C++11 ECMAScript regex

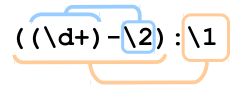
cpprocks.com

Syntax

.	Match any character
^	Match beginning of input
$	Match end of input
\b	Match word boundary
\B	Match anything other than a word boundary
|	Or operator

Capture groups

Denoted with parentheses
Referred to as \1, \2 etc.
Counted in order of left parentheses:



Repetition

Symbol	Repeats matched
?	<= 1
*	>= 0
+	>= 1
{n}	n
{n,}	>= n
{n, m}	>= n && <= m

Sets

Symbol	Matches
[abc]	Any of the characters included
[^abc]	Any of the characters NOT included
[a-z]	Any characters in the range
[a-zA-Z]	Any characters in the ranges
[=c=]	Equivalence class for the character
[.ae.]	Specified collating element

Classes

alpha	Lowercase and uppercase letters
digit or d	Digits; shorthand: \d
alnum or w	Characters from either *alpha* or *digit* classes shorthand for *[_[:alnum:]]*: \w
space or s	Whitespace characters; shorthand: \s
blank	Space or tab
cntrl	File format escape characters (\n, \r etc.)
punct	Punctuation characters
lower	Lowercase letters
upper	Uppercase letters
graph	Characters from *lower*, *upper*, *digit* or *punct*
print	Characters from either *graph* or *space*
xdigit	Hexadecimal digits (including both lowercase and uppercase a-f)

Algorithms

bool **regex_match** (first_iter, last_iter, match_res&, const& regex, [flags])
(first_iter, last_iter, const& regex, [flags])
(str, match_res&, const& regex, [flags])
(str, const& regex, [flags])

Returns true if the whole input string matches the regex;
details of the matches in *match_res*

bool **regex_search**

Returns true if a substring of the input string matches the regex;
Same parameters as *regex_match*

Regex constructor flags affecting *regex_match* & *regex_search*

match_not_bol	Don't treat the first position in the input as the beginning of line
match_not_eol	Don't treat the past-the-end position in the input as the end of a line
match_not_bow	Don't treat the first position in the input as the beginning of a word
match_not_eow	Don't treat the past-the-end position in the input as the end of a word
match_any	Any match is acceptable when more than one match is possible
match_not_null	Don't match an empty input
match_continuous	Don't search for matches other than at the beginning of the input
match_prev_avail	--*first* is a valid iterator; if set, ignore *match_not_bol* and *match_not_bow*

out_iter **regex_replace**(out_iter, first_iter, last_iter, const& regex,
const& format_str, [flags])

out_str **regex_replace**(const& input, const& regex, const& format_str,
[flags])

Replace substrings matching the regex according to the formatting string

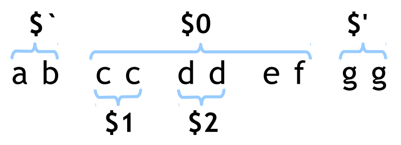
Regex flags affecting *regex_replace*

format_no_copy	Don't output the parts of the input string before and after the match
format_first_only	Only replace the first occurrence of the found pattern

Format specifiers

$0 or $&	The string matching the whole regex
$n	The string matching the n-th capture group, where *n* >= 1
$`	The part of the source string that comes before the substring in *$0*
$'	The part of the source string that comes after the substring in *$0*

Given the regex **(c+)(d+)ef** and the input **abccddefgg**, the format specifiers will denote the following:



Classes

basic_regex<CharT, Traits>(const& regex_str, [flags])
(first_iter, last_iter, [flags])
(const* regex_str, [flags])

Stores a regular expression

Constructor flags

icase	Perform case-insensitive matching
nosubs	Don't store sub-matches in the *match_results* object
optimize	Pay more attention to matching speed instead of the speed of constructing a regex object. Constructing a regex object with this flag can be much slower. Use only when you really need to speed up the matching
collate	Make character ranges locale sensitive

Methods

operator=/assign	Assign a different regular expression
flags	Return a copy of flags passed to the ctor
getloc	Get the locale
imbue	Set the locale
mark_count	Return the number of marked sub-expressions
swap	Swap with another regex object

Typedefs

regex	basic_regex<char>
wregex	basic_regex<wchar_t>

sub_match<BidirectionalIter>

Stores a sequence of characters matched by a capture group

Data members

first	Iterator pointing to the start of the submatch
second	Iterator pointing to the end of the submatch
matched	True if the object describes a submatch

Methods

length	Length of the submatch string
str/ operator str_type	Convert to string type
compare	Compare matched subsequence

Typedefs

csub_match	sub_match<const char*>
wcsub_match	sub_match<const wchar_t*>
ssub_match	sub_match<std::string::const_iterator>
wssub_match	sub_match<std::wstring::const_iterator>

match_results<BidirectionalIter, Alloc>

Holds the results of a regex match

Methods

operator=	Assign another match results object
get_allocator	Return the allocator
ready	Return true if result state is fully established
empty	Return true if *size()* == 0
size	Return 1 + the number of marked sub-expressions
max_size	The max possible number of sub_match elements
format	Produce an output sequence using a format string
swap	Swap with another match_results object
length	The length of a given submatch
position	Distance from start of input to given submatch
str	Convert specified submatch to string type
operator[]	Return a reference to the given sub_match object
prefix	A reference to the sub_match object representing the substrsing of the input before the match
suffix	A reference to the sub_match object for the rest of the input after the match
begin/cbegin	Start iterator that enumerates submatches
end/cend	End iterator that enumerates submatches

Typedefs

smatch	match_results<string::const_iterator>
wsmatch	match_results<wstring::const_iterator>
cmatch	match_results<const char*>
wcmatch	match_results<const wchar_t*>

Iterators

regex_iterator

Uses *regex_search* to iterate over regex matches in the input string

Typedefs

sregex_iterator	regex_iterator<string::const_iterator>
wsregex_iterator	regex_iterator<wstring::const_iterator>
cregex_iterator	regex_iterator<const char*>
wcregex_iterator	regex_iterator<const wchar_t*>

regex_token_iterator

Iterates over matches or submatches in the input string

Typedefs

sregex_token_iterator, wsregex_token_iterator, cregex_token_iterator and wcregex_token_iterator defined similarly to the typedefs for *regex_iterator*