Re: Other properties in segmentation rules
From: Mark Davis
Date: 2016-07-18
Draft: https://goo.gl/MxceSH

Proposal

In the segmentation UAXs (#29, #14) add the following text.

Segmentation rules may use other sets of characters based on Unicode properties in addition to the associated segmentation property values. When that is done, the regular property set syntax is used, such as:

Numeric × \p{script=Arab}

This is especially important for customizations of rules, such as in CLDR, but also allows for future use in the UAX rules.

Background

We have followed the practice in the UAXs that the segmentation rules only use segmentation properties. We instituted this very early on, when the rules were simpler. There is, however, no hard requirement that a Unicode algorithm use only a single property; normalization, for example, uses several.

As the rules have gotten more complicated, however, at this point the practice makes the rules more difficult to follow and to implement. The segmentation properties are enumerated and single-valued, thus forming a partition. That is not very flexible, since any time we would like to test against a new set of items, we end up splitting every property value whose range intersects with that set. We've attempted to make the rules a bit shorter by introducing "macros", but everything is more complicated than it should be. We are often forced to fall short of the ideal implementation solely because the rules would just get too complicated.

Examples of splits:

- (ALetter | Hebrew_Letter)
- (MidNumLet | Single_Quote)
- (STerm | ATerm)
- (Glue_After_Zwj | EBG)
- (Extend | ZWJ)
- (AL | HL)

Most recently, we didn't set Glue_After_Zwj on all the characters we wanted, because it would cause us to split more sets than we wanted. We introduced rules like:

ZWJ × (Glue_After_Zwj | EBG)

When what we really wanted to do (for future-proofing) was something like:

ZWJ × \p{Emoji}

We were forced to push a more thorough approach into CLDR ("Some changes to rules and data are needed for best segmentation behavior of additional emoji zwj sequences [UTR51], prior to the eventual publication of Unicode 10.0. Such changes are planned for inclusion in CLDR Version 30 [CLDR].")

Indicating that other property-based sets can be used in the rules not only normalizes their use in CLDR (and by any other customizations), but also allows us to avoid jumping through hoops in the future in the UAXes.

Testing Implications

The segmentation test files use a set of test characters, one from each of the segmentation property value sets. Where we use a set based on a property X, the set of test characters would need to be expanded. (This is both in the UAX rules and for customizations such as in CLDR.) The expansion would only occur when there is overlap: a segmentation property value set has some characters included in X and some not included in X. In cases of overlap, there would be two test characters instead of one.

This is a straightforward modification. Below is sample code for where the LineBreak rules also use [:script=Hebr:].

```
for (Line_Break_Values lineBreakValue : Line_Break_Values.values()) {
    UnicodeSet lineBreakPropertyValueSet = lineBreakProperty.getSet(lineBreakValue);
    String lbvName = lineBreakValue.getShortName();

    UnicodeSet difference = new UnicodeSet(lineBreakPropertyValueSet).removeAll(hebrewScript);
    UnicodeSet intersection = new UnicodeSet(lineBreakPropertyValueSet).retainAll(hebrewScript);
    if (difference.isEmpty() || intersection.isEmpty()) {
        addFirstCharacter(lbvName + "\t " + otherPropName, lineBreakPropertyValueSet, testCharacters);
    } else {
        addFirstCharacter(lbvName + "\t-" + otherPropName, difference, testCharacters);
        addFirstCharacter(lbvName + "\t+" + otherPropName, intersection, testCharacters);
    }
}
```

And the results would be the following (with +/- and underline where an additional test character is generated).

AL	-Hebr	0023	NUMBER SIGN
AL	+Hebr	05C0	HEBREW PUNCTUATION PASEQ
B2	Hebr	2014	EM DASH
BA	-Hebr	0009	<control-0009>
BA	+Hebr	05BE	HEBREW PUNCTUATION MAQAF
BB	Hebr	00B4	ACUTE ACCENT
BK	Hebr	000B	<control-000B>
CB	Hebr	FFFC	OBJECT REPLACEMENT CHARACTER
CJ	Hebr	3041	HIRAGANA LETTER SMALL A
CL	Hebr	007D	RIGHT CURLY BRACKET
CM	-Hebr	0000	<control-0000>
CM	+Hebr	0591	HEBREW ACCENT ETNAHTA
CP	Hebr	0029	RIGHT PARENTHESIS
CR	Hebr	000D	<control-000D>
EB	Hebr	261D	WHITE UP POINTING INDEX
EM	Hebr	1F3FB	EMOJI MODIFIER FITZPATRICK TYPE-1-2
EX	-Hebr	0021	EXCLAMATION MARK
EX	+Hebr	05C6	HEBREW PUNCTUATION NUN HAFUKHA
GL	Hebr	00A0	NO-BREAK SPACE
H2	Hebr	AC00	HANGUL SYLLABLE GA
H3	Hebr	AC01	HANGUL SYLLABLE GAG
HL	Hebr	05D0	HEBREW LETTER ALEF
HY	Hebr	002D	HYPHEN-MINUS
ID	Hebr	231A	WATCH
IN	Hebr	2024	ONE DOT LEADER
IS	Hebr	002C	COMMA
JL	Hebr	1100	HANGUL CHOSEONG KIYEOK
JT	Hebr	11A8	HANGUL JONGSEONG KIYEOK
JV	Hebr	1160	HANGUL JUNGSEONG FILLER
LF	Hebr	000A	<control-000A>
NL	Hebr	0085	<control-0085>
NS	Hebr	17D6	KHMER SIGN CAMNUC PII KUUH
NU	Hebr	0030	DIGIT ZERO
OP	Hebr	0028	LEFT PARENTHESIS
PO	Hebr	0025	PERCENT SIGN
PR	Hebr	0024	DOLLAR SIGN
QU	Hebr	0022	QUOTATION MARK
RI	Hebr	1F1E6	REGIONAL INDICATOR SYMBOL LETTER A
SA	Hebr	0E01	THAI CHARACTER KO KAI
SG	Hebr	D800	<surrogate-D800>
SP	Hebr	0020	SPACE
SY	Hebr	002F	SOLIDUS
WJ	Hebr	2060	WORD JOINER
XX	Hebr	0378	<reserved-0378>
ZW	Hebr	200B	ZERO WIDTH SPACE
ZWJ	Hebr	200D	ZERO WIDTH JOINER