Character Sets

Visual Studio 2015

For the latest documentation on Visual Studio 2017, see Visual Studio 2017 Documentation.

The text of a C++ program is stored in source files that use a particular character encoding. The C++ standard specifies a basic source character set for source files and a basic execution character set for compiled files. Visual C++ allows an additional set of locale-specific characters to be used in source files and compiled files.

Character sets

The C++ standard specifies a *basic source character set* that may be used in source files. To represent characters outside of this set, additional characters can be specified by using a *universal character name*. When compiled, the *basic execution character set* and *basic execution wide-character set* represent the characters and strings that can appear in a program. The Visual C++ implementation allows additional characters in source code and compiled code.

Basic source character set

The *basic source character set* consists of 96 characters that may be used in source files. This set includes the space character, horizontal tab, vertical tab, form feed and new-line control characters, and this set of graphical characters:

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

0 1 2 3 4 5 6 7 8 9

_ { } [ ] # ( ) < > % : ; . ? * + - / ^ & | ~ ! = , \ ″ ’

Microsoft Specific

Visual C++ includes the $ character as a member of the basic source character set. Visual C++ also allows an additional set of characters to be used in source files, based on the file encoding. By default, Visual Studio stores source files by using the default codepage. When source files are saved by using a locale-specific codepage or a Unicode codepage, Visual C++ allows you to use any of the characters of that code page in your source code, except for the control codes not explicitly allowed in the basic source character set. For example, you can put Japanese characters in comments, identifiers, or string literals if you save the file using a Japanese codepage. Visual C++ does not allow character sequences that cannot be translated into valid multibyte characters or Unicode code points. Depending on compiler options, not all allowed characters may appear in identifiers. For more information, see Identifiers.

END Microsoft Specific

Universal character names

Because C++ programs can use many more characters than the ones specified in the basic source character set, you can specify these characters in a portable way by using *universal character names*. A universal character name consists of a sequence of characters that represent a Unicode code point. These take two forms. Use \UNNNNNNNN to represent a Unicode code point of the form U+NNNNNNNN, where NNNNNNNN is the eight-digit hexadecimal code point number. Use four-digit \uNNNN to represent a Unicode code point of the form U+0000NNNN.

Universal character names can be used in identifiers and in string and character literals. A universal character name cannot be used to represent a surrogate code point in the range 0xD800-0xDFFF. Instead, use the desired code point; the compiler

automatically generates any required surrogates. Additional restrictions apply to the universal character names that can be used in identifiers. For more information, see Identifiers and String and Character Literals.

Microsoft Specific

The Visual C++ compiler treats a character in universal character name form and literal form interchangeably. For example, you can declare an identifier using universal character name form, and use it in literal form:

C++

```
auto \u30AD = 42; // \u30AD is 'キ'
if (キ == 42) return true; // \u30AD and キ are the same to the compiler
```

The format of extended characters on the Windows clipboard is specific to application locale settings. Cutting and pasting these characters into your code from another application may introduce unexpected character encodings. This can result in parsing errors with no visible cause in your code. We recommend that you set your source file encoding to a Unicode codepage before pasting extended characters. We also recommend that you use an IME or the Character Map app to generate extended characters.

END Microsoft Specific

Basic execution character set

The *basic execution character set* and the *basic execution wide-character set* consist of all the characters in the basic source character set, and the control characters that represent alert, backspace, carriage return, and the null character. The *execution character set* and *execution wide-character set* are supersets of the basic sets. They include the implementation-defined source characters outside the basic source character set. The execution character set has a locale-specific representation.