Telerik Academy

Arrays

Processing Sequences of Elements

JavaScript Fundamentals

Telerik Software Academy

https://telerikacademy.com

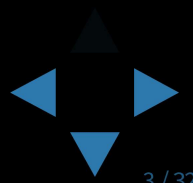
Telerik Academy

Table of Contents

Telerik Academy

Arrays Overview

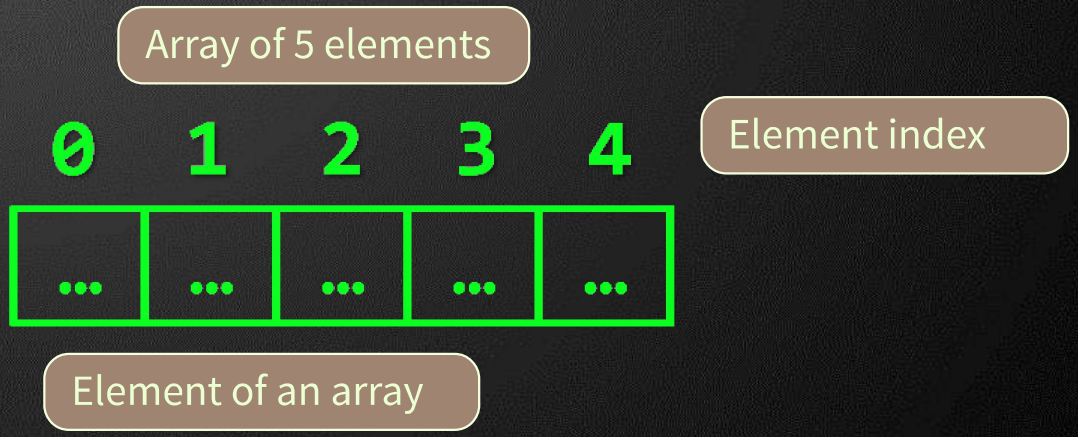
What are arrays? How to use arrays?



Telerik Academy

What are Arrays?

- An array is a sequence of elements
  - The order of the elements is fixed
  - Does not have fixed size
    - Can get the current length (`Array.length`)



Telerik Academy

Declaring Arrays

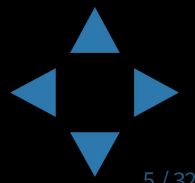
- Declaring an array in JavaScript

```
// Array holding integers
var numbers = [1, 2, 3, 4, 5];

// Array holding strings
var weekDays = ['Monday', 'Tuesday', 'Wednesday',
  'Thursday', 'Friday', 'Saturday', 'Sunday']

// Array of different types
var mixedArr = [1, new Date(), 'hello'];

// Array of arrays (matrix)
var matrix = [
  ['0,0', '0,1', '0,2'],
  ['1,0', '1,1', '1,2'],
  ['2,0', '2,1', '2,2']];
```

Declare and Initialize Arrays

- Initializing an array in JavaScript can be done in three ways:
  - Using `new Array(elements)`:

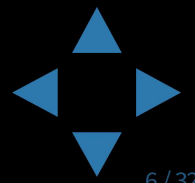
```
var arr = new Array(1, 2, 3, 4, 5);
```

  - Using `new Array(initialLength)`:

```
var arr = new Array(10);
```

  - Using `array literal` (recommended):

```
var arr = [1, 2, 3, 4, 5];
```



Telerik Academy

Creating Arrays

Demo

Telerik Academy

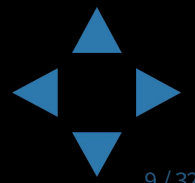
Using arrays

Read and Modify Elements by Index

Telerik Academy

How to Access Array Element?

- Array elements are accessed using the indexer operator: `[]` (square brackets)
  - Array indexer takes element's index as parameter in the range `[0; length-1]`
  - The first element has index `0`
  - The last element has index `length-1`
- Array elements can be retrieved and changed by the `[]` (indexer) operator



Reversing an Array – *Example*

- *Example:* Reversing the elements of an array

```
//always declare var variables on the top of the scope!
var array,
    len,
    reversed,
    i,
    j;

array = [1, 2, 3, 4, 5];
reversed = [];

for (i = 0, len = array.length; i < len; i += 1) {
    j = len - i - 1;
    reversed.push(array[j]);
}
```

Telerik Academy

Reversing an Array

Demo

Telerik Academy

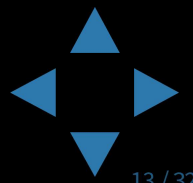
Iterating Arrays

Telerik Academy

Iterating Arrays with `for`

- Use `for` loop to process an array when you need to keep track of the index
- In the loop body use the element at the loop index (`array[index]`):

```
var i, len;
for (i = 0, len = array.length; i < len; i += 1) {
   squares[i] = array[i] * array[i];
}
```

Telerik Academy

Iterating Arrays with for

- *Example:* Printing array of numbers in reversed order:

```
var arr, i, len;
arr = [1, 2, 3, 4, 5];
for (len = arr.length, i = len - 1; i >= 0; i -= 1) {
    console.log(arr[i]);
}
// Result: 5 4 3 2 1
```

- *Example:* Initialize all array elements with their corresponding index number:

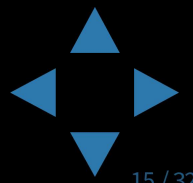
```
var i, len
for (i = 0, len = array.length; i < len; i += 1) {
    array[i] = i;
}
```


Telerik Academy

Iterating Arrays using `for-in`

- How `for-in` loop works?
  - `index` iterates through the indexes of the array
- Used when the indexes are unknown
  - All elements are accessed one by one
  - Order is not guaranteed
  - Works for objects as well

```
var index;
for (index in array) {
  // great code
}
```

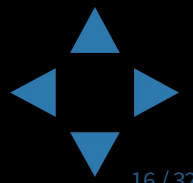
Telerik Academy

Iterating Arrays with `for-in`

- Print all elements of an array of strings:

```
var capitals, i;
capitals = [
    'Sofia',
    'Washington',
    'London',
    'Paris'
];

for (i in capitals) {
    console.log(capitals[i]);
}
```

Telerik Academy

Processing Arrays

Demo

Telerik Academy

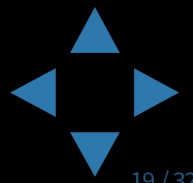
Inserting and Removing Elements from Arrays

`push`, `pop`, `shift`, `unshift`

Telerik Academy

Inserting and Removing Elements from Arrays

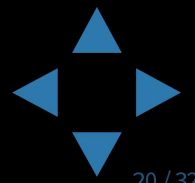
- All arrays in JavaScript are dynamic
  - Their size can be changed at runtime
  - New elements can be inserted to the array
  - Elements can be removed from the array



Telerik Academy

Inserting and Removing Elements from Arrays

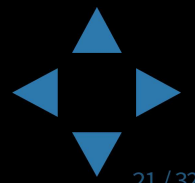
- Methods for array manipulation:
  - `Array#push(element)`
    - Inserts a new element at the tail of the array
  - `Array#pop()`
    - Removes the element at the tail
    - Returns the removed element



Telerik Academy

Inserting and Removing Elements from Arrays

- Methods for array manipulation (cont.)
  - `Array#unshift(element)`
    - Inserts a new element at the head of the array
  - `Array#shift()`
    - Removes the element at the head
    - Returns the remove element



Telerik Academy

Inserting and Removing Elements from Arrays

Demo

Telerik Academy

Array Methods

Reversing, joining, etc...

Telerik Academy

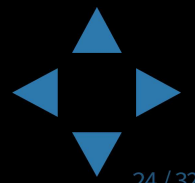
Array Methods

- `Array#reverse()`
  - Reverses the elements of the array
  - Returns a new arrays

```
var items = [1, 2, 3, 4, 5, 6];
var reversed = items.reverse();
//reversed = [6, 5, 4, 3, 2, 1]
```

- `Array#join(separator)`
  - Concatenates the elements with a separator
  - Returns a string

```
var names = ["John", "Jane", "George", "Helen"];
var namesString = names.join(", ");
//namesString = "John, Jane, George, Helen"
```

Telerik Academy

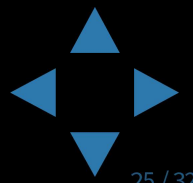
Concatenating Arrays

- `arr1.concat(arr2)`
  - Inserts the elements of `arr2` at the end of `arr1`
  - Returns a new array
  - `arr1` and `arr2` remain unchanged!

```
var arr1 = [1, 2, 3];
var arr2 = ["one", "two", "three"];
var result = arr1.concat(arr2);
//result = [1, 2, 3, "one", "two", "three"]
```

- Adding the elements of an array to other array

```
var arr1 = [1, 2, 3];
var arr2 = ["one", "two", "three"];
[].push.apply(arr1, arr2);
//arr1 = [1, 2, 3, "one", "two", "three"]
```

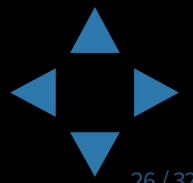



Telerik Academy

Getting Parts of Arrays

- `Array#slice(fromIndex [, toIndex])`
  - Returns a new array
    - A shallow copy of a portion of the array
  - The new array contains the elements from indices `fromIndex` to `to` (excluding `toIndex`)
  - Can be used to clone an array

```
var items = [1, 2, 3, 4, 5];
var part = items.slice(1, 3);
//part = [2, 3]
var clonedItems = items.slice();
```

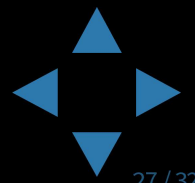


Telerik Academy

Splicing Arrays

- `Array#splice(index, count, elements)`
  - Removes `count` elements, starting from `index` position
  - Adds `elements` at position `index`
  - Returns a new array, containing the removed elements

```
var numbers = [1, 2, 3, 4, 5, 6, 7];
var result = numbers.splice(3, 2, "four", "five", "five.fiv
//result = [4, 5]
//numbers = [1, 2, 3, "four", "five", "five.five", 6, 7];
```

Telerik Academy

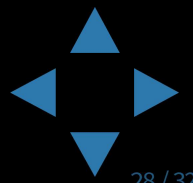
Splicing Arrays

- `Array#splice(index, count, elements)`
  - Example uses:
    - Remove elements from any index of the array:

```
//removes a single element at position index
items.splice(index, 1);
//removes count elements starting from position index
items.splice(index, count);
```

    - Insert elements at any index of the array:

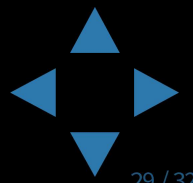
```
//Inserts a single element at position index
items.splice(index, 0, element);
//Inserts many elements starting from position index
items.splice(index, 0, item1, item2, item3);
```



Telerik Academy

Searching in Arrays

- `Array#indexOf(element [, rightOf])`
  - Returns the index of the first match in the array
  - Returns `-1` is the element is not found
- `Array#lastIndexOf(element, [leftOf])`
  - Returns the index of the first match in the array
  - Returns `-1` is the element is not found
- `Array#indexOf()` and `Array#lastIndexOf()` do not work in all browsers
  - Need to be shimmed



Telerik Academy

Other Arrays Functions

- Arrays official documentation:
  - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array
- Checking for array
  - `typeof([1, 2, 3]) -> object`
    - Not working
  - `Array.isArray([1, 2, 3]) -> true`
  - Supported on all modern browsers