<http://eloquentjavascript.net/1st_edition/chapter10.html>

<https://www.w3schools.com/js/js_regexp.asp>

RegEx is a search pattern used to describe what is being looked for.

The syntax is /pattern/modifiers;

Var patt = /hello/I;

The words beween forward slashes is the pattern to be searched for

I is the modifier, suggesting the case be case-insensitive

There are two string methods:

Search() retursn the position of the match

Replace() returns a modified string where the pattern is replaced

var str = “hello, you’ve arrived”;

var n = str.search(/hello/i);

n is 0

search() also accepts a string str.search(“hello”);

replace()

var str = "Visit Microsoft!";  
var res = str.replace("Microsoft", "W3Schools");

Regular Expression Modifiers

**Modifiers** can be used to perform case-insensitive more global searches:

|  |  |
| --- | --- |
| **Modifier** | **Description** |
| i | Perform case-insensitive matching |
| g | Perform a global match (find all matches rather than stopping after the first match) |
| m | Perform multiline matching |

Regular Expression Patterns

**Brackets** are used to find a range of characters:

|  |  |
| --- | --- |
| **Expression** | **Description** |
| [abc] | Find any of the characters between the brackets |
| [0-9] | Find any of the digits between the brackets |
| (x|y) | Find any of the alternatives separated with | |

**Metacharacters** are characters with a special meaning:

|  |  |
| --- | --- |
| **Metacharacter** | **Description** |
| \d | Find a digit |
| \s | Find a whitespace character |
| \b | Find a match at the beginning or at the end of a word |
| \uxxxx | Find the Unicode character specified by the hexadecimal number xxxx |

**Quantifiers** define quantities:

|  |  |
| --- | --- |
| **Quantifier** | **Description** |
| n+ | Matches any string that contains at least one *n* |
| n\* | Matches any string that contains zero or more occurrences of *n* |
| n? | Matches any string that contains zero or one occurrences of *n* |

Using the RegExp Object

In JavaScript, the RegExp object is a regular expression object with predefined properties and methods.

Using test()

The test() method is a RegExp expression method.

It searches a string for a pattern, and returns true or false, depending on the result.

The following example searches a string for the character "e":

Example

var patt = /e/;  
patt.test("The best things in life are free!");

Since there is an "e" in the string, the output of the code above will be:

true

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_regexp_test)

You don't have to put the regular expression in a variable first. The two lines above can be shortened to one:

/e/.test("The best things in life are free!");

Using exec()

The exec() method is a RegExp expression method.

It searches a string for a specified pattern, and returns the found text.

If no match is found, it returns *null.*

The following example searches a string for the character "e":

Example 1

/e/.exec("The best things in life are free!");

Since there is an "e" in the string, the output of the code above will be:

e

Just like strings get written between quotes, regular expression patterns get written between slashes (/). This means that slashes inside the expression have to be escaped.

var slash = /\//;

show("AC/DC".search(slash));