JavaScript RegExp Reference

RegExp Object

- A regular expression is an object that describes a pattern of characters.
- Regular expressions are used to perform pattern-matching and "search-and-replace" functions on text.
- Regular expressions can make your search much more powerful (case insensitive for example).

Syntax

/pattern/modifiers;

Example

var v1 = /hello/i;

Example explained:

/hello/i is a regular expression.

hello is a pattern (to be used in a search).

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

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

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.

Ques:Search a string for "university", and display the position of the match

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>

<script>
var str = "Visit the university!";
var n = str.search("university");
document.getElementById("demo").innerHTML = n;
</script>

</body>
</html>
```

Output:

10

Another example

```
<html>

<body>


<p id="demo"></p>

<script>
var str = "Visit the university!";
var n = str.search("visit");
document.getElementById("demo").innerHTML = n;
</script>

</body>
</html>
```

Output: -1(because search is case sensitive)

Using String search() With a Regular Expression

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

```
<html>

<body>


<p id="demo"></p>

<script>
var str = "Visit the University!";
var n = str.search(/visit/i);
document.getElementById("demo").innerHTML = n;
</script>

</body>
</html>
```

Output: 0

Using String replace() With a String

Replace “school" with “University" in string paragraph below:

```
<html>
<body>

<p id="demo">Please visit the school!</p>
<script>

   var str = document.getElementById("demo").innerHTML;
   var txt = str.replace("school","University");
   document.getElementById("demo").innerHTML = txt;

</script></body></html>
```

Output:

Please visit the University!

Use String replace() With a Regular Expression

Example

Use a case insensitive regular expression to replace school with University in given string:

```
<html>
<body>

<p id="demo">Please visit my SchooL!</p>
<script>

   var str = document.getElementById("demo").innerHTML;
   var txt = str.replace(/school/i,"University");
   document.getElementById("demo").innerHTML = txt;

</script></body></html>
```

output

Please visit my University !

match() Method

match() method searches a string for a match against a regular expression, and returns the matches

Syntax

string.match(*regexp*)

- If the regular expression does not include the *g* modifier (to perform a *global* search), the match() method will return only the first match in the string.
- This method returns *null* if no match is found.

Perform a global match(g modifier)

Ques:

do a global search for "is" in a string.

```
<html>
<body>
<p id="demo"></p>
<script>
    var str = "Is this all there is?";
    var patt1 = /is/g;
    var result = str.match(patt1);
    document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

is,is

Ques: do a global, case-insensitive search for "is" in a string.

```
<html><body>
<p id="demo"></p>
<script>
  var str = "Is this all there is?";
   var p1 = /is/gi;
   var result = str.match(p1);
   document.getElementById("demo").innerHTML =
   result;

</script></body></html>
```

Output:

Is,is,is

m modifier

- used to perform a multiline match.
- The m modifier treat beginning (^) and end ($) characters to match the beginning or end of **each line** of a string (delimited by \n or \r), rather than just the beginning or end of the string.

Ques: Do a multiline search for "is" at the beginning of each line in a string.

```
<html>
<body>
<p id="demo"></p>
<script>
var str = "\nIs th\nis it?";
    var p1 = /^is/m;
    var result = str.match(p1);
    document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output

is

Ques: Write code to do a **global, case-insensitive, multiline** search for "is" at the beginning of each line in a string

```
<html><body>
<p id="demo"></p>
<script>
   var str = "\nIs th\nis h\nis?";
   var p1 = /^is/gmi;
   var result = str.match(p1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

Is,is,is

Ques: do a **global, multiline** search for "is" at the end of each line in a string

```
<html>
<body>

<p id="demo"></p>
<script>

   var str = "Is\nthis\nhis\n?";
   var p1 = /is$/gm;
   var result = str.match(p1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

is,is

Brackets

Brackets are used to find a range of characters:

Expression	Description
[abc]	Find any character between the brackets
[^abc]	Find any character NOT between the brackets
[0-9]	Find any character between the brackets (any digit)
[^0-9]	Find any character NOT between the brackets (any non-digit)
(x|y)	Find any of the alternatives specified

[abc]

- The [abc] expression is used to find any character between the brackets.
- The characters inside the brackets can be any characters or span of characters:
- [abcde..] - Any character between the brackets
- [A-Z] - Any character from uppercase A to uppercase Z
- [a-z] - Any character from lowercase a to lowercase z
- [A-z]- Any character from uppercase A to lowercase z

Ques: do a global search for the character "h" in a string.

```
<html>
<body>
<p id="demo"></p>
<script>
   var str = "Is this all there is?";
   var patt1 = /[h]/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

h,h

Example: JavaScript RegExp [^abc] Expression

Ques: to do a global search for characters NOT inside the brackets [h] in a string.

```
<html><body>
<p id="demo"></p>
<script>
var str = "Is this all there is?";
   var p1 = /[^h]/g;
   var result = str.match(p1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

I,s, ,t,i,s, ,a,l,l, ,t,e,r,e, ,i,s,?

Example of [0-9]

Ques: do a global search for the numbers 1 to 4 in a string

```
<html><body>
<p id="demo"></p>
<script>
   var str = "123456789";
   var patt1 = /[1-4]/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

1,2,3,4

Example of [^0-9] Expression

Ques: do a global search for numbers that are NOT 1 to 4 in a string

```
<html>
<body>
<p id="demo"></p>
<script>
   var str = "123456789";
   var patt1 = /[^1-4]/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

5,6,7,8,9

JavaScript RegExp (x|y) Expression

Ques

do a global search for any of the specified alternatives (red|green).

```
<html><body>

<p id="demo"></p>

<script>
   var str = "re, green, red, pink,, green, gren, gr, blue, yellow";
   var patt1 = /(red|green)/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output

green,red,green

Ques: do a **global search** for any of the specified alternatives (0|5|7)

```
<html>
<body>
<p id="demo"></p>
<script>
   var str = "01234567890123456789";
   var patt1 = /(0|5|7)/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

0,5,7,0,5,7

Metacharacters

Metacharacters are characters with a special meaning

Metacharacter	Description
.	Find a single character, except newline or line terminator
\w	Find a word character
\W	Find a non-word character
\d	Find a digit
\D	Find a non-digit character
\s	Find a whitespace character
\S	Find a non-whitespace character
\b	Find a match at the beginning/end of a word, beginning like this: \bHI, end like this: HI\b
\B	Find a match, but not at the beginning/end of a word
\0	Find a NULL character
\n	Find a new line character
\f	Find a form feed character
\r	Find a carriage return character
\t	Find a tab character
\v	Find a vertical tab character

the . Metacharacter:

The . metacharacter is used to find a single character, except newline or other line terminators.

JavaScript RegExp . Metacharacter

Ques: do a global search for "h.t" in a string.

```
<html>
<body>
<p id="demo"></p>
<script>
   var str = "That's hot!";
   var patt1 = /h.t/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

hat,hot

JavaScript RegExp \w Metacharacter

- The \w metacharacter is used to find a word character.
- A word character is a character from a-z, A-Z, 0-9, including the _ (underscore) character.

Ques: do a global search for word characters in a string.

```
<html>
<body>
<p id="demo"></p>
<script>
var str = "Give 100%!";
   var patt1 = / \w/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

G,i,v,e,1,0,0

JavaScript RegExp \W Metacharacter

- The \W metacharacter is used to find a **non-word character.**
- A word character is a character from a-z, A-Z, 0-9, including the _ (underscore) character.

Ques: do a global search for non-word characters in a string.

```
<html><body>
<p id="demo"></p>
<script>
   var str = "Give 100%!";
   var patt1 = / \W/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

,%,!

JavaScript RegExp \d Metacharacter

The \d metacharacter is used to find a digit from 0-9.

Ques: do a global search for digits in a string

```
<html><body>
<p id="demo"></p>

<script>
  var str = "Give 100%!";
   var patt1 = / \d/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

1,0,0

JavaScript RegExp \D Metacharacter

- The \D metacharacter is used to find a non-digit character.

Ques: do a global search for non-digit characters in a string

```
<html><body>
<p id="demo"></p>
<script>
var str = "Give 100%!";
var patt1 = / \D/g;
var result = str.match(patt1);
document.getElementById("demo").innerHTML=result;

</script></body></html>
```

Output:

G,i,v,e, ,%,!

JavaScript RegExp \s Metacharacter

- to find a whitespace character.

A whitespace character can be:

- A space character
- A tab character
- A new line character…. E.t.c

Ques: do a global search for whitespace characters in a string

```
<html>
<body>

<p id="demo"></p>

<script>
   var str = "Is this all there is?";
   var patt1 = / \s/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

, , ,

JavaScript RegExp \S Metacharacter

The \S metacharacter is used to find a non-whitespace character.

Ques: to search for non-whitespace characters in a string

```
<html>
<body>
<p id="demo"></p>
<script>
   var str = "Is this all there is?";
   var patt1 = /\S/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

I,s,t,h,i,s,a,l,l,t,h,e,r,e,i,s,?

JavaScript RegExp \b Metacharacter

- The \b metacharacter is used to find a match at the beginning or end of a word.
- If no match is found, it returns null.

Ques: do a global search for “UNI" at the beginning or end of a word in a string.

```
<html><body>
<p id="demo"></p>
<script>

   var str = "Visit the UNIVERSITY !";
   var patt1 = / \bUNI/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

UNI

JavaScript RegExp \B Metacharacter

- The \B metacharacter is used to find a match not at the beginning or end of a word.

- If no match is found, it returns null.

Ques:do a global search for “UNI" NOT at the beginning or end of a word in a string.

```
<html><body>
<p id="demo"></p>
<script>

   var str = "Visit myUNIVERSITY !";
   var patt1 = / \BUNI/g;
   var result = str.match(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

UNI

JavaScript RegExp \0 Metacharacter

- Search for a NUL character in a string
- \0 returns the position where the NUL character was found. If no match is found, it returns -1.

Ques: write JS code to return the position where the NUL character was found in a string

```
<html><body>
<p id="demo"></p>
<script>
   var str = "Visit Website.\0Learn JavaScript.";
   var patt1 = /\0/;
   var result = str.search(patt1);
   document.getElementById("demo").innerHTML =
   result;

</script></body></html>
```

Output:

14

JavaScript RegExp \n Metacharacter

- The \n character is used to find a newline character.
- \n returns the position where the newline character was found. If no match is found, it returns -1.

Ques: Write JS code to return the position where the newline character was found in a string.

```
<html>
<body>

<p id="demo"></p>
<script>
  var str = "Visit website.\nLearn JavaScript.";
   var patt1 = /\n/;
   var result = str.search(patt1);
   document.getElementById("demo").innerHTML = result;

</script></body></html>
```

Output:

14

\f metacharacter

- The \f metacharacter is used to find a form feed character.
- \f returns the position where the form feed character was found. If no match is found, it returns -1.
- Syntax

/\f/

Ques: to return the position where the form feed character was found in a string.

```
<html>
<body>

<p id="demo"></p>

<script>

  var str = "Visit the class.\fLearn JavaScript.";
  var patt1 = /\f/;
  var result = str.search(patt1);
  document.getElementById("demo").innerHTML = result;

</script>

</body>
</html>
```

Output

16

\r Metacharacter

- The \r metacharacter is used to find a carriage return character.
- \r returns the position where the carriage return character was found. If no match is found, it returns -1.
- Syntax

/ \r/

QUESTION:
write JS code to Search for a carriage return character in a string:

```
<html>
<body>


<p id="demo"></p>

<script>

  var str = "Visit the class.\rLearn JavaScript.";
  var patt1 = /\r/;
  var result = str.search(patt1);
  document.getElementById("demo").innerHTML = result;

</script>

</body>
</html>
```

\t Metacharacter

- The \t metacharacter is used to find a tab character.
- \t returns the position where the tab character was found. If no match is found, it returns -1.

Ques: Write code to Search for a tab character in a string:

```
<html>
<body>


<p id="demo"></p>

<script>

  var str = "Visit the class.\tLearn JavaScript.";
  var patt1 = /\t/;
  var result = str.search(patt1);
  document.getElementById("demo").innerHTML = result;

</script>

</body>
</html>
```

Output:

16

\v Metacharacter

- The \v metacharacter is used to find a vertical tab character.
- \v returns the position where the vertical tab character was found. If no match is found, it returns -1.

Question: Search for a vertical tab character in a string:

```
<html>
<body>

<p id="demo"></p>

<script>

  var str = "Visit the class.\vLearn JavaScript.";
  var patt1 = /\v/;
  var result = str.search(patt1);
  document.getElementById("demo").innerHTML = result;

</script>

</body>
</html>
```

Output:

16