

**Subject:** Fixing breaking properties for emoji  
**Date:** 2016-01-27  
**From:** Mark Davis, Emoji Subcommittee  
**Working Draft:** <https://goo.gl/5p3dLx>

Certain sequences of emoji should be displayed with a single glyph if possible. For historical reasons, we have a number types of such sequences. However, these sequences are not yet fully reflected in the specs and data files for segmentation: **UAX #29** (grapheme cluster break, word break) and **UAX #14** (line break). That means that various bad effects can occur, such as having what the user normally sees as single characters being broken across lines.

While it is certainly possible for implementations to customize Unicode segmentation by testing for exactly those sequences of characters listed in [emoji/latest](#), that is not very robust, nor very fast, nor very future-proof (that is, where an old implementation recognizes a sequence sent to it by a new implementation).

This document proposes changes to property values and rules in the segmentation UAXes that encompass all the current emoji sequences, and anticipate to some extent possible future sequences with existing characters. Note that as usual with segmentation, it is not a problem to be broader than just the valid emoji sequences, and even prevent some non-emoji sequences from breaking—as long as those sequences don't occur with an significant frequency, or it doesn't matter that they don't break.

## Contents

[Background](#)

[Tag Sequences](#)

[Proposal](#)

[Flag Sequences](#)

[Proposal](#)

[Note](#)

[Modifier Sequences](#)

[Proposal—one of the following options:](#)

[Joiner Sequences](#)

[Proposal—one of the following options:](#)

## Background

The sequence types are listed below, with trailing characters having the following property values. Sentence break is not included, because none of the sequences would allow a break. The first 2 rows are already handled well enough, and are just presented for comparison.

Nº	Sequences	Examples	GCBreak	WBreak	LBreak
1	Non-spacing marks	20E3, 20E0	Extend	Extend	Combining_Mark
2	Variation	FE0E, FE0F	Extend	Extend	Combining_Mark
3	Tags*	E0020...	Control	Format	Combining_Mark
4	Flags	1F1E6...	Regional_Indicator	Regional_Indicator	Regional_Indicator
5	Modifier	1F3FB..F	Other	Other	Alphabetic
	Modifier_Base	261D...	Other	Other	Ideographic
6	Joiner	200D	Extend	Extend	Combining_Mark
	After_Joiner	1F466..1F469, 1F48B, 1F5E8 (current)	Other	Other	Ideographic Alphabetic (♥)

\* Tags are included because of prospective customizations using the TAG characters.

The raw data for these is listed in <http://www.unicode.org/L2/L2016/16011-data-file.txt>.

For reference, the rule sets cited below are:

- [Grapheme Cluster Boundary](#)
- [Word Boundary](#)
- [Linebreak Boundary](#)

An important feature is that the rules GB9, WB4, and LB9 cause characters with properties of GCB=Extend; WB=Extend; LB=Combining\_Mark to be “absorbed” into the previous characters. That is, there is never a break between them and the previous character, and subsequent rules ignore them.

The vast majority of the Emoji have LB=Ideographic (835) or LB=Alphabetic (143). If we restrict LB changes to those sets, we’d want to strongly discourage the use of any other LB classes in emoji zwj sequences. That shouldn’t be a problem since the remainder is a small number of characters, and much less likely to be used in sequences.

## Tag Sequences

The Tag characters are currently GCB=Control, which means that they don’t extend Grapheme\_Clusters. They also have General\_Category=Format. So that they work more properly in customization, they should have the same properties as Variation\_Selectors. That will cause them to normally “glue” to the previous characters, and otherwise be ignored by subsequent segmentation rules.

### Proposal

Change the GCB, WB, and LB property values of Tag characters to be the same as those of Variation Selectors.

Add a review note considering whether to also change the General\_Category to be Mn and the Bidi\_Class property correspondingly to BC=Nonspacing\_Mark, so that they work more like Variation Selectors in other processing.

## Flag Sequences

The basic support for flag sequences is present in TR29 and TR14, which is not to break between emoji flag sequences. Rule (GB8a/WB13c/LB30a) forbids any break between RI characters. So a sequences <RI RI RI RI> and <RI RI RI> will not break. A more sophisticated mechanism will break between any pairs, starting at the first. Thus there would be the following breaks in those sequences <RI RI | RI RI> and <RI RI | RI>.

### Proposal

Change current rules GB8a/WB13c/LB30a to:

**Break between two Regional Indicators if and only if there is an odd number of them before the point being considered:**

sot	(RI RI)*	RI	×	RI
[^RI]	(RI RI)*	RI	×	RI
		RI	÷	RI

## Modifier Sequences

These are special because they only combine with specific previous characters (modifier bases). When a Modifier doesn’t combine, it should be treated like a stand-alone character. So, for example, there can be a line break before it, etc.

However, such sequences will be extremely unusual cases, and for simplicity and future-proofing, it may be better to assign it properties that simply disallow breaks from most previous characters: for example, by giving it the same GC/W/LBreak property values as Variation.

If we want to limit the range of characters, then we'd introduce more new property values and rules.

## Proposal

Add GC/W/LBreak property values Emoji\_Base (EB) and Emoji\_Modifier (EM) using macros, and the following rules:

GB9c*	Emoji_Base	×	Emoji_Modifier
WB13d	Emoji_Base	×	Emoji_Modifier
LB30b	EB	×	EM

The Emoji\_Base and Emoji\_Modifier characters are derived from the emoji properties: characters with Emoji\_Modifier\_Base=Yes, and characters with Emoji\_Modifier=Yes, respectively. Also add a Review Note pointing out some alternatives for comparison.

### Alternative A. Simple but broad

- Set the GC/W/LBreak property values to be the same as Variation Selectors, and have no rule changes. The disadvantage of this is that Emoji\_Modifier characters wouldn't break from any previous base (including letters).

### Alternative B. Narrow (all and only emoji modifier sequences)

- Same as narrow, except limit LB property value changes to only LB=Ideographic (835) & LB=Alphabetic (143). The purpose of this would be to limit the changes to LB property values.

\* Rules after 9b would have a different header, and not only apply to extended grapheme clusters

We don't need to have Emoji Variation Selectors listed in the Narrow rules, because all of these rules are after those are absorbed.

## Joiner Sequences

The joiners already disallow breaks from the previous characters, but they do allow breaks from the following characters. For grapheme cluster breaks, we don't really have to do much. The text layout software (like Harfbuzz) should be aware of the boundaries, and not let people click in the middle of a sequence that shows up as a single glyph. However, when hitting a delete button, it might be better to remove the trailing joiner. We don't really specify "delete key behavior" in Unicode, however, although we have talked about doing something in CLDR.

For the other segmentations, we again have a choice. The simplest change would be to disallow breaks between a joiner and most following characters, but this is likely too broad (see Alternative A for more information). The proposed change disallows such breaks only between a joiner and characters that actually occur in current cataloged ZWJ sequences.

## Proposal (Narrow approach)

Add property values for After\_Joiners (AJ), using macros, and the following rules:

GB9d	ZWJ	×	After_Joiners
WB3c	ZWJ	×	After_Joiners
LB8a	ZWJ	×	After_Joiners

The AJ characters are all those that occur after joiners in the current cataloged ZWJ sequences. (These would need to be updated periodically).

Also add a review note pointing out some alternatives for comparison.

### Alternative A. Simple but broad

- Disallow breaks between a joiner and most following characters. The joiners are currently targeted for usage with letters from languages that would not allow word- or line-breaks around them anyway,

such as between Arabic letters or within Devanagari words. This would have the slight disadvantage of disallowing (say) linebreaks within <chinese\_character> + ZWJ + <chinese\_character>. However, those sequences are unnecessary and probably always just accidental. Thus the chances of a negative impact is extremely small.

- Even this simple change requires some rule changes, since we have to distinguish between Joiners and other extending characters: Separate ZWJ into its own WB, and LB property values, called ZWJ (ZWJ). Use macros to do this so that the current rules using Word\_Break=Extend, and Line\_Break=Combining\_Mark remain the same. Add the following rules:

WB3c	ZWJ	×
LB8a	ZWJ	×

### Alternative B. Intermediate

- Like the proposal, but future-proof the expressions for most emoji by expanding After\_Joiners to encompass all characters with GCB=Other; WB=Other; LB=Ideographic; and Emoji=Yes, thus including 769 other emoji.

GB9d	ZWJ	×	After_Joiners
WB3c	ZWJ	×	After_Joiners
LB8a	ZWJ	×	After_Joiners

### Alternative C. Narrower

- Only handle the existing emoji joiner sequences. Add property values for Adjacent\_Joining (AJ), using macros. The AJ characters are restricted to the ones appearing either before **or after** ZWJ in the current sequences, and would need to be updated periodically.

GB9d	Adjacent_Joining	ZWJ	×	Adjacent_Joining
WB3c	Adjacent_Joining	ZWJ	×	Adjacent_Joining
LB8a	Adjacent_Joining	ZWJ	×	Adjacent_Joining