| **字符** | **含义** |
| --- | --- |
| [\](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-backslash) | 匹配将依照下列规则：  在非特殊字符之前的反斜杠表示下一个字符是特殊的，不能从字面上解释。例如，没有前面'\'的'b'通常匹配小写'b'，无论它们出现在哪里。如果加了'\',这个字符变成了一个特殊意义的字符，意思是匹配一个[字符边界](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions#note)。  反斜杠也可以将其后的特殊字符，转义为字面量。例如，模式 /a\*/ 代表会匹配 0 个或者多个 a。相反，模式 /a\\*/ 将 '\*' 的特殊性移除，从而可以匹配像 "a\*" 这样的字符串。  使用 new RegExp("pattern") 的时候不要忘记将 \ 进行转义，因为 \ 在字符串里面也是一个转义字符。 |
| [^](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-caret) | 匹配输入的开始。如果多行标志被设置为true，那么也匹配换行符后紧跟的位置。  例如，/^A/ 并不会匹配 "an A" 中的 'A'，但是会匹配 "An E" 中的 'A'。  当 '^' 作为第一个字符出现在一个字符集合模式时，它将会有不同的含义。[补充字符集合](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions#note) 一节有详细介绍和示例。 |
| [$](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-dollar) | 匹配输入的结束。如果多行标示被设置为true，那么也匹配换行符前的位置。  例如，/t$/ 并不会匹配 "eater" 中的 't'，但是会匹配 "eat" 中的 't'。 |
| [\*](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-asterisk) | 匹配前一个表达式0次或多次。等价于 {0,}。  例如，/bo\*/会匹配 "A ghost boooooed" 中的 'booooo' 和 "A bird warbled" 中的 'b'，但是在 "A goat grunted" 中将不会匹配任何东西。 |
| [+](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-plus) | 匹配前面一个表达式1次或者多次。等价于 {1,}。  例如，/a+/匹配了在 "candy" 中的 'a'，和在 "caaaaaaandy" 中所有的 'a'。 |
| [?](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-questionmark) | 匹配前面一个表达式0次或者1次。等价于 {0,1}。  例如，/e?le?/ 匹配 "angel" 中的 'el'，和 "angle" 中的 'le' 以及"oslo' 中的'l'。  如果**紧跟在任何量词 \*、 +、? 或 {} 的后面**，将会使量词变为**非贪婪**的（匹配尽量少的字符），和缺省使用的**贪婪模式**（匹配尽可能多的字符）正好相反。  例如，对 "123abc" 应用 /\d+/ 将会返回 "123"，如果使用 /\d+?/,那么就只会匹配到 "1"。  还可以运用于先行断言，如本表的 x(?=y) 和 x(?!y) 条目中所述。 |
| [.](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-dot) | （小数点）匹配除换行符之外的任何单个字符。  例如，/.n/将会匹配 "nay, an apple is on the tree" 中的 'an' 和 'on'，但是不会匹配 'nay'。 |
| [(x)](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-capturing-parentheses) | 匹配 'x' 并且记住匹配项，就像下面的例子展示的那样。括号被称为 *捕获括号*。  模式/(foo) (bar) \1 \2/中的 '(foo)' 和 '(bar)' 匹配并记住字符串 "foo bar foo bar" 中前两个单词。模式中的 \1 和 \2 匹配字符串的后两个单词。注意 \1、\2、\n 是用在正则表达式的匹配环节。在正则表达式的替换环节，则要使用像 $1、$2、$n 这样的语法，例如，'bar foo'.replace( /(...) (...)/, '$2 $1' )。 |
| [(?:x)](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-non-capturing-parentheses) | 匹配 'x' 但是不记住匹配项。这种叫作非捕获括号，使得你能够定义为与正则表达式运算符一起使用的子表达式。来看示例表达式 /(?:foo){1,2}/。如果表达式是 /foo{1,2}/，{1,2}将只对 ‘foo’ 的最后一个字符 ’o‘ 生效。如果使用非捕获括号，则{1,2}会匹配整个 ‘foo’ 单词。 |
| [x(?=y)](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-lookahead) | 匹配'x'仅仅当'x'后面跟着'y'.这种叫做正向肯定查找。  例如，/Jack(?=Sprat)/会匹配到'Jack'仅仅当它后面跟着'Sprat'。/Jack(?=Sprat|Frost)/匹配‘Jack’仅仅当它后面跟着'Sprat'或者是‘Frost’。但是‘Sprat’和‘Frost’都不是匹配结果的一部分。 |
| [x(?!y)](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-negated-look-ahead) | 匹配'x'仅仅当'x'后面不跟着'y',这个叫做正向否定查找。  例如，/\d+(?!\.)/匹配一个数字仅仅当这个数字后面没有跟小数点的时候。正则表达式/\d+(?!\.)/.exec("3.141")匹配‘141’但是不是‘3.141’ |
| [x|y](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-or) | 匹配‘x’或者‘y’。  例如，/igreen|red/匹配“green apple”中的‘green’和“red apple”中的‘red’ |
| [{n}](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-quantifier) | n是一个正整数，匹配了前面一个字符刚好发生了n次。  比如，/a{2}/不会匹配“candy”中的'a',但是会匹配“caandy”中所有的a，以及“caaandy”中的前两个'a'。 |
| [{n,m}](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-quantifier-range) | n 和 m 都是整数。匹配前面的字符至少n次，最多m次。如果 n 或者 m 的值是0， 这个值被忽略。  例如，/a{1, 3}/ 并不匹配“cndy”中的任意字符，匹配“candy”中的a，匹配“caandy”中的前两个a，也匹配“caaaaaaandy”中的前三个a。注意，当匹配”caaaaaaandy“时，匹配的值是“aaa”，即使原始的字符串中有更多的a。 |
| [[xyz]](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-character-set) | 一个字符集合。匹配方括号的中任意字符，包括[转义序列](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Grammar_and_types)。你可以使用破折号（-）来指定一个字符范围。对于点（.）和星号（\*）这样的特殊符号在一个字符集中没有特殊的意义。他们不必进行转义，不过转义也是起作用的。  例如，[abcd] 和[a-d]是一样的。他们都匹配"brisket"中得‘b’,也都匹配“city”中的‘c’。/[a-z.]+/ 和/[\w.]+/都匹配“test.i.ng”中的所有字符。 |
| [[^xyz]](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-negated-character-set) | 一个反向字符集。也就是说， 它匹配任何没有包含在方括号中的字符。你可以使用破折号（-）来指定一个字符范围。任何普通字符在这里都是起作用的。  例如，[^abc] 和 [^a-c] 是一样的。他们匹配"brisket"中的‘r’，也匹配“chop”中的‘h’。 |
| [[\b]](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-backspace) | 匹配一个退格(U+0008)。（不要和\b混淆了。） |
| [\b](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-word-boundary) | 匹配一个词的边界。一个词的边界就是一个词不被另外一个“字”字符跟随的位置或者没有其他“字”字符在其前面的位置。注意，一个匹配的词的边界并不包含在匹配的内容中。换句话说，一个匹配的词的边界的内容的长度是0。（不要和[\b]混淆了）  例子：  /\bm/匹配“moon”中得‘m’；  /oo\b/并不匹配"moon"中得'oo'，因为'oo'被一个“字”字符'n'紧跟着。  /oon\b/匹配"moon"中得'oon'，因为'oon'是这个字符串的结束部分。这样他没有被一个“字”字符紧跟着。  /\w\b\w/将不能匹配任何字符串，因为在一个单词中间的字符永远也不可能同时满足没有“字”字符跟随和有“字”字符跟随两种情况。  **注意:** JavaScript的正则表达式引擎将[特定的字符集](http://www.ecma-international.org/ecma-262/5.1/#sec-15.10.2.6)定义为“字”字符。不在该集合中的任何字符都被认为是一个断词。这组字符相当有限：它只包括大写和小写的罗马字母，十进制数字和下划线字符。不幸的是，重要的字符，例如“é”或“ü”，被视为断词。 |
| [\B](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-non-word-boundary) | 匹配一个非单词边界。他匹配一个前后字符都是相同类型的位置：都是“字”字符或者都不是“字”字符。一个字符串的开始和结尾都被认为不是“字”字符，或者空字符串。  例如，/\B../匹配"noonday"中的'oo', 而/y\B../匹配"possibly yesterday"中的’ye‘ |
| [\c](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-control)*[X](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-control)* | 当X是处于A到Z之间的字符的时候，匹配字符串中的一个控制符。  例如，/\cM/ 匹配字符串中的 control-M (U+000D)。 |
| [\d](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-digit) | 匹配一个数字。  等价于[0-9]。  例如， /\d/ 或者 /[0-9]/ 匹配"B2 is the suite number."中的'2'。 |
| [\D](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-non-digit) | 匹配一个非数字字符。  等价于[^0-9]。  例如， /\D/ 或者 /[^0-9]/ 匹配"B2 is the suite number."中的'B' 。 |
| [\f](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-form-feed) | 匹配一个换页符 (U+000C)。 |
| [\n](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-line-feed) | 匹配一个换行符 (U+000A)。 |
| [\r](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-carriage-return) | 匹配一个回车符 (U+000D)。 |
| [\s](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-white-space) | 匹配一个空白字符，包括空格、制表符、换页符和换行符。  等价于[ \f\n\r\t\v\u00a0\u1680\u180e\u2000-\u200a\u2028\u2029\u202f\u205f\u3000\ufeff]。  例如, /\s\w\*/ 匹配"foo bar."中的' bar'。 |
| [\S](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-non-white-space) | 匹配一个非空白字符。  等价于[^ \f\n\r\t\v\u00a0\u1680\u180e\u2000-\u200a\u2028\u2029\u202f\u205f\u3000\ufeff]。  例如， /\S()\*/ 匹配"foo bar."中的'foo'。 |
| [\t](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-tab) | 匹配一个水平制表符 (U+0009)。 |
| [\v](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-vertical-tab) | 匹配一个垂直制表符 (U+000B)。 |
| [\w](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-word) | 匹配一个单字字符（字母、数字或者下划线）。  等价于[A-Za-z0-9\_]。  例如, /\w/ 匹配 "apple," 中的 'a'，"$5.28,"中的 '5' 和 "3D." 中的 '3'。 |
| [\W](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-non-word) | 匹配一个非单字字符。  等价于[^A-Za-z0-9\_]。  例如, /\W/ 或者 /[^A-Za-z0-9\_]/ 匹配 "50%." 中的 '%'。 |
| [\](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-backreference)*[n](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-backreference)* | 在正则表达式中，它返回最后的第n个子捕获匹配的子字符串(捕获的数目以左括号计数)。  比如 /apple(,)\sorange\1/ 匹配"apple, orange, cherry, peach."中的'apple, orange,' 。 |
| [\0](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-null) | 匹配 NULL (U+0000) 字符， 不要在这后面跟其它小数，因为 \0<digits> 是一个八进制转义序列。 |
| [\xhh](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-hex-escape) | 与代码 hh 匹配字符（两个十六进制数字） |
| [\uhhhh](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions" \l "special-unicode-escape) | 与代码 hhhh 匹配字符（四个十六进制数字）。 |