Remove undefined behavior for invalid multibyte characters and non-initial shift states in preprocessing tokens and header names

Document: n3808

Author: Ryan Karl

Date: 2026-02-20

Changes: Identifiers, comments, string literals, character constants, and header names must form valid multibyte characters and begin and end in the initial shift state. Currently, encountering an invalid multibyte sequence or mis-aligned shift state in any of these contexts is “undefined behavior”. We propose making such occurrences constraint violations so compilers must diagnose them.

Undefined Behavior: (7) An identifier, comment, string literal, character constant, or header name contains an invalid multibyte character or does not begin and end in the initial shift state (J.2 (7), N3301). The editor should remove this from the Annex J.2 table.

Analysis:

Section 5.2.2 currently treats invalid multibyte sequences and shift-state mismatches in identifiers, comments, string literals, character constants, and header names as undefined behavior. This could lead to inconsistent diagnostics and could lead to inconsistent warnings between some toolchains (e.g. pre-ANSI or legacy toolchains vs. modern toolchains).

Recommendation:

The standard should be reworded to denote this as a constraint violation. This would promote consistency in diagnosing adverse situations involving invalid multibyte characters or initial shift state exceptions and align the standard with widespread compiler behavior and existing lexical constraints. See the appendix for examples.

Suggested Rewording (relative to N3685):

5.2.2 — Multibyte characters

1 The source character set may contain multibyte characters, used to represent members of the extended character set. The execution character set may also contain multibyte characters, which are not required to have the same encoding as for the source character set. For both character sets, the following shall hold:
— The basic character set shall be present and each character shall be encoded as a single byte.
— The presence, meaning, and representation of any additional members is locale-specific.
— A multibyte character set may have a state-dependent encoding, wherein each sequence of multibyte characters begins in an initial shift state and enters other locale-specific shift states when specific multibyte characters are encountered. While in the initial shift state, all single-byte characters retain their usual interpretation and do not alter the shift state. The interpretation for subsequent bytes in the sequence is a function of the current shift state.

— A byte with all bits zero shall be interpreted as a null character independent of shift state. Such a byte shall not occur as part of any other multibyte character.

Lexical constraints regarding initial shift state and the validity of multibyte character sequences appear in **6.4.1**.

~~For source files, the following shall hold:~~
~~— An identifier, comment, string literal, character literal, or header name shall begin and end in the initial shift state.~~
~~— An identifier, comment, string literal, character literal, or header name shall consist of a sequence of valid multibyte characters.~~

…

6.4.1 General

…

2 Each preprocessing token that is converted to a token shall have the lexical form of a keyword, an identifier, a constant, a string literal, or a punctuator. A single universal character name shall match one of the other preprocessing token categories.

For source files, the following shall hold:

— An identifier, comment, string literal, character constant, or header name shall begin and end in the initial shift state.
— An identifier, comment, string literal, character constant, or header name shall consist of a sequence of valid multibyte characters.

Example

To illustrate a constraint violation without relying on glyph rendering, consider this excerpt (byte offsets in hex) of a source file encoded as UTF-8:

```
00000020: 2f 2a 20 43 6f 6d 6d 65 6e 74 20 77 69 74 68 20

00000030: c3 28 20 62 79 74 65 73 20 2a 2f
```

The comment begins at offset 0x20 with the bytes 2f 2a ("/*"). The comment terminator "*/" is the two-byte sequence 2a 2f at offsets 0x39-0x3A. The byte sequence at offset 0x30 is c3 28, which does not form a valid UTF-8 multibyte character. Therefore, the comment does not "consist of a sequence of valid multibyte characters." A similar example can be constructed for an encoding with state-dependent shift states in which an identifier, comment, string literal, character literal, or header name fails to begin and end in the initial shift state.

Acknowledgments: Thanks to the UB study group, David Svoboda, Dave Banham, and Joseph S. Meyers.