Remove undefined behavior for mismatched quote characters

Document: n3806

Author: Ryan Karl

Date: 2026-02-20

Changes: Remove undefined behavior if there are nonmatching single or double quotes (based on working draft N3685).

Undefined Behavior: An unmatched ' or " character is encountered on a logical source line during tokenization (J.2 (20), N3685). The editor should remove this from Annex J.2.

Analysis: Section 6.4 classifies preprocessing tokens into several categories and then includes a final catch-all category for a single non-white-space character that does not match any of the earlier categories. In that context, the sentence "If a ' or a " character matches the last category, the behavior is undefined" is the rule that reaches unmatched quote characters during tokenization. The current wording offers no guidance to implementors and can lead to inconsistent implementations. Unmatched quotes might slip through in some toolchains (e.g. pre-ANSI or legacy toolchains), or emit confusing error messages in others.

Recommendation: A clear rule should be added to the standard that requires every quote to form a valid character constant or string literal. This will formally align the C standard with virtually every compiler in widespread use. See the appendix for examples.

Suggested rewording (relative to N3685):

6.4.1 Lexical elements

…
Constraints
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. Every ' or " character shall be accompanied by a matching ' or " character. If a **'** or **"** character would otherwise match the last category, the program violates the constraint; a diagnostic is required.

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

Appendix:
Consider the program below:

```
/* example.c */

#include <stdio.h>

int main(void) {
```

```
    char c = 'A;            /* unmatched single-quote */
    char *s = "Hello, world;/* unmatched double-quote */
    (void)c;
    (void)s;
    return 0;
}
```

Compiling this code with older releases of popular compilers demonstrates longstanding practices. For example, compiling on gcc 6.1 we observe the following output:

```
<source>: In function 'main':
<source>:5:14: error: missing terminating ' character
     char c = 'A;            /* unmatched single-quote */
              ^
<source>:5:14: error: missing terminating ' character
     char c = 'A;            /* unmatched single-quote */
              ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<source>:6:5: error: expected expression before 'char'
     char *s = "Hello, world;/* unmatched double-quote */
     ^~~~
<source>:6:15: error: missing terminating " character
     char *s = "Hello, world;/* unmatched double-quote */
               ^
<source>:6:15: error: missing terminating " character
     char *s = "Hello, world;/* unmatched double-quote */
               ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<source>:8:11: error: 's' undeclared (first use in this function)
     (void)s;
           ^
<source>:8:11: note: each undeclared identifier is reported only once for
each function it appears in
Compiler returned: 1
```

Compiling on clang 3.5 we observe the following output:

```
<source>:5:14: error: missing terminating ' character [-Werror,-Winvalid-pp-
token]
    char c = 'A;            /* unmatched single-quote */
             ^
<source>:5:14: error: expected expression
<source>:6:15: error: missing terminating '"' character [-Werror,-Winvalid-
pp-token]
    char *s = "Hello, world;/* unmatched double-quote */
              ^
<source>:8:11: error: use of undeclared identifier 's'
    (void)s;
          ^
```

```
4 errors generated.
Compiler returned: 1
```

Compiling on TI C6x gcc 12.4.0 we observe the following output:

```
<source>: In function 'main':
<source>:5:14: error: missing terminating ' character
    5 |     char c = 'A;            /* unmatched single-quote */
      |              ^
<source>:5:14: error: missing terminating ' character
    5 |     char c = 'A;            /* unmatched single-quote */
      |              ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<source>:6:5: error: expected expression before 'char'
    6 |     char *s = "Hello, world;/* unmatched double-quote */
      |     ^~~~
<source>:6:15: error: missing terminating " character
    6 |     char *s = "Hello, world;/* unmatched double-quote */
      |               ^
<source>:6:15: error: missing terminating " character
    6 |     char *s = "Hello, world;/* unmatched double-quote */
      |               ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<source>:8:11: error: 's' undeclared (first use in this function)
    8 |     (void)s;
      |           ^
<source>:8:11: note: each undeclared identifier is reported only once for
each function it appears in
Compiler returned: 1
```

Compiling on icc 16.0.3 we observe the following output:

```
<source>(5): error: missing closing quote
      char c = 'A;            /* unmatched single-quote */
               ^

<source>(6): error: expected a ";"
      char *s = "Hello, world;/* unmatched double-quote */
      ^

<source>(6): error: missing closing quote
      char *s = "Hello, world;/* unmatched double-quote */
                ^

<source>(8): error: identifier "s" is undefined
      (void)s;
            ^

compilation aborted for <source> (code 2)
Compiler returned: 2
```

Compiling on msvc 16.1 we observe the following output:

```
example.c
<source>(5): error C2001: newline in constant
<source>(5): error C2015: too many characters in constant
<source>(6): error C2143: syntax error: missing ';' before 'type'
<source>(6): error C2001: newline in constant
<source>(7): error C2064: term does not evaluate to a function taking 259
arguments
<source>(7): error C2143: syntax error: missing ')' before 'type'
<source>(7): error C2059: syntax error: ')'
Compiler returned: 2
```