3 we should add that the criteria for Misplaced Default Ignorable Code Point may be refined in successive versions of the standard. (Criteria should be clear-cut and easily testable.)]
>
> [Ed. Note: Section 5.3, *Unknown and Missing Characters* should be renamed when a discussion of Default Ignorable Code Points is included.]

Section 5.20, Unicode Security

Add the following clarification (or language like it) to the discussion of "invisible" characters in Section 5.20, Unicode Security and cite the new discussion from Section 5.21.6, Characters Ignored for Display

> The purpose of the default ignorable property is to allow implementations to use affected characters in contexts where they serve a purpose defined in the standard, while ensuring that implementations that do not, or do not wish to, support them, have a silent fall-back by ignoring the character.
>
> However, for certain default ignorable characters, the standard defines contexts in which they are *misplaced*. For example, a variation selector after a carriage return is misplaced. Using misplaced default ignorable code points can hide information from users, and can be used for various attacks, including spoofing or cross-prompt injection attacks. For that reason, implementations should display those misplaced default ignorable code points using the missing glyph. (See also Section 5.21.6 “Characters Ignored for Display”.)
>
> [Ed. Note: this text covers the intent, the actual text may be revised or extended as part of Editorial Review.]