This is a continuation of L2/21-002 UTS #18 Editorial additions, a proposal for changes to https://unicode.org/reports/tr18/. It is broken out because a landscape view makes it much easier to see these changes. This is simply for review; the final text in UTS #18 would not be this wide: the 2nd two columns will fit in a normal page width.

**Problem:** There is EBNF used to describe the structure of a Character Class. It needs some cleanup, because there are some omissions, typos, and stylistic inconsistencies. (These were obscured because the EBNF is split by section, as each feature is introduced.)

**Proposal:** the old text (OLD) would be replaced by a table with the second & third columns, in each respective section, applying any necessary cleanup to references to the EBNF terms. Note that the precise syntax is not normative in UTS #18, since it is typically adapted to the current syntax of any particular Regex Engine. The new EBNF may deviate from the particulars of the old EBNF.
The EBNF can be modified for compatibility with existing syntax, or enhanced with other features. For example, to allow ignored spaces for readability, add `\u{20}` to SYNTAX_CHAR, and add `SP?` around various elements, change `ITEM*` to `SP? ITEM (SP? ITEM)*`, etc.

In subsequent sections of this document, additional EBNF lines will be added for additional features. In one case, marked in a comment, one of the above lines will be replaced.
Add below: \u{3b1 3b3 3b5 3b9} is just semantic sugar for \u{3b1}\u{3b3}\u{3b5}\u{3b9} — useful for readability and concision but not a requirement. Thus [a-\{a\u{3b1} 3b3\}-ζ] behaves like [a-\u{3b1}\u{3b3}-ζ] == [a-αγ-ζ]
OPERATOR := "&&" // intersection: A \cap B
 := "--" // set difference: A \setminus B
 := "~~" // symmetric difference: (A \cup B) \setminus (A \cap B)

SEQUENCE := ITEM (SEQ_EXTEND)*
SEQ_EXTEND := OPERATOR CHARACTER_CLASS | ITEM
OPERATOR := '||'
 := '&&'
 := '--'
 := '~~'

---

Add below the rules:

For better backwards compatibility (and clarity), this syntax requires a character class after an operator. So 
[ab&&[bc]] is valid, but [ab&&bc] is not. If there might be an ITEM to the right of the operator, [ab--cd] could be 
interpreted as [[ab-c]d]. So it reduces the ambiguity to require a CHARACTER_CLASS and not just an ITEM.

However, the exact way that operator precedence is handled may differ by regex implementation.

---

ITEM := "\q{ (CODE_POINT (SP CODE_POINT)*)? }"
ITEM := '\q{ LITERAL* }'

SP := \u{20}

**Review Note:** the OLD SP was a duplicate of SEP and is now merged

---

<codepoint> := \"N( " <character_name> "\"
LITERAL := \"N( ' ID_CHAR+ '\"

**Add to previous LITERAL rules.**

Constraint: ID_CHAR+ = valid Unicode name or alias
Add in text below: For example, \p{name=/^(SMILING|GRINNING)/} is the set of all characters whose name matches the expression, such as 😄 U+1F929 GRINNING FACE WITH STAR EYES