P1041R1:
Make char16_t/char32_t string literals be UTF-16/32

Today...

§ [lex.ccon]p3:

A character literal that begins with u8, such as u8'w', is a character literal of type char, known as a ***UTF-8** character literal*. The value of a **UTF-8** character literal is equal to its **ISO/IEC 10646** code point value, provided that the code point value is representable with a single **UTF-8** code unit (that is, provided it is in the **C0 Controls and Basic Latin Unicode block**). If the value is not representable with a single **UTF-8** code unit, the program is ill-formed. A **UTF-8** character literal containing multiple c-chars is ill-formed.

Today...

§ [lex.ccon]p4:

A character literal that begins with the letter u, such as u'x', is a character literal of type char16_t. The value of a char16_t character literal containing a single c-char is equal to its **ISO/IEC 10646** code point value, provided that the code point value is representable with a single 16-bit code unit (that is, provided it is in the **basic multi-lingual plane**). If the value is not representable with a single 16-bit code unit, the program is ill-formed. A char16_t character literal containing multiple c-chars is ill-formed.

Today...

§ [lex.ccon]p5:

A character literal that begins with the letter `U`, such as `U'y'`, is a character literal of type `char32_t`. The value of a `char32_t` character literal containing a single c-char is equal to its **ISO/IEC 10646** code point value. A `char32_t` character literal containing multiple c-chars is ill-formed.

Today...

§ [lex.string]p7:

A string-literal that begins with `u8`, such as `u8"asdf"`, is a **UTF-8** string literal.

---

§ [lex.string]p10:

A string-literal that begins with `u`, such as `u"asdf"`, is a `char16_t` string literal. … A single c-char may produce more than one `char16_t` character in the form of **surrogate pairs**.

---

§ [lex.string]p11:

A string-literal that begins with `U`, such as `U"asdf"`, is a `char32_t` string literal.

Meanwhile, over in WG14...

§ 6.10.8.2 Environment macros:

__STDC_ISO_10646__:
An integer constant of the form **yyyymmL ...** ***(****if wchar_t literals are UCS-2 or UTF-32)*.

__STDC_UTF_16__:
The integer constant 1, intended to indicate that values of type char16_t are UTF–16 encoded. If some other encoding is used, the macro shall not be defined and the actual encoding used is implementation-defined.

__STDC_UTF_32__:
The integer constant 1, intended to indicate that values of type char32_t are UTF–32 encoded. If some other encoding is used, the macro shall not be defined and the actual encoding used is implementation-defined.

Back in WG21...

§ [cpp.predefined]p2:

__STDC_ISO_10646__:
An integer literal of the form **yyyymmL ...** ***(****if wchar_t literals are UCS-2 or UTF-32)*.

No mention of **__STDC_UTF_16__** or **__STDC_UTF_16__**...

Proposed...

§ [lex.ccon]p4:

A character literal that begins with the letter u, such as u'x', is a character literal of type char16_t, known as a UTF-16 character literal. The value of a ~~char16_t~~UTF-16 character literal containing a single c-char is equal to its ISO/IEC 10646 code point value, provided that the code point value is representable with a single 16-bit code unit (that is, provided it is in the basic multi-lingual plane). If the value is not representable with a single 16-bit code unit, the program is ill-formed. A ~~char16_t~~UTF-16 character literal containing multiple c-chars is ill-formed.

Proposed...

§ [lex.ccon]p5:

A character literal that begins with the letter `U`, such as `U'x'`, is a character literal of type `char32_t`, known as a UTF-32 character literal. The value of a ~~`char32_t`~~UTF-32 character literal containing a single c-char is equal to its ISO/IEC 10646 code point value. A ~~`char32_t`~~UTF-32 character literal containing multiple c-chars is ill-formed.

Proposed...

§ [lex.string]p10:

A string-literal that begins with u, such as u"asdf", is a ~~char16_t~~UTF-16 string literal. A ~~char16_t~~UTF-16 string literal has type "array of *n* const char16_t", where *n* is the size of the string as defined below; it is initialized with the given characters. A single c-char may produce more than one char16_t character in the form of surrogate pairs.

§ [lex.string]p10+1:

For a UTF-16 string literal, each successive element of the object representation has the value of the corresponding code unit of the UTF-16 encoding of the string.

Proposed...

§ [lex.string]p11:

A string-literal that begins with U, such as U"asdf", is a ~~char32_t~~UTF-32 string literal. A ~~char32_t~~UTF-32 string literal has type “array of *n* const char32_t”, where *n* is the size of the string as defined below; it is initialized with the given characters.

§ [lex.string]p11+1:

For a UTF-32 string literal, each successive element of the object representation has the value of the corresponding code unit of the UTF-32 encoding of the string.

Questions...

1) Do we want to mandate use of UTF-16 and UTF-32?

2) Do we want to require predefined macros?
__STDC_UTF_16__=1
__STDC_UTF_32__=1

Existing practice:

Gcc and Clang define __STDC_UTF_16__=1 and __STDC_UTF_32__=1 for both C and C++.

Microsoft does not not define __STDC_UTF_16__ or __STDC_UTF_32__ for either C or C++.