char, wchar_t, char8_t, char16_t, char32_t

Article • 05/08/2024

The types `char`, `wchar_t`, `char8_t`, `char16_t`, and `char32_t` are built-in types that represent alphanumeric characters, nonalphanumeric glyphs, and nonprinting characters.

Syntax

C++

```
char     ch1{ 'a' };  // or { u8'a' }
wchar_t  ch2{ L'a' };
char16_t ch3{ u'a' };
char32_t ch4{ U'a' };
```

Remarks

The `char` type was the original character type in C and C++. The `char` type stores characters from the ASCII character set or any of the ISO-8859 character sets, and individual bytes of multi-byte characters such as Shift-JIS or the UTF-8 encoding of the Unicode character set. In the Microsoft compiler, `char` is an 8-bit type. It's a distinct type from both `signed char` and `unsigned char`. By default, variables of type `char` get promoted to `int` as if from type `signed char` unless the /J compiler option is used. Under `/J`, they're treated as type `unsigned char` and get promoted to `int` without sign extension.

The type `unsigned char` is often used to represent a *byte*, which isn't a built-in type in C++.

The `wchar_t` type is an implementation-defined wide character type. In the Microsoft compiler, it represents a 16-bit wide character used to store Unicode encoded as UTF-16LE, the native character type on Windows operating systems. The wide character versions of the Universal C Runtime (UCRT) library functions use `wchar_t` and its pointer and array types as parameters and return values, as do the wide character versions of the native Windows API.

The `char8_t`, `char16_t`, and `char32_t` types represent 8-bit, 16-bit, and 32-bit wide characters, respectively. (`char8_t` is new in C++20 and requires the /std:c++20 or `/std:c++latest` compiler option.) Unicode encoded as UTF-8 can be stored in the `char8_t` type. Strings of `char8_t` and `char` type are referred to as *narrow* strings, even when used to encode Unicode or multi-byte characters. Unicode encoded as UTF-16 can be stored in the `char16_t` type, and Unicode encoded as UTF-32 can be stored in the `char32_t` type. Strings of these types and `wchar_t` are all referred to as *wide* strings, though the term often refers specifically to strings of `wchar_t` type.

In the C++ standard library, the basic_string type is specialized for both narrow and wide strings. Use `std::string` when the characters are of type `char`, `std::u8string` when the characters are of type `char8_t`, `std::u16string` when the characters are of type `char16_t`, `std::u32string` when the characters are of type `char32_t`, and `std::wstring` when the characters are of type `wchar_t`.

Other types that represent text, including std::stringstream and std::cout have specializations for narrow and wide strings.

Feedback