JavaScript Regular Expressions

What Is a Regular Expression?

A regular expression is a sequence of characters that forms a **search pattern**. 일련의 문자. 형성하다. 검색 패턴

The search pattern can be used for text search and text replace operations. 텍스트 대체 동작

When you search for data in a text, you can use this search pattern to describe what you are searching for.

A regular expression can be a single character, or a more complicated pattern.

Regular expressions can be used to perform all types of **text search** and **text replace** operations. 수행하다

Syntax

/*pattern*/*modifiers*;

Example

var patt = /w3schools/i;

Example explained:

**/w3schools/i**  is a regular expression.

**w3schools**  is a pattern (to be used in a search).

**i**  is a modifier (modifies the search to be case-insensitive).

//대소문자 수정 무감각한

Using String Methods

In JavaScript, regular expressions are often used with the two **string methods**: search() and replace().

**The search() method** uses an expression to search for a match, and returns the position of the match.

**The replace() method** returns a modified string where the pattern is replaced. 교체

Using String search() With a Regular Expression

Example

Use a regular expression to do a case-insensitive search for "w3schools" in a string:

var str = "Visit W3Schools";  
var n = str.search(/w3schools/i);

The result in n will be:

6

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

Using String search() With String

The search method will also accept a string as search argument. 인자

The string argument will be converted to a regular expression:

Example

Use a string to do a search for "W3schools" in a string:

var str = "Visit W3Schools!";  
var n = str.search("W3Schools");

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

Use String replace() With a Regular Expression

Example

Use a case insensitive regular expression to replace Microsoft with W3Schools in a string: 무감각한

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

The result in res will be:

Visit W3Schools!

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

Using String replace() With a String

The replace() method will also accept a string as search argument:

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

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

Did You Notice?

|  |  |
| --- | --- |
|  | Regular expression arguments (instead of string arguments) can be used  in the methods above. Regular expressions can make your search much more powerful  (case insensitive for example). |

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. 미리정의된

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. ~에 의존하다. 따라

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 »](http://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://단축

/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

[Try it yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_regexp_exec)

Complete RegExp Reference

For a complete reference, go to our [Complete JavaScript RegExp Reference](http://www.w3schools.com/jsref/jsref_obj_regexp.asp).

The reference contains descriptions and examples of all RegExp properties and methods