

Array

Property Summary

Number [#length](#)
Get the length of the array.

Method Summary

Array [#concat](#)(value1, [value2], [valueN])
Returns a new array comprised of this array joined with other array(s) and/or value(s).

Boolean [#every](#)(callback, [thisObject])
Runs a function on items in the array while that function is returning true.

Array [#filter](#)(callback, [thisObject])
Runs a function on every item in the array and returns an array of all items for which the function returns true.

void [#forEach](#)(callback, [thisObject])
Runs a function on every item in the array.

Number [#indexOf](#)(searchElement, [fromIndex])
Returns the first index at which a given element can be found in the array, or -1 if it is not present.

String [#join](#)(delimiter)
Puts all elements in the array into a string, separating each element with the specified delimiter

Number [#lastIndexOf](#)(searchElement, [fromIndex])
Returns the last index at which a given element can be found in the array, or -1 if it is not present.

Array [#map](#)(callback, [thisObject])
Runs a function on every item in the array and returns the results in an array.

Object [#pop](#)()
Pops the last string off the array and returns it.

Number [#push](#)(value1, [value2], [valueN])
Mutates an array by appending the given elements and returning the new length of the array.

Array [#reverse](#)()
Puts array elements in reverse order.

Object [#shift](#)()
Decreases array element size by one by shifting the first element off the array and returning it.

Array [#slice](#)(begin, [end])
The slice method creates a new array from a selected section of an array.

Boolean [#some](#)(callback, [thisObject])
Runs a function on items in the array while that function returns false.

Array [#sort](#)([function])
Sorts the array elements in dictionary order or using a compare function passed to the method.

Array [#splice](#)(arrayIndex, length, [value1], [value2], [valueN])
It is used to take elements out of an array and replace them with those specified.

Array [#unshift](#)(value1, value2, valueN)
Places element data at the start of an array.

Property Details

`length`
Get the length of the array.

Returns

Number

Sample

```
array.length
```

Method Details

`concat`

Array [concat](#)(value1, [value2], [valueN])

Returns a new array comprised of this array joined with other array(s) and/or value(s).

Parameters

{Object} value1

{Object} [value2]

{Object} [valueN]

Returns

[Array](#)

Sample

```
array.concat();
```

[every](#)

[Boolean](#) **every**(callback, [thisObject])

Runs a function on items in the array while that function is returning true. It returns true if the function returns true for every item it could visit.

Parameters

[{Function}](#) callback

[{Array}](#) [thisObject]

Returns

[Boolean](#)

Sample

```
function isNumber(value) { return typeof value == 'number'; }
var a1 = [1, 2, 3];
application.output(a1.every(isNumber));
var a2 = [1, '2', 3];
application.output(a2.every(isNumber));
```

[filter](#)

[Array](#) **filter**(callback, [thisObject])

Runs a function on every item in the array and returns an array of all items for which the function returns true.

Parameters

[{Function}](#) callback

[{Array}](#) [thisObject]

Returns

[Array](#)

Sample

```
var a1 = ['a', 10, 'b', 20, 'c', 30];
var a2 = a1.filter(function(item) { return typeof item == 'number'; });
application.output(a2);
```

[forEach](#)

void **forEach**(callback, [thisObject])

Runs a function on every item in the array.

Parameters

[{Function}](#) callback

[{Array}](#) [thisObject]

Returns

void

Sample

```
function printThemOut(params) { application.output(params);}
var a = ['a', 'b', 'c'];
a.forEach(printThemOut);
```

[indexOf](#)

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

Returns the first index at which a given element can be found in the array, or -1 if it is not present.

Parameters

[{Object}](#) searchElement

[{Number}](#) [fromIndex]

Returns

[Number](#)

Sample

```
var a = ['a', 'b', 'a', 'b', 'a'];
application.output(a.indexOf('b'));
application.output(a.indexOf('b', 2));
application.output(a.indexOf('z'));
```

join

String **join**(delimiter)

Puts all elements in the array into a string, separating each element with the specified delimiter

Parameters

{String} delimiter

Returns

String

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
var jwords = words.join(";");
```

lastIndexOf

Number **lastIndexOf**(searchElement, [fromIndex])

Returns the last index at which a given element can be found in the array, or -1 if it is not present. The array is searched backwards, starting at fromIndex.

Parameters

{Object} searchElement

{Number} [fromIndex]

Returns

Number

Sample

```
var a = ['a', 'b', 'c', 'd', 'a', 'b'];
application.output(a.lastIndexOf('b'));
application.output(a.lastIndexOf('b', 4));
application.output(a.lastIndexOf('z'));
```

map

Array **map**(callback, [thisObject])

Runs a function on every item in the array and returns the results in an array.

Parameters

{Object} callback

{Array} [thisObject]

Returns

Array

Sample

```
var a = ['a', 'b', 'c'];
var a2 = a.map(function(item) { return item.toUpperCase(); });
application.output(a2);
```

pop

Object **pop**()

Pops the last string off the array and returns it.

Returns

Object

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
var lastword = words.pop();
```

push

Number **push**(value1, [value2], [valueN])

Mutates an array by appending the given elements and returning the new length of the array.

Parameters

[{Object}](#) value1
[{Object}](#) [value2]
[{Object}](#) [valueN]

Returns

[Number](#)

Sample

```
var words = new Array("limit","lines","finish","complete");
words.push("In","Out");
```

[reverse](#)

[Array](#) **reverse()**

Puts array elements in reverse order.

Returns

[Array](#)

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
words.reverse();
```

[shift](#)

[Object](#) **shift()**

Decreases array element size by one by shifting the first element off the array and returning it.

Returns

[Object](#)

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
words.shift();
```

[slice](#)

[Array](#) **slice**(begin, [end])

The slice method creates a new array from a selected section of an array.

Parameters

[{Object}](#) begin
[{Object}](#) [end]

Returns

[Array](#)

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
var nwords1 = words.slice(3, 5);
```

[some](#)

[Boolean](#) **some**(callback, [thisObject])

Runs a function on items in the array while that function returns false. It returns true if the function returns true for any item it could visit.

Parameters

[{Function}](#) callback
[{Array}](#) [thisObject]

Returns

[Boolean](#)

Sample

```
function isNumber(value) { return typeof value == 'number'; }
var a1 = [1, 2, 3];
application.output(a1.some(isNumber));
var a2 = [1, '2', 3];
application.output(a2.some(isNumber));
```

[sort](#)

[Array](#) **sort**([function])

Sorts the array elements in dictionary order or using a compare function passed to the method.

Parameters

[{Function}](#) [function]

Returns

[Array](#)

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
words.sort();
```

splice

[Array](#) **splice**(arrayIndex, length, [value1], [value2], [valueN])

It is used to take elements out of an array and replace them with those specified.

Parameters

[{Object}](#) arrayIndex

[{Object}](#) length

[{Object}](#) [value1]

[{Object}](#) [value2]

[{Object}](#) [valueN]

Returns

[Array](#)

Sample

```
var words = new Array("limit","lines","finish","complete","In","Out");
var nwords1 = words.splice(3, 2, "done", "On");
```

unshift

[Array](#) **unshift**(value1, value2, valueN)

Places element data at the start of an array.

Parameters

[{Object}](#) value1

[{Object}](#) value2

[{Object}](#) valueN

Returns

[Array](#)

Sample

```
var words = new Array("finish","complete","In","Out");
words.unshift("limit","lines");
```