

String

Property Summary

Number	length Gives the length of the string.
--------	---

Method Summary

String	anchor (nameAttribute) returns a copy of the string embedded within an anchor <A> tag set.
String	big () returns a copy of the string embedded within an <BIG> tag set.
String	blink () returns a copy of the string embedded within an <BLINK> tag set.
String	bold () returns a copy of the string embedded within an tag set.
Number	charAt (index) returns a character of the string.
Number	charCodeAt (index) returns a decimal code of the char in the string.
String	concat (string2) returns a string that appends the parameter string to the string.
String	concat (string2, stringN) returns a string that appends the parameter string to the string.
Boolean	equals (other) returns a boolean that checks if the given string is equal to the string
Boolean	equalsIgnoreCase (other) returns a boolean that checks if the given string is equal to the string ignoring case
String	fixed () returns a copy of the string embedded within an anchor <TT> tag set.
String	fontcolor (color) returns a copy of the string embedded within an tag set, the color param is assigned the the color attribute.
String	fontsize (size) returns a copy of the string embedded within an tag set, The size param is set to the SIZE attribute
String	fromCharCode (num) returns a string created by using the specified sequence of Unicode values.
Number	indexOf (searchValue, fromIndex) returns the found index of the given string in string.
String	italics () returns a copy of the string embedded within an <I> tag set
Number	lastIndexOf (searchValue, fromIndex) returns the found index of the given string in string from the end.
String	link (hrefAttribute) returns a copy of the string embedded within an <A> tag set.
Number	localeCompare (otherString)
Array	match (regexp) returns an array of strings within the current string that matches the regexp.
String	replace (regexp, function) returns a new string where the matches of the given regexp are replaced by the return value of the function.
String	replace (regexp, newSubStr) returns a new string where the matches of the given regexp are replaced by newSubStr.
String	replace (substr, function) returns a new string where the first match of the given substr is replaced by the return value of the function.
String	replace (substr, newSubStr) returns a new string where the first match of the given substr is replaced by newSubStr.
Number	search (regexp) returns an index where the first match is found of the regexp
String	slice (beginSlice) returns a substring of the string.
String	slice (beginSlice, endSlice) returns a substring of the string.
String	small () returns a copy of the string embedded within an <SMALL> tag set.
String	split (separator, limit) returns an array of objects whose elements are segments of the current string.

String	strike() returns a copy of the string embedded within an <STRIKE> tag set.
String	sub() returns a copy of the string embedded within an <SUB> tag set.
String	substr(start) returns a substring of the string from the start with the number of chars specified.
String	substr(start, length) returns a substring of the string from the start with the number of chars specified.
String	substring(indexA) Returns a substring of the string from the start index until the end index.
String	substring(indexA, indexB) Returns a substring of the string from the start index until the end index.
String	sup() returns a copy of the string embedded within an <SUP> tag set.
String	toLocaleLowerCase()
String	toLocaleUpperCase()
String	toLowerCase() returns a string with all lowercase letters of the current string.
String	toUpperCase() returns a string with all uppercase letters of the current string.

Property Details

length

Gives the length of the string.

Returns

[Number](#)

Sample

```
string.length;
```

Method Details

anchor

[String](#) **anchor** (nameAttribute)

returns a copy of the string embedded within an anchor <A> tag set.

Parameters

{[String](#)} nameAttribute

Returns

[String](#)

Sample

```
string.anchor();
```

big

[String](#) **big** ()

returns a copy of the string embedded within an <BIG> tag set.

Returns

[String](#)

Sample

```
string.big();
```

blink

[String](#) **blink** ()

returns a copy of the string embedded within an <BLINK> tag set.

Returns

[String](#)

Sample

```
string.blink();
```

bold

[String](#) **bold** ()

returns a copy of the string embedded within an tag set.

Returns

[String](#)

Sample

```
string.bold();
```

charAt

[Number](#) **charAt** (index)

returns a character of the string.

Parameters

{[Number](#)} index

Returns

[Number](#)

Sample

```
string.charAt(integer_position);
```

charCodeAt

[Number](#) **charCodeAt** (index)

returns a decimal code of the char in the string.

Parameters

{[Number](#)} index

Returns

[Number](#)

Sample

```
string.charCodeAt(integer_position);
```

concat

[String](#) **concat** (string2)

returns a string that appends the parameter string to the string.

Parameters

{[String](#)} string2

Returns

[String](#)

Sample

```
string.concat(string);
```

concat

[String](#) **concat** (string2, stringN)

returns a string that appends the parameter string to the string.

Parameters

[{String}](#) string2

[{String}](#) stringN

Returns

[String](#)

Sample

```
string.concat(string);
```

equals

[Boolean](#) **equals** (other)

returns a boolean that checks if the given string is equal to the string

Parameters

[{String}](#) other

Returns

[Boolean](#)

Sample

```
string.equals(string);
```

equalsIgnoreCase

[Boolean](#) **equalsIgnoreCase** (other)

returns a boolean that checks if the given string is equal to the string ignoring case

Parameters

[{String}](#) other

Returns

[Boolean](#)

Sample

```
string.equalsIgnoreCase(string);
```

fixed

[String](#) **fixed** ()

returns a copy of the string embedded within an anchor <TT> tag set.

Returns

[String](#)

Sample

```
string.fixed();
```

fontcolor

[String](#) **fontcolor** (color)

returns a copy of the string embedded within an tag set, the color param is assigned the the color attribute.

Parameters

[{String}](#) color

Returns

[String](#)

Sample

```
string.fontcolor(color);
```

fontsize

[String](#) **fontsize** (size)

returns a copy of the string embedded within an tag set, The size param is set to the SIZE attribute

Parameters

{[Number](#)} size

Returns

[String](#)

Sample

```
string.fontsize(size);
```

fromCharCode

[String](#) **fromCharCode** (num)

returns a string created by using the specified sequence of Unicode values.

Parameters

{[Number](#)...} num

Returns

[String](#)

Sample

```
String.fromCharCode(num)  
// String.fromCharCode(num1,num2,num3)
```

indexOf

[Number](#) **indexOf** (searchValue, fromIndex)

returns the found index of the given string in string.

Parameters

{[String](#)} searchValue

{[Number](#)} fromIndex

Returns

[Number](#)

Sample

```
string.indexOf(string,startPosition);
```

italics

[String](#) **italics** ()

returns a copy of the string embedded within an <I> tag set

Returns

[String](#)

Sample

```
string.italics();
```

lastIndexOf

[Number](#) **lastIndexOf** (searchValue, fromIndex)

returns the found index of the given string in string from the end.

Parameters

[{String}](#) searchValue
[{Number}](#) fromIndex

Returns

[Number](#)

Sample

```
string.lastIndexOf(string, startPosition);
```

link

[String](#) **link** (hrefAttribute)

returns a copy of the string embedded within an <A> tag set.

Parameters

[{String}](#) hrefAttribute

Returns

[String](#)

Sample

```
string.link(url);
```

localeCompare

[Number](#) **localeCompare** (otherString)

Parameters

[{String}](#) otherString

Returns

[Number](#)

Sample

```
var s = "Have a nice day!";  
application.output(s.localeCompare("Hello"));
```

match

[Array](#) **match** (regexp)

returns an array of strings within the current string that matches the regexp.

Parameters

[{RegExp}](#) regexp

Returns

[Array](#)

Sample

```
string.match(regexpr);
```

replace

[String](#) **replace** (regexp, function)

returns a new string where the matches of the given regexp are replaced by the return value of the function.

The function parameter is the function to be invoked to create the new substring (to put in place of the substring received from parameter #1).

Parameters

[{RegExp}](#) regexp
[{Function}](#) function

Returns

[String](#)

Sample

```
//the callback definition
function replacer(match, p1, p2, p3, offset, string){
    // match is the matched substring
    // p1 is non-digits, p2 digits, and p3 non-alphanumerics
    // offset is the offset of the matched substring within the total string being examined
    // string is the total string being examined
    return [p1, p2, p3].join(' - ');
}
// using replace method with replacer callback
newString = "abc12345#$*%".replace(/([^\d]*)(\d*)([^\w]*)/, replacer);
```

replace

String **replace** (regexp, newSubStr)

returns a new string where the matches of the given regexp are replaced by newSubStr.

Parameters

{RegExp} regexp
{String} newSubStr

Returns

String

Sample

```
string.replace(regexp,newSubStr);
//var re = /(\w+)\s(\w+)/;
//var str = "John Smith";
//var newstr = str.replace(re, "$2, $1");
//application.output(newstr);
```

replace

String **replace** (substr, function)

returns a new string where the first match of the given substr is replaced by the return value of the function.

The function parameter is the function to be invoked to create the new substring (to put in place of the substring received from parameter #1).

Parameters

{String} substr
{Function} function

Returns

String

Sample

```
// the callback definition
function replacer(match){
    return match.toUpperCase()
}
// using replace method with replacer callback
var newString = "abc".replace("a", replacer);
```

replace

String **replace** (substr, newSubStr)

returns a new string where the first match of the given substr is replaced by newSubStr.

Parameters

{String} substr
{String} newSubStr

Returns

String

Sample

```
string.replace(substr,newSubStr);
```

search

[Number](#) **search** (regex)

returns an index where the first match is found of the regex

Parameters

[{RegExp}](#) regex

Returns

[Number](#)

Sample

```
string.search(regexpr);
```

slice

[String](#) **slice** (beginSlice)

returns a substring of the string.

Parameters

[{Number}](#) beginSlice

Returns

[String](#)

Sample

```
string.slice(start,end);
```

slice

[String](#) **slice** (beginSlice, endSlice)

returns a substring of the string.

Parameters

[{Number}](#) beginSlice

[{Number}](#) endSlice

Returns

[String](#)

Sample

```
string.slice(start,end);
```

small

[String](#) **small** ()

returns a copy of the string embedded within an <SMALL> tag set.

Returns

[String](#)

Sample

```
string.small();
```

split

[String](#) **split** (separator, limit)

returns an array of objects whose elements are segments of the current string.

Parameters

{[String](#)} separator

{[Number](#)} limit

Returns

[String](#)

Sample

```
string.split(delimiter,limitInteger);
```

strike

[String](#) **strike** ()

returns a copy of the string embedded within an <STRIKE> tag set.

Returns

[String](#)

Sample

```
string.strike();
```

sub

[String](#) **sub** ()

returns a copy of the string embedded within an <SUB> tag set.

Returns

[String](#)

Sample

```
string.sub();
```

substr

[String](#) **substr** (start)

returns a substring of the string from the start with the number of chars specified.

Parameters

{[Number](#)} start

Returns

[String](#)

Sample

```
string.substr(start, number_of_chars);
```

substr

[String](#) **substr** (start, length)

returns a substring of the string from the start with the number of chars specified.

Parameters

{[Number](#)} start

{[Number](#)} length

Returns

[String](#)

Sample

```
string.substr(start, number_of_chars);
```

substring

[String](#) **substring** (indexA)

Returns a substring of the string from the start index until the end index.

Parameters

[{Number}](#) indexA

Returns

[String](#)

Sample

```
string.substring(start, end);
```

substring

[String](#) **substring** (indexA, indexB)

Returns a substring of the string from the start index until the end index.

Parameters

[{Number}](#) indexA

[{Number}](#) indexB

Returns

[String](#)

Sample

```
string.substring(start, end);
```

sup

[String](#) **sup** ()

returns a copy of the string embedded within an <SUP> tag set.

Returns

[String](#)

Sample

```
string.sup();
```

toLocaleLowerCase

[String](#) **toLocaleLowerCase** ()

Returns

[String](#)

Sample

```
var s = "Have a nice day!";  
application.output(s.toLocaleLowerCase());
```

toLocaleUpperCase

[String](#) **toLocaleUpperCase** ()

Returns

[String](#)

Sample

```
var s = "Have a nice day!";  
application.output(s.toLocaleUpperCase());
```

toLowerCase

[String](#) **toLowerCase** ()

returns a string with all lowercase letters of the current string.

Returns

[String](#)

Sample

```
string.toLowerCase();
```

toUpperCase

[String](#) **toUpperCase** ()

returns a string with all uppercase letters of the current string.

Returns

[String](#)

Sample

```
string.toUpperCase();
```