JavaScript String Methods

JavaScript, methods and properties are also available to primitive values, because JavaScript treats primitive values as objects when executing methods and properties.

1. String Length

The length property returns the length of a string:

var name = "Graphic";  
var ln = name.length;

2. Finding a String in a String

indexof() and lastIndexOf() method

The indexOf() method returns the index of (the position of) the first occurrence of a specified text in a string:

JavaScript counts positions from zero.   
0 is the first position in a string, 1 is the second, 2 is the third ...

The lastIndexOf() method returns the index of the **last** occurrence of a specified text in a string:

Both indexOf(), and lastIndexOf() return -1 if the text is not found.

<!DOCTYPE html>

<html>

<body>

<p>Located String Position is</p>

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

<script>

var str = "how are you. you are welcome";

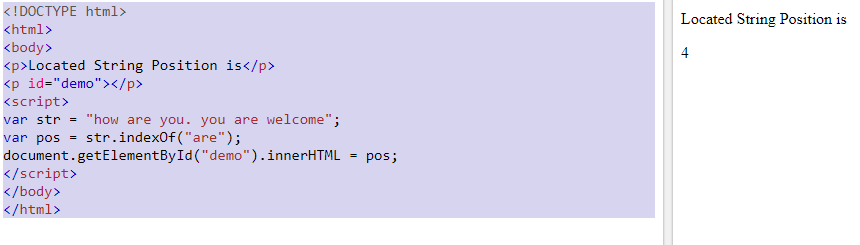
var pos = str.indexOf("are");

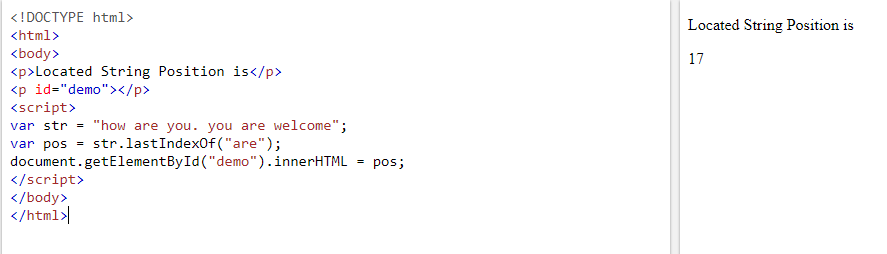
document.getElementById("demo").innerHTML = pos;

</script>

</body>

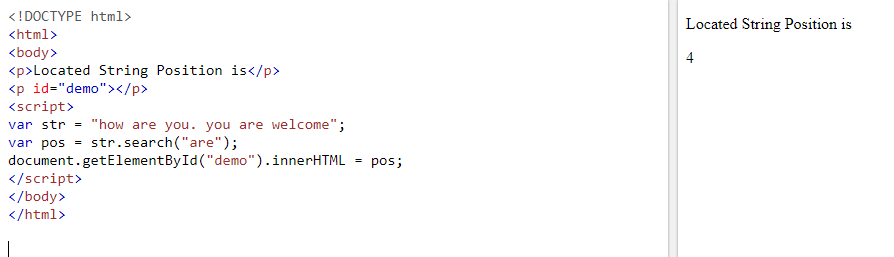
</html>





Searching for a String in a String

The search() method searches a string for a specified value and returns the position of the match:



The two methods, indexOf() and search(), are **equal?**

They accept the same arguments (parameters), and return the same value?

The two methods are **NOT** equal. These are the differences:

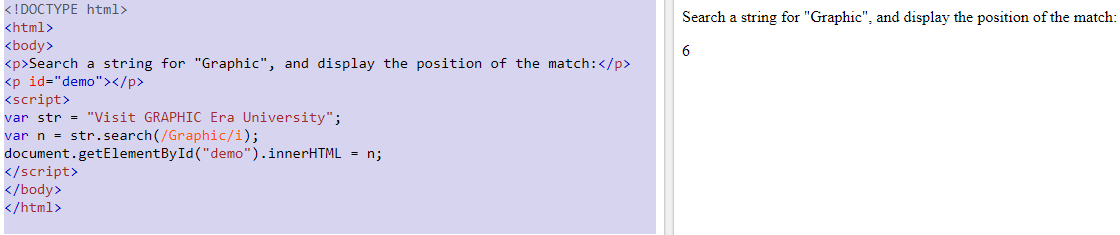
* The search() method cannot take a second start position argument.
* The indexOf() method cannot take powerful search values (regular expressions).

Using String search() With a Regular Expression

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

var str = "Visit GRAPHIC Era University";

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



<!DOCTYPE html>

<html>

<body>

<h2>JavaScript String Methods</h2>

<script>

var str = "How are you. You are Welcome.";

var pos = str.indexOf("are");

while(pos!=-1)

{

document.write(pos+" ");

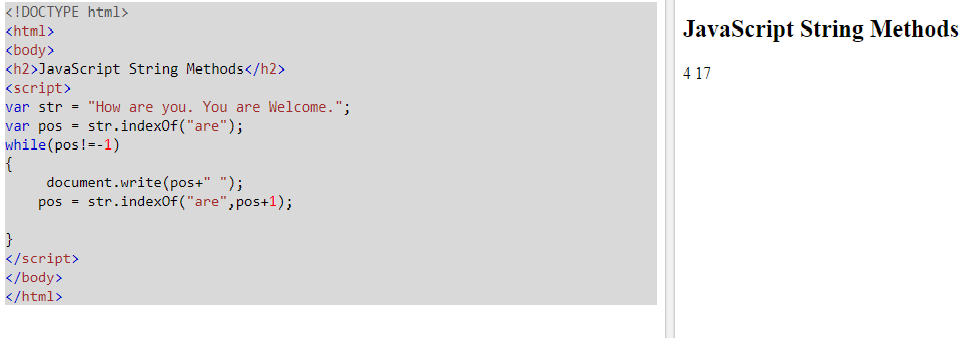
pos = str.indexOf("are",pos+1);

}

</script>

</body>

</html>



3.Extracting String Parts

There are 3 methods for extracting a part of a string:

* slice(start, end)
* substring(start, end)
* substr(start, length)

The slice() Method

slice() extracts a part of a string and returns the extracted part in a new string.

var str = "Apple, Banana, Kiwi";  
var res = str.slice(7, 13);

This example slices out a portion of a string from position 7 to position 12 (13-1):

**The result of res will be:**

Banana

If you omit the second parameter, the method will slice out the rest of the string:

var str = "Apple, Banana, Kiwi";  
var res = str.slice(7);

**The result of res will be:**

Banana, Kiwi

The substring() Method

substring() is similar to slice().

The difference is that substring() cannot accept negative indexes.

### Example

var str = "Apple, Banana, Kiwi";  
var res = str.substring(7, 13);

The result of *res* will be:

Banana

If you omit the second parameter, substring() will slice out the rest of the string.

The substr() Method

substr() is similar to slice().

The difference is that the second parameter specifies the **length** of the extracted part.

### Example

var str = "Apple, Banana, Kiwi";  
var res = str.substr(7, 6);

The result of res will be:

Banana

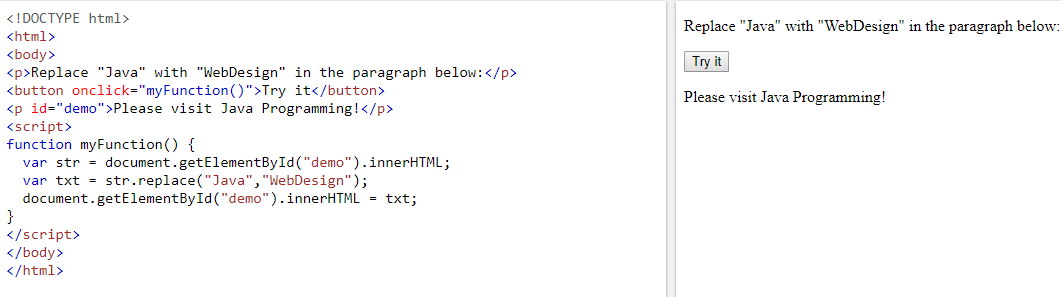
If you omit the second parameter, substr() will slice out the rest of the string.

4.Replacing String Content

The replace() method replaces a specified value with another value in a string:

var str="Please visit Java Programming!"

var n=str.replace("java", "WebDesign");



By default, the replace() method replaces **only the first** match:

To replace case insensitive, use a **regular expression** with an /i flag (insensitive):

str = "Please visit Microsoft!";  
var n = str.replace(/MICROSOFT/i, "WIPRO");

To replace all matches, use a **regular expression** with a /g flag (global match):

### Example

str = "Please visit Microsoft and Microsoft!";  
var n = str.replace(/Microsoft/g, "WIPRO");

To replace all matches, with a /g flag (global match)and an /i flag (insensitive):

var n = str.replace(/Microsoft/gi, "WIPRO");

## 5.Converting to Upper and Lower Case

A string is converted to upper case with toUpperCase()

### Example

var text1 = "Hello World!";       // String  
var text2 = text1.toUpperCase();  // text2 is text1 converted to upper

A string is converted to lower case with toLowerCase()

### Example

var text1 = "Hello World!";       // String  
var text2 = text1.toLowerCase();  // text2 is text1 converted to lower

## 6.The concat() Method

concat() joins two or more strings:

### Example

var text1 = "Hello";  
var text2 = "World";  
var text3 = text1.concat(" ", text2);

The concat() method can be used instead of the plus operator. These two lines do the same:

### Example

var text = "Hello" + " " + "World!";  
var text = "Hello".concat(" ", "World!");

All string methods return a new string. They don't modify the original string.  
Formally said: Strings are immutable: Strings cannot be changed, only replaced.

## 7.String.trim()

The trim() method removes whitespace from both sides of a string:

### Example

var str = "       Hello World!        ";  
alert(str.trim());

## 8.Extracting String Characters

There are 3 methods for extracting string characters:

* charAt(position)
* charCodeAt(position)
* Property access [ ]

## The charAt() Method

The charAt() method returns the character at a specified index (position) in a string:

### Example

var str = "HELLO WORLD";  
str.charAt(0);            // returns H

## The charCodeAt() Method

The charCodeAt() method returns the unicode of the character at a specified index in a string:

The method returns a UTF-16 code (an integer between 0 and 65535).

### Example

var str = "HELLO WORLD";  
  
str.charCodeAt(0);         // returns 72

## Property Access

ECMAScript 5 (2009) allows property access [ ] on strings:

### Example

var str = "HELLO WORLD";  
str[0];                   // returns H

## Converting a String to an Array

A string can be converted to an array with the split() method:

txt.split(",");          // Split on commas  
txt.split(" ");          // Split on spaces  
txt.split("|");   // Split on pipe

**<!DOCTYPE html>**

**<html>**

**<body>**

**<button onclick="myFunction()">Try it</button>**

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

**<script>**

**function myFunction() {**

**var str = "how,are,you";**

**var arr = str.split(",");**

**var i;**

**for(i=0;i<arr.length;i++)**

**{**

**document.write(arr[i]+"</br>");**

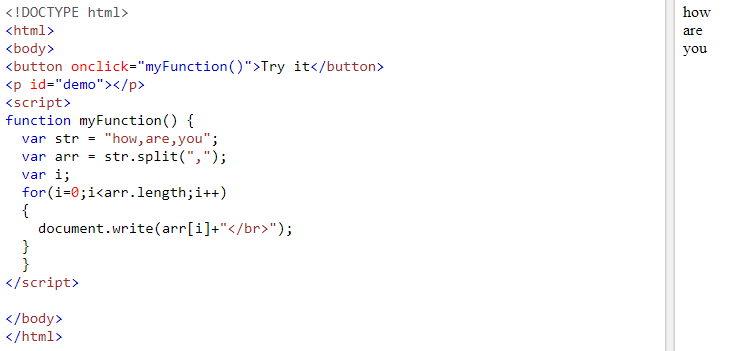
**}**

**}**

**</script>**

**</body>**

**</html>**

****

If the separator is omitted, the returned array will contain the whole string in index [0].

If the separator is "", the returned array will be an array of single characters:

**<!DOCTYPE html>**

**<html>**

**<body>**

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

**<script>**

**var str = "Web Design";**

**var arr = str.split("");**

**var text = "";**

**var i;**

**for (i = 0; i < arr.length; i++) {**

**text += arr[i] + "<br>"**

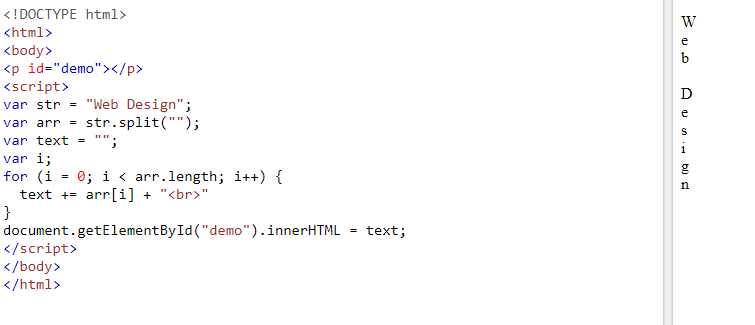
**}**

**document.getElementById("demo").innerHTML = text;**

**</script>**

**</body>**

**</html>**

****