JavaScript RegExp Patterns

Back to JS page

Table of Contents

- JavaScript RegExp Patterns
  - Full RegExp Flag Reference
    - Example 1
  - Full Character Classes Reference
    - Example 2
  - Full Metacharacter Reference
    - Example 3
  - Full RegExp Assertions Reference
    - Example 4
  - RexExp Quantifiers
    - Example 5
  - Document
  - Reference

Full RegExp Flag Reference

All RegExp flags:

Flag	Description
`/g`	Global - matches all occurrences
`/i`	Insensitive - case insensitive
`/m`	Multiline - ^ and $ match line breaks
`/s`	DotAll - `.` matches newlines
`/y`	Sticky - matches from lastIndex
`/u`	Unicode - proper Unicode
`/v`	Unicode sets

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript RegExp Patterns</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<h4>Full RegExp Flag Reference</h4>
<p id="demo"></p>

<script>
const text = "JavaScript JAVASCRIPT javascript";

let output = "Text: " + text + "<br><br>";
output += "flags: " + /javascript/gi.flags + "<br>";
output += "match(/javascript/gi): " + text.match(/javascript/gi) + "<br><br>";

output += "<b>All flags:</b><br>";
output += "/g global: " + /a/g.flags + "<br>";
output += "/i insensitive<br>/m multiline<br>";
output += "/s dotAll<br>/y sticky<br>/u unicode<br>/v unicodeSets";

document.getElementById("demo").innerHTML = output;
</script>

</body>
</html>
```

Example 1

Result View Example

Full Character Classes Reference

Character classes match characters in brackets:

Class	Description
`[abc]`	Any character in the set
`[^abc]`	Any character NOT in the set
`[0-9]`	Any digit from 0 to 9
`[A-Z]`	Any uppercase letter
`[a-z]`	Any lowercase letter
`[A-Za-z]`	Any letter

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript RegExp Patterns</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<h4>Full Character Classes Reference</h4>
<p id="demo"></p>

<script>
const text = "Hello World 123!";

let output = "Text: " + text + "<br><br>";
output += "[A-Z] uppercase: " + text.match(/[A-Z]/g) + "<br>";
output += "[a-z] lowercase: " + text.match(/[a-z]/g) + "<br>";
output += "[0-9] digits: " + text.match(/[0-9]/g) + "<br>";
output += "[^a-z] not lowercase: " + text.match(/[^a-z]/g);

document.getElementById("demo").innerHTML = output;
</script>

</body>
</html>
```

Example 2

Result View Example

Full Metacharacter Reference

Metacharacters with special meanings:

Meta	Description
`\d`	Digit (0-9)
`\D`	Non-digit
`\w`	Word character
`\W`	Non-word character
`\s`	Whitespace
`\S`	Non-whitespace
`.`	Any character (except newline)
`\b`	Word boundary

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript RegExp Patterns</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<h4>Full Metacharacter Reference</h4>
<p id="demo"></p>

<script>
const text = "Test 123";

let output = "Text: " + text + "<br><br>";
output += "\\d digits: " + text.match(/\d/g) + "<br>";
output += "\\D non-digits: " + text.match(/\D/g) + "<br>";
output += "\\w+ words: " + text.match(/\w+/g) + "<br>";
output += "\\s spaces: " + text.match(/\s/g).length + "<br>";
output += ". any chars: " + text.match(/./g).length;

document.getElementById("demo").innerHTML = output;
</script>

</body>
</html>
```

Example 3

Result View Example

Full RegExp Assertions Reference

Assertions check positions:

Assertion	Description
`^`	Start of string
`$`	End of string
`\b`	Word boundary
`x(?=y)`	Positive lookahead
`x(?!y)`	Negative lookahead
`(?<=y)x`	Positive lookbehind
`(?<!y)x`	Negative lookbehind

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript RegExp Patterns</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<h4>Full RegExp Assertions Reference</h4>
<p id="demo"></p>

<script>
const text = "Hello Hello";

let output = "Text: " + text + "<br><br>";
output += "^Hello start: " + text.match(/^Hello/) + "<br>";
output += "Hello$ end: " + text.match(/Hello$/) + "<br>";
output += "Hello(?= Hello) lookahead: " + text.match(/Hello(?= Hello)/) + "<br>";
output += "(?<=Hello )Hello lookbehind: " + text.match(/(?<=Hello )Hello/);

document.getElementById("demo").innerHTML = output;
</script>

</body>
</html>
```

Example 4

Result View Example

RexExp Quantifiers

Quantifiers specify repetition:

Quantifier	Description
`+`	1 or more
`*`	0 or more
`?`	0 or 1
`{n}`	Exactly n
`{n,m}`	Between n and m
`{n,}`	n or more

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript RegExp Patterns</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<h4>RexExp Quantifiers</h4>
<p id="demo"></p>

<script>
const text = "100 1000 10000";

let output = "Text: " + text + "<br><br>";
output += "\\d+ 1+ digits: " + text.match(/\d+/g) + "<br>";
output += "1\\d? optional: " + text.match(/1\d?/g) + "<br>";
output += "\\d{4} exactly 4: " + text.match(/\d{4}/g) + "<br>";
output += "\\d{3,} 3+ digits: " + text.match(/\d{3,}/g);

document.getElementById("demo").innerHTML = output;
</script>

</body>
</html>
```

Example 5

Result View Example

Document

Document in project

You can Download PDF file.

Reference

- W3Schools JavaScript RegExp Patterns