

Title: Preventing Sentence Breaks within Words like “Mr.Hamster”
Source: Mark Davis (Google, Inc.), Peter Edberg (Apple, Inc.),
 Laurențiu Iancu (Microsoft Corporation), Steven Loomis (IBM Corporation)
Status: Individual contribution
Action: For consideration by the Unicode Technical Committee
Date: 2015-02-02

1. Proposal

In response to Action Item [\[138-A94\]](#), this document proposes a possible solution to the issue of undesired sentence breaks occurring within a single word segment such as “Mr.Hamster”. The proposal, scoped to the specific issue originally reported [\[PR1240\]](#), is the following:

Proposal statement. Modify rule SB7 of the Sentence Boundary Algorithm in Unicode 8.0 to trigger on both uppercase and lowercase letters preceding ATerm characters:

SB7	(Upper Lower) ATerm × Upper
-----	-------------------------------

This solution, described in [Section 4](#), has the advantage of involving minimal changes, but may not necessarily prevent all cases of sentence breaks within single words. A more systematic approach to the broader problem, although one incurring more extensive changes, is outlined in [Section 5](#). An action is suggested to investigate it further, for possible inclusion in Unicode 9.0.

2. Background

In Unicode 5.1, the Word_Break property value MidNumLet was introduced, and corresponding UAX #29 rules were added, to allow a few punctuation characters like U+002E FULL STOP to appear within numbers and alphabetic words such as “U.S.A” without segmenting them. Feedback was then received pointing out that an undesired side effect is that a word such as “Mr.Hamster” (with no space after the period) contains no word break opportunities yet it contains a sentence break after the period [\[14-061\]](#).

While the example of “Mr.Hamster” or, more generally, a sequence of the form <lowercase letter, period, uppercase letter> with no intervening space may be somewhat artificial, the presence of a sentence boundary within a word segment is anomalous and ought to be addressed in UAX #29.

3. Problem analysis

The problem can be analyzed by comparing word and sentence boundaries in all character sequences of the form <letter, period, letter> with no spaces around the period, where either *letter* may be lowercase or uppercase. As of Version 7.0 of UAX #29, in three of the four sequences there are no word or sentence boundaries on either side of the period. One sequence is the anomalous case, namely <lowercase letter, period, uppercase letter>. In that sequence, there are no word boundaries while there is a sentence boundary after the period. The four combinations are summarized in the following table, with the problematic case highlighted in yellow:

Sequence	Word boundaries	Sentence boundaries
c . d	c . d	c . d
C . d	C . d	C . d
c . D	c . D	c . D
C . D	C . D	C . D

In the above table, the character sequences contain no spaces around the period; the spacing was added only for illustration. The word and sentence boundaries are marked with vertical bars.

The lack of word break opportunities on either side of U+002E FULL STOP (Word_Break = MidNumLet) in all four sequences is the result of rules WB6 and WB7 of the Word Boundary Algorithm [UAX29]:

WB6	(ALetter Hebrew_Letter) × (MidLetter MidNumLet Single_Quote) (ALetter Hebrew_Letter)
WB7	(ALetter Hebrew_Letter) (MidLetter MidNumLet Single_Quote) × (ALetter Hebrew_Letter)

In terms of sentence boundaries, rules SB7 and SB8 of the Sentence Boundary Algorithm prevent breaks after the period U+002E FULL STOP (Sentence_Break = ATerm) when the period is between two uppercase letters (SB7) or is followed by a lowercase letter (SB8). The case when the period is between a lowercase and an uppercase letter falls through to rule SB11, which assigns a sentence break opportunity after the period. In all four cases, a sentence break before the period is prevented by rule SB12:

SB7	Upper ATerm × Upper
SB8	ATerm Close* Sp* × (-(OLetter Upper Lower Sep CR LF STerm ATerm))* Lower
SB11	(STerm ATerm) Close* Sp* (Sep CR LF)? ÷
SB12	Any × Any

4. Proposed solution

One scoped solution to prevent a sentence break opportunity in the problematic case identified earlier, i.e., after the period in a sequence of the form *<lowercase letter, period, uppercase letter>*, is to modify rule SB7 of the Sentence Break Algorithm to trigger on both uppercase and lowercase letters preceding ATerm characters:

SB7	(Upper Lower) ATerm × Upper
-----	---------------------------------------

This approach has the advantage that no Sentence_Break categories are introduced or refactored and, hence, no updates to other rules resulting from the refactored categories are incurred, either. For that reason, it is the solution proposed in this document. Its disadvantage is that it addresses a specific, anomalous sequence, without attempting to solve the broader problem of inconsistency between word and sentence boundaries. A possible approach to solving that problem is outlined in the next section.

5. Alternative approach

A more systematic way in which the Sentence Boundary Algorithm could be modified to prevent sentence breaks in sequences that currently trigger rule SB11 would be to insert two new sentence boundary rules between SB8 and SB11, ported from the word boundary rules WB6 and WB7, to explicitly catch the cases from the latter. The two additional rules are highlighted in the following table:

SB8	ATerm Close* Sp*	×	(-(OLetter Upper Lower Sep CR LF STerm ATerm))* Lower
SB8a	(STerm ATerm) Close* Sp*	×	(SContinue STerm ATerm)
SB8b	<i>ALetter</i>	×	<i>ASep ALetter</i>
SB8c	<i>ALetter ASep</i>	×	<i>ALetter</i>
SB9	(STerm ATerm) Close*	×	(Close Sp Sep CR LF)

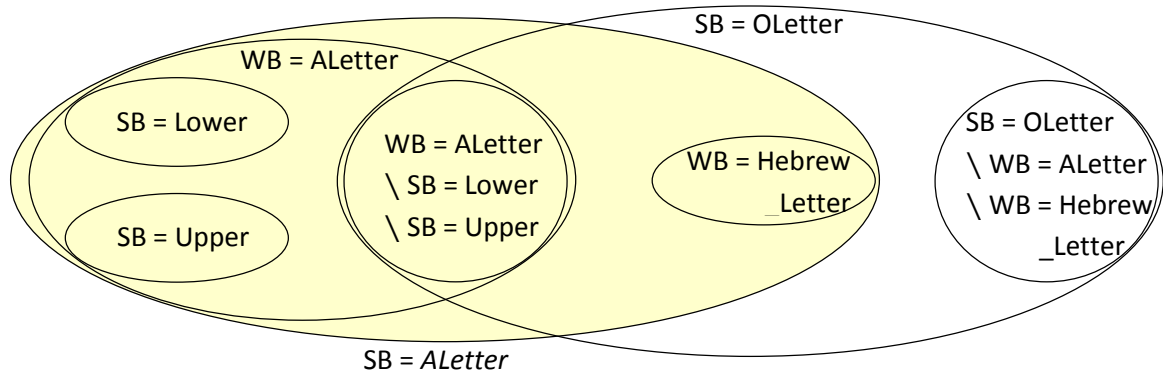
The new rules, SB8b and SB8c, use two new Sentence_Break categories labeled *ALetter* and *ASep*. These categories consist of the sets of characters from the ported rules WB6 and WB7:

Sentence_Break category	Summary list of characters
<i>ALetter</i>	Word_Break = ALetter, or Word_Break = Hebrew_Letter
<i>ASep</i>	Word_Break = MidLetter, or Word_Break = MidNumLet, or Word_Break = Single_Quote

The sets of characters in the new categories *ALetter* and *ASep* overlap with the sets of characters formed by existing Sentence_Break property values. To properly define the new categories, that overlap needs

to be eliminated by refactoring the sets of characters between existing and newly introduced property values.

Thus, the category *ALetter* overlaps with the character sets formed by the *Sentence_Break* property values Lower, Upper, and OLetter, as illustrated in the following diagram:

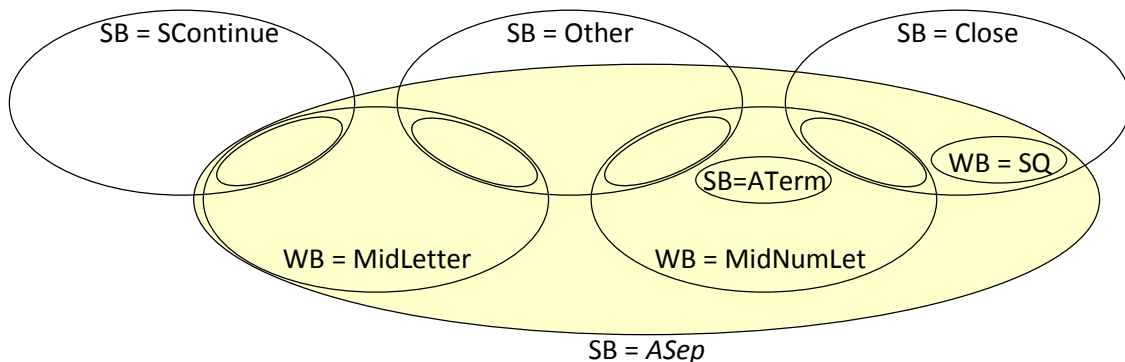


To eliminate the overlap and create disjoint sets of characters for the *Sentence_Break* property values, the codespace can be repartitioned as follows:

- Lower and Upper remain as is
- OLetter is shrunk to keep only the characters that are currently contained in the set $\{ \text{Sentence_Break} = \text{OLetter} \} \setminus \{ \text{Word_Break} = \text{ALetter} \} \setminus \{ \text{Word_Break} = \text{Hebrew_Letter} \}$
- A new *Sentence_Break* property value is introduced, consisting of the set of characters left from $\{ \text{Sentence_Break} = \text{OLetter} \}$ after the above refactoring

The repartitioning implies that the derivation of the *Sentence_Break* property value OLetter would be updated accordingly.

Similarly, the category *ASep* overlaps with the character sets formed by the *Sentence_Break* property values SContinue, Other and Close, and it fully includes the set of *ATerm*, as illustrated in the next diagram.



The Sentence_Break property values SContinue, Other and Close would all be refactored, and new property values introduced, to eliminate the overlap.

The rules of the Sentence Boundary Algorithm would then be updated to use the new Sentence_Break property values, combined by logical *or* wherever affected by the repartitioning. That update can be done mechanically.

5. Conclusion

The solution described in [Section 4](#) seems preferable because it is minimal in terms of changes to algorithm rules and properties, and it is the solution proposed here to fix the issue reported in [\[PRI240\]](#). If, instead, a more systematic approach is favored, then further investigation is needed, in an attempt to reduce the amount of changes incurred.

6. References

- [\[138-A94\]](#) Action Item for Mark Davis et al., *Investigate changes to sentence break to prevent words from spanning sentences in UAX #29*, February 2014, <http://www.unicode.org/L2/L2014/14026.htm#138-A94>.
- [\[14-061\]](#) Peter Edberg, *Clarifying UAX#29 sentence break vs word break: Action item 134-A78, handle PRI #240 feedback from Konstantin Ritt*, February 2014, <http://www.unicode.org/L2/L2014/14061-sent-word-brk.pdf>.
- [\[PRI240\]](#) Feedback on Public Review Issue #240, *Proposed Update UAX #29, Unicode Text Segmentation*, April 2013, <http://www.unicode.org/review/pri240/>.
- [\[UAX29\]](#) Mark Davis, *Unicode Standard Annex #29, Unicode Text Segmentation, Version 7.0.0*, June 2014, <http://www.unicode.org/reports/tr29/tr29-25.html>.