**In this article**

1. [Replacement patterns](https://docs.microsoft.com/en-us/visualstudio/ide/using-regular-expressions-in-visual-studio?view=vs-2017#replacement-patterns)
2. [Regular expression examples](https://docs.microsoft.com/en-us/visualstudio/ide/using-regular-expressions-in-visual-studio?view=vs-2017#regular-expression-examples)

Visual Studio uses [.NET Framework regular expressions](https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expressions) to find and replace text.

**Replacement patterns**

To use a numbered capture group, surround the group with parentheses in the regular expression pattern. Use $number, where number is an integer starting at 1, to specify a specific, numbered group in a replacement pattern. For example, the grouped regular expression (\d)([a-z]) defines two groups: the first group contains a single decimal digit, and the second group contains a single character between **a** and **z**. The expression finds four matches in the following string: **1a 2b 3c 4d**. The replacement string z$1 references the first group only, and converts the string to **z1 z2 z3 z4**.

For information about regular expressions that are used in replacement patterns, see [Substitutions in regular expressions (.NET guide)](https://docs.microsoft.com/en-us/dotnet/standard/base-types/substitutions-in-regular-expressions).

**Regular expression examples**

Here are some examples:

|  |  |  |
| --- | --- | --- |
| **Purpose** | **Expression** | **Example** |
| Match any single character (except a line break) | . | a.o matches "aro" in "around" and "abo" in "about" but not "acro" in "across". |
| Match zero or more occurrences of the preceding expression (match as many characters as possible) | \* | a\*r matches "r" in "rack", "ar" in "ark", and "aar" in "aardvark" |
| Match any character zero or more times (Wildcard \*) | .\* | c.\*e matches "cke" in "racket", "comme" in "comment", and "code" in "code" |
| Match one or more occurrences of the preceding expression (match as many characters as possible) | + | e.+d matches "eed" in "feeder" but not "ed". |
| Match any character one or more times (Wildcard ?) | .+ | e.+e matches "eede" in "feeder" but not "ee". |
| Match zero or more occurrences of the preceding expression (match as few characters as possible) | \*? | e.\*?e matches "ee" in "feeder" but not "eede". |
| Match one or more occurrences of the preceding expression (match as few characters as possible) | +? | e.+?e matches "ente" and "erprise" in "enterprise", but not the whole word "enterprise". |
| Anchor the match string to the beginning of a line or string | ^ | ^car matches the word "car" only when it appears at the beginning of a line. |
| Anchor the match string to the end of a line | \r?$ | end\r?$ matches "end" only when it appears at the end of a line. |
| Anchor the match string to the end of the file | $ | end$ matches "end" only when it appears at the end of the file. |
| Match any single character in a set | [abc] | b[abc] matches "ba", "bb", and "bc". |
| Match any character in a range of characters | [a-f] | be[n-t] matches "bet" in "between", "ben" in "beneath", and "bes" in "beside", but not "below". |
| Capture and implicitly number the expression contained within parenthesis | () | ([a-z])X\1 matches "aXa"and "bXb", but not "aXb". "\1" refers to the first expression group "[a-z]". |
| Invalidate a match | (?!abc) | real(?!ity) matches "real" in "realty" and "really" but not in "reality." It also finds the second "real" (but not the first "real") in "realityreal". |
| Match any character that is not in a given set of characters | [^abc] | be[^n-t] matches "bef" in "before", "beh" in "behind", and "bel" in "below", but not "beneath". |
| Match either the expression before or the one after the symbol. | | | (sponge\|mud) bath matches "sponge bath" and "mud bath." |
| Escape the character following the backslash | \ | \^ matches the character ^. |
| Specify the number of occurrences of the preceding character or group | {x}, where x is the number of occurrences | x(ab){2}x matches "xababx", and x(ab){2,3}x matches "xababx" and "xabababx" but not "xababababx". |
| Match text in a Unicode character class. For more information about Unicode character classes, see [Unicode Standard 5.2 Character Properties](http://www.unicode.org/versions/Unicode5.2.0/ch04.pdf). | \p{X}, where "X" is the Unicode number. | \p{Lu} matches "T" and "D" in "Thomas Doe". |
| Match a word boundary | \b (Outside a character class \b specifies a word boundary, and inside a character class \b specifies a backspace.) | \bin matches "in" in "inside" but not "pinto". |
| Match a line break (that is, a carriage return followed by a new line). | \r?\n | End\r?\nBegin matches "End" and "Begin" only when "End" is the last string in a line and "Begin" is the first string in the next line. |
| Match any alphanumeric character | \w | a\wd matches "add" and "a1d" but not "a d". |
| Match any whitespace character. | (?([^\r\n])\s) | Public\sInterface matches the phrase "Public Interface". |
| Match any numeric character | \d | \d matches and "3" in "3456", "2" in 23", and "1" in "1". |
| Match a Unicode character | \uXXXX where XXXX specifies the Unicode character value. | \u0065 matches the character "e". |
| Match an identifier | \b[\_\w-[0-9]][\_\w]\*\b | Matches "type1" but not "&type1" or "#define". |
| Match a string inside quotes | ((\".+?\")|('.+?')) | Matches any string inside single or double quotes. |
| Match a hexadecimal number | \b0[xX]([0-9a-fA-F])\b | Matches "0xc67f" but not "0xc67fc67f". |
| Match integers and decimals | \b[0-9]\*\.\*[0-9]+\b | Matches "1.333". |

Tip

In Windows operating systems, most lines end in "\r\n" (a carriage return followed by a new line). These characters aren't visible, but are present in the editor and are passed to the .NET regular expression service.