# Javascript Regular Expressions

- description : Javascript Regular Expressions
- author : 오션
- email : shlim@repia.com
- lastupdate : 2021-04-22

### The source of this article

Javascript Regular Expressions

정규식(Regular Expression, 정규 표현식)은 검색 패턴을 형성하는 연속된 문자와 기호입니다.

검색 패턴은 텍스트 검색 및 텍스트 replace 작업에 사용할 수 있습니다.

## What is a Regular Expression?

정규식(Regular Expression, 정규 표현식)은 검색 패턴을 형성하는 연속된 문자와 기호입니다.

텍스트에서 데이터를 검색할 때, 정규식 검색 패턴을 사용하여 찾고 있는 것을 나타낼 수 있습니다.

정규식은 단일 문자이거나 더 복잡한 패턴이 될 수 있습니다.

정규식을 사용하여 모든 유형의 **텍스트 검색(text search)** 및 **텍스트 바꾸기(text replace)** 작업을 할 수 있습니다.

## Syntax

```
/pattern/modifiers;
```

### Example

```
var patt = /w3schools/i;
```

예제 설명:

**/w3schools/i**는 정규식(정규표현식)입니다.

**w3schools**는 (검색에서 사용되는) 패턴입니다.

**i**는 (검색이 대소문자를 구분하지 않도록 수정하는) 변경자(modifier)입니다.

# Using String Methods

JavaScript에서, 정규식은 종종 `search()` 및 `replace()`의 두 가지 **문자열 메서드**와 함께 사용됩니다.

`search()` 메서드는 표현식을 사용하여 일치 항목을 검색하고, 일치 항목의 위치를 반환합니다.

`replace()` 메서드는 패턴이 대체된 수정된 문자열을 반환합니다.

# Using String search() With a String

`search()` 메서드는 문자열에서 지정된 값을 검색하고 일치 항목의 위치를 반환합니다.

### Example

문자열에[서 “W3schools”를 검색하기 위해 string을 사용합니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript String Methods</h2>
  <p>Search a string for "W3schools", and display the position of the
match.</p>
  <p id="demo"></p>   <!-- 6 -->
  <script>
    let str = "Visit W3Schools!";
    let n = str.search("W3Schools");  /* W3Schools가 시작되는 인덱스를 반환? */
    document.getElementById("demo").innerHTML = n;
  </script>
</body>
</html>
```

# Using String search() With a Regular Expression

### Example

정규식을 사용하여 문자열에서 “w3schools”를 대소문자를 구분하지 않으며 검색합니다.

```
<!DOCTYPE html>
```

```
<html>
<body>
  <h2>JavaScript Regular Expressions</h2>
  <p>Search a string for "w3schools", and display the position of the
match:</p>
  <p id="demo"></p>   <!-- 6 -->
  <script>
    let str = "Visit W3Schools!";
    let n = str.search(/w3Schools/i); /* 대소문자 구별 없이 검색하고 시작 인덱스를 반
환? */
    document.getElementById("demo").innerHTML = n;
  </script>
</body>
</html>
```

## Using String replace() With a String

replace() 메서드는 지정된 값을 문자열의 다른 값으로 바꿉니다.

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript String Methods</h2>
  <p>Replace "Microsoft" with "W3Schools" in the paragraph below:</p>
  <button onclick="myFunction()">Try it</button>
  <p id="demo">Please visit Microsoft!</p>   <!-- Please visit W3Schools! -
->
  <script>
    function myFunction() {
      let str = document.getElementById("demo").innerHTML;
      let txt = str.replace("Microsoft", "W3Schools");
      document.getElementById("demo").innerHTML = txt;
    }
  </script>
</body>
</html>
```

## Use String replace() With a Regular Expression

대소문자를 구분하지 않는 정규식(a case insensitive regular expression)을 사용하여 Microsoft를 문자열에서 W3Schools로 바꿉니다.

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

```
  <h2>JavaScript Regular Expressions</h2>
  <p>Replace "Microsoft" with "W3Schools" in the paragraph below:</p>
  <button onclick="myFunction()">Try it</button>
  <p id="demo">Please visit Microsoft and Microsoft!</p>
  <script>
    function myFunction() {
      let str = document.getElementById("demo").innerHTML;
      let txt = str.replace(/microsoft/i, "W3Schools");
      document.getElementById("demo").innerHTML = txt;
    }
  </script>
</body>
</html>
```

# Did You Notice?

위의 메서드에서, 정규식 인수(문자열 인수 대신)를 사용할 수 있습니다.
정규식은 검색을 훨씬 더 강력하게 만들 수 있습니다 (예 : 대소 문자를 구분하지 않음).

# Regular Expression Modifiers

**변경자(Modifiers)**를 사용하여 대소문자를 구분하지 않는 전역검색을 할 수 있습니다.

| Modifier | Description |
|---|---|
| i | 대소문자를 구분하지 않는 매칭을 합니다. |
| g | 전역 매칭을 합니다(첫 번째 매칭 이후에 중단되지 않고 모든 매칭을 찾습니다) |
| m | 여러 줄의 매칭을 합니다. |

## Example of i modifier

```
<script>
    function myFunction() {
      let str = "Visit W3Schools";
      let patt1 = /w3schools/i; /* 대소문자 구분 없이 w3schools를 변수 patt1에 대입 */
      let result = str.match(patt1);  /* 변수str에서 변수patt1(대소문자 구분 없이 w3schools와 일치하는 것을 변수 result에 대입) */
      document.getElementById("demo").innerHTML = result; /* 변수 result를 id="demo"를 가진 p요소에 출력 */
    }
  </script>
```

### Example of g modifier

```
<script>
    function myFunction() {
      let str = "Is this all there is?";
      let patt1 = /is/g;  /* 대소문자 구분없이 검색 */
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result;
    }
```

### Example of m modifier

```
  <script>
    function myFunction() {
      let str = "\nIs th\nis it?";
      let patt1 = /^is/m;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result;  /* is */
    }
  </script>
```

## Regular Expression Patterns

괄호는 문자의 범위를 찾는 데 사용합니다:

| Expression | Description |
|---|---|
| [abc] | 대괄호 사이에 있는 문자와 동일한 문자를 찾습니다. |
| [0-9] | 대괄호 사이에 있는 숫자를 찾습니다. |
| (x\|y) | \|로 구분된 alternatives를 찾습니다. |

### Example of [abc] pattern

```
<script>
    function myFunction() {
      let str = "Is this all there is?";
      let patt1 = /[h]/g;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* h,h */
    }
</script>
```

### Example of [0-9] pattern

```
  <script>
```

```
    function myFunction() {
      let str = "123456789";
      let patt1 = /[1-4]/g; /* 변수 str에서 1~4까지의 숫자를 변수 patt1에 대입 */
      var result = str.match(patt1);  /* 변수 patt1과 일치하는 것을 변수 str에서 찾아서 변수 result에 대입 */
      document.getElementById("demo").innerHTML = result; /* 1,2,3,4 */
    }
  </script>
```

### Example of (x|y) pattern

```
  <script>
    function myFunction() {
      let str = "re, green, red, green, gren, gr, blue, yellow";
      let patt1 = /(red|green)/g;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* green,red,green */
    }
  </script>
```

## Metacharacters are characters with a special meaning:

Metacharacter(메타문자) : 다른 문자의 정보를 운반하기 위해 프로그램 소스나 데이터에 끼워 넣는 문자

| Metacharacter | Description |
|---|---|
| \d | 숫자(digit)를 찾습니다. digit: 0에서 9까지의 10개 숫자를 나타내는 기호 |
| \s | 공백문자 (whitespace character)를 찾습니다 |
| \b | \ bWORD와 같이 단어의 시작 부분에서 일치 항목을 찾거나 또는 WORD \ b처럼 단어의 끝 부분에서 일치 항목을 찾습니다 |
| \uxxx | 16 진수 xxxx로 지정된 유니 코드 문자를 찾습니다. |

### Example of Metachatacters ( \d )

```
  <script>
    function myFunction() {
      let str = "Give 100%!";
      let patt1 = /\d/g;  /* 숫자를 찾는 \d 정규식을 변수 patt1에 대입 */
      let result = str.match(patt1);  /* 변수 str에서 변수 patt1과 일치하는 것을 변수 result에 대입 */
      document.getElementById("demo").innerHTML = result; /* 1,0,0 */
    }
  </script>
```

## Example of Metacharacters ( \s )

```
  <script>
    function myFunction() {
      let str = "Is this all there is?";
      let patt1 = /\s/g;  /* 공백 (whitespace)를 검색 */
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* [" ", " ", " ",
" "] */
      console.log(result, result.length);
    }
  </script>
```

## Exmaple of Metacharacters ( \b )

```
  <script>
    let str = "HELLO, LOOK AT YOU!";
    let patt1 = /\bLO/; /* LO로 시작하는 단어를 검색 */
    let result = str.search(patt1);
    document.getElementById("demo").innerHTML = result; /* Found in
position: 7 */
    console.log(result, result.length); /* 7 undefined */
  </script>
```

## Exmaple of Metacharacters ( \b )

```
  <script>
    let str = "HELLO, LOOK AT YOU";
    let patt1 = /LO\b/; /* 단어의 끝 부분에 LO가 들어가는 것을 검색 */
    let result = str.search(patt1);
    document.getElementById("demo").innerHTML = result; /* Found in
position: 3 */
  </script>
```

## Example of Metacharacters ( \uxxxx )

```
  <script>
    function myFunction() {
      let str = "Visit W3Schools. Hello World!";
      let patt1 = /\u0057/g;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* W,W */
    }
  </script>
```

# Quantifiers define quantities

| Quantifier | Description |
|---|---|
| n+ | 최소 하나의 n을 포함하는 모든 문자열과 일치합니다. |
| n* | n이 0개 이상 포함된 모든 문자열과 일치합니다. |
| n? | n이 0개 또는 1개 포함된 모든 문자열과 일치합니다. |

### Example of Quantifiers ( n+ )

```
  <script>
    function myFunction() {
      let str = "Hellooo World! Hello W3Schools!";
      let patt1 = /o+/g;  /* 각 단어에서 o가 들어간 만큼 표시 */
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* ooo,o,o,oo */
    }
  </script>
```

### Example of Quantifiers ( n* )

```
  <script>
    function myFunction() {
      let str = "Hellooo World! Hello W3Schools!";
      let patt1 = /lo*/g;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* l,looo,l,l,lo,l
*/
    }
  </script>
```

### Example of Quantifiers ( n? )

```
  <script>
    function myFunction() {
      let str = "1, 100 or 1000?";
      let patt1 = /10?/g;
      let result = str.match(patt1);
      document.getElementById("demo").innerHTML = result; /* 1,10,10 */
    }
  </script>
```

# Using the RegExp Object

JavaScript에서 RegExp 오브젝트는 미리 정의된 속성 및 메서드가 있는 정규식 오브젝트입니다.

# Using test()

test() 메서드는 RegExp(정규식) 메서드입니다.

문자열에서 패턴을 검색하고, 결과에 따라 true 또는 false를 반환합니다.

다음 예제는 문자열에서 "e" 문자를 검색합니다.

### Example

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

문자열에 "e"가 있으므로, 상기 코드의 출력값은 true입니다.

먼저 정규식을 변수에 넣을 필요가 없습니다. 위의 두 줄을 아래와 같이 한 줄로 줄일 수 있습니다.

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

```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Regular Expressions</h2>
  <p>Search for an "e" in the next paragraph:</p>
  <p id="p01">The best things in life are free!</p>
  <p id="demo"></p>
  <script>
    text = document.getElementById("p01").innerHTML;
    document.getElementById("demo").innerHTML = /e/.test(text);
  </script>
</body>
</html>
```

# Using exec()

exec() 메서드는 RegExp 정규식 메서드입니다.

문자열에서 지정된 패턴을 검색하고, 찾은 텍스트를 오브젝트로 반환합니다.

일치하는 항목이 없으면, 빈 (null) 개체를 반환합니다.

다음 예제는 문자열에서 “e” 문자를 검색합니다.

### Example

```
  <script>
    let obj = /e/.exec("The best things in life are free!");
    document.getElementById("demo").innerHTML = "Found " + obj[0] + " in
position " + obj.index + " in the text: " + obj.input;
    /* Found e in position 2 in the text: The best things in life are free!
*/
  </script>
```

## Complete RegExp Reference

전체 참조를 보려면 Complete JavaScript RegExp Reference로 이동하십시오.

모든 RegExp 속성 및 메서드에 대한 설명과 예제가 포함되어 있습니다.

오션, Javascript, Regular, Expressions

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
**http://125.132.25.164/dokuwiki/doku.php?id=wiki:javascript:javascript_note:js_regexp&rev=1619082864**

Last update: **2022/03/10 19:52**