Proposal for C2X

WG14 N 3046

Title:	$ in Identifiers
Author, affiliation:	Robert C. Seacord, Woven Planet rcseacord@gmail.com Steve Downey, Bloomberg, USA <sdowney@gmail.com, sdowney2@bloomberg.net> Peter Bindels, TomTom, Netherlands, <dascandy@gmail.com>
Date:	2022-7-26
Proposal category:	Defect
Target audience:	Implementers
Abstract:	Allow `$` as an implementation extension in identifiers
Prior art:	C23

$ in Identifiers

Reply-to: Robert C. Seacord (rcseacord@gmail.com)

Document No: **N 3046**

Reference Document: N2939, N2836**,** P1949R7 (http://wg21.link/p1949)

Date: 2022-3-02

This paper is to repair a potential defect introduced by voting N2836 Identifier Syntax using Unicode Standard Annex 31 into C23.

Change Log

2022-7-26:

· Initial version

1.0 PROBLEM DESCRIPTION

A question was raised at the July 2022 WG14 meeting concerning going back to the original identifier rules. The following straw poll was taken:

Straw poll: Does WG14 want to bring back the original identifier rules (e.g., allow `$` in identifiers as an extension, but not required to allow it)?

The results had clear consensus:

Results: 10 yes 2 no 8 abstain

Further discussion showed that the actual direction was less clear with the following opinions being noted:

- Each programming language can define its identifier syntax as relative to the Unicode identifier syntax, such as saying that identifiers are defined by the Unicode properties, with the addition of `$`.
- The original text allowed any implementation-defined characters, not just $
- I am strongly against what I'm suggesting but the "best" solution is to revert the "other implementation-defined characters" that got removed
- I would be much strongly opposed to something that would mention $ or any other specific character explicitly

- Allowing $ in identifiers would be a massive and unjustifiable land grab for both C and C++
- Would the following change suffice?

  6.4.2.1#1 add to `identifier-nondigit:`

  other implementation-defined characters

- Probably adding that sentence to both `identifier-start` and `identifier-continue`

As can be seen, opinions ranged from reverting to implementation-defined characters to keeping the current wording.

A quick survey of existing practice shows that current versions of gcc, clang, and icc all allow the `$` character anywhere in an identifier by default:

https://godbolt.org/z/frGzcTWoK

Only clang will diagnose the use of a `$` in an identifier, but only in `-pedantic` mode.

In both GCC and Clang, this is controlled by the `-f[no-]dollars-in-identifiers` flag which defaults to allow.

This paper proposes allowing `$` anywhere in identifiers as an implementation extension.

2.0 PROPOSED WORDING

Wording Alternative #1

The $ does not currently appear in any production for identifiers. Using $ in an identifier is consequently undefined behavior. Implementations are free to provide their own definition for this otherwise undefined behavior, and allow $ in identifiers.

Add the text in green to the end of Subclause 5.2.1 Character sets, paragraph 3:

If any other characters are encountered in a source file (except in an identifier, a character constant, a string literal, a header name, a comment, or a preprocessing token that is never converted to a token), the behavior is undefined. The $ character is reserved for use in identifiers as an implementation-defined extension.

Wording Alternative #2

Add the text in green to Subclause 6.4.2.1 paragraph 2 in the N2912 working draft:

An identifier is a sequence of nondigit characters (including the underscore _, the lowercase and uppercase Latin letters, and other characters) and digits, which designates one or more entities as described in ??. The nondigit characters may also include a dollar sign $. Lowercase and uppercase letters are distinct. There is no specific limit on the maximum length of an identifier.

Wording Alternative #3

Add the text in green in the N2912 working draft:

Subclause 6.4.2.1 paragraph 1

nondigit: one of

```
        _ $ a b c d e f g h i j k l m
            n o p q r s t u v w x y z
            A B C D E F G H I J K L M
            N O P Q R S T U V W X Y Z
```

Subclause 6.4.2.1 paragraph 2

An identifier is a sequence of nondigit characters (including the underscore _, the dollar sign $, the lowercase and uppercase Latin letters, and other characters) and digits, which designates one or more entities as described in ??. It is implementation-defined if a dollar sign $ may be used as a nondigit character. Lowercase and uppercase letters are distinct. There is no specific limit on the maximum length of an identifier.

4.0 Acknowledgements

We would like to recognize the following people for their help with this work: Jens Maurer, Zach Laine, Tom Honermann, Corentin Jabot, and Aaron Ballman.

5.0 References

[AltId] Unicode Standard Annex.

http://www.unicode.org/reports/tr31/tr31-11.html#Alternative_Identifier_Syntax

[DefId] Unicode Standard Annex.

http://www.unicode.org/reports/tr31/tr31-11.html#Default_Identifier_Syntax

[N3146] Clark Nelson. 2010. Recommendations for extended identifier characters for C and C++.

https://wg21.link/n3146

[UAX15] Ken Whistler. Unicode Normalization Forms.

http://www.unicode.org/reports/tr15

[UAX31] Mark Davis. Unicode Identifier and Pattern Syntax.

http://www.unicode.org/reports/tr31

[UAX36] Mark Davis and Michel Suignard. Unicode Security Considerations.

http://www.unicode.org/reports/tr36

[UAX44] Ken Whistler and Laurențiu Iancu. Unicode Character Database.

http://www.unicode.org/reports/tr44

[UTS51] Mark Davis and Peter Edberg. Unicode Emoji.

http://www.unicode.org/reports/tr51