
THE //
FLATIRON
SCHOOL

Arrays And Loops

- Creating Arrays
- Array Methods
- Iterating Over Arrays Using Loops

Arrays

Arrays

An array is a list-like object that allows us to store a series of information into a single entity.

Kind of like a bookshelf, where we can store a book in each slot on the shelf. And then retrieve that book by referring to its index number.

Arrays 5

```
//declaring an empty array using the Array constructor.
// var myArr = new Array();

//declaring an empty array using literal notation.
var myArr = [ ];

//Arrays are filled with elements: i.e. myArr3 = [element, anotherElement];
//Elements can be strings, numbers, or boolean.
myArr = ['Hello', , 54.3, true];

//If you leave a blank spot in an array it creates a blank shelf space
(undefined) placeholder.
```

```
myArr = ['Hello', , 54.3, true];

//Array elements can be fetched by their index number (starts from 0).
console.log(myArr[0]); //prints Hello
console.log(myArr[1]); //prints undefined
console.log(myArr[2]); //prints 54.3
console.log(myArr[3]); //prints true

//We can insert new values into any space in the array using the positions index.
myArr[1] = 'Stuff';
```

Arrays 7

Let’s imagine the shelf below represents an array containing books. We will call it myArray for this example.

Array INDEXES

Array INDEXES

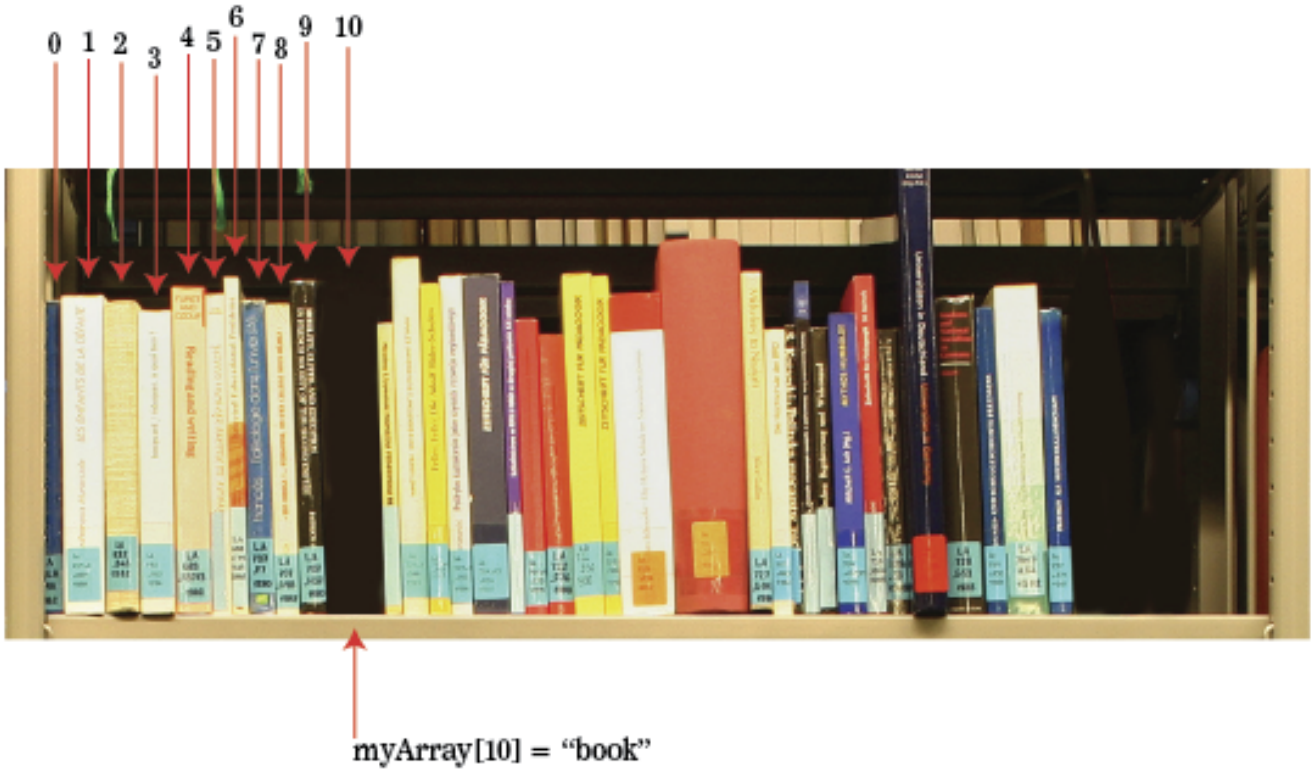
Array INDEXES

Array INDEXES

StorE or ChangE ELEMENT VALUES

Let's say we want to add a new book (element) into the 10th index position since it is currently blank.

StorE or ChangE ELEMENT VALUES



StorE or ChangE ELEMENT VALUES

```
myArr = ['Hello', , 54.3, true];

//Associative arrays allow us to define the index as a string.

myArr['Greeting'] = myArr[0];

//Now we can refer to that specific position by using its association.

console.log(myArr['Greeting']); //prints Hello
```

```
//We can overwrite all the elements of an array simply by giving the array new values. Or passing one array into another.

var fruits = ['Apples', 'Oranges', 'Pears', 'Bananas'];

myArr = fruits;

console.log(myArr); //prints Apples, Oranges, Pears, Bananas
```

```
myArr = ['Apples', 'Oranges', 'Pears', 'Bananas'];

//What if I would like to know how long my array is (how many elements)?

console.log(myArr.length); //prints 4

//To get the last elements index position I can subtract one (remember
indexes start with zero instead of one).

var pos = myArr.length - 1;
console.log(myArr[pos]); //prints Bananas
```

Array Methods

Array Methods

Array methods allow us to further manipulate arrays. For example what if we would like to push a new item up onto the end of myArray?

Array Methods 20

```
myArr = ['Apples', 'Oranges', 'Pears', 'Bananas'];
//We can insert on to the end of an Array simply by using the push method.

myArr.push('Strawberries'); // you can push multiple items onto the end by coma separating if you wish.

console.log(myArr); //prints Apples, Oranges, Pears, Bananas, Strawberries
```

Array Methods

myArray.push("book2");

Array Methods

myArray.push("book2");

What about the opposite? How would I remove the last element off the end of the array?

```
//you can pull the last element off the end using the pop method.
myArr.pop();

console.log(myArr); //prints Apples, Oranges, Pears, Bananas.
//Notice Strawberries is now missing.
```

Array Methods

myArray.pop();

That’s great, but how do I remove a specific element from the array no matter where it is located?

//We can use splice method to insert content at a given position or to remove content from a given position. splice(index, how many elements to remove, optional content to add);

myArr.splice(2, 0, 'Tiger'); //This goes to index position 2 and after it removes 0 (none) and adds new value of 'Tiger'.

console.log(myArr); //prints Apples, Oranges, Tiger, Pears, Bananas where previously was Apples, Oranges, Pears, Bananas. Tiger has been inserted After Oranges an the others followered have been bumped forward 1 index.

//lets say we needed to remove the second to last item, but we do not know how long the array is... We can use negative numbers to go to end of array and index from end to beginning.

```
myArr.splice(-2, 1, "Lion"); //replace Pears with Lion


console.log(myArr); //prints Apples, Oranges, Tiger, Lion, Bananas
```

```
//Let's remove Lion from the 3rd index position using splice.

myArr.splice(3,1); //Go to index position 3 and remove 1.

console.log(myArr); //prints Apples, Oranges, Tiger, Bananas
//Lion has been removed.
```

Array Methods

myArray.splice(10, 1);

Array Methods

Array Methods 32

For many more Array methods see:

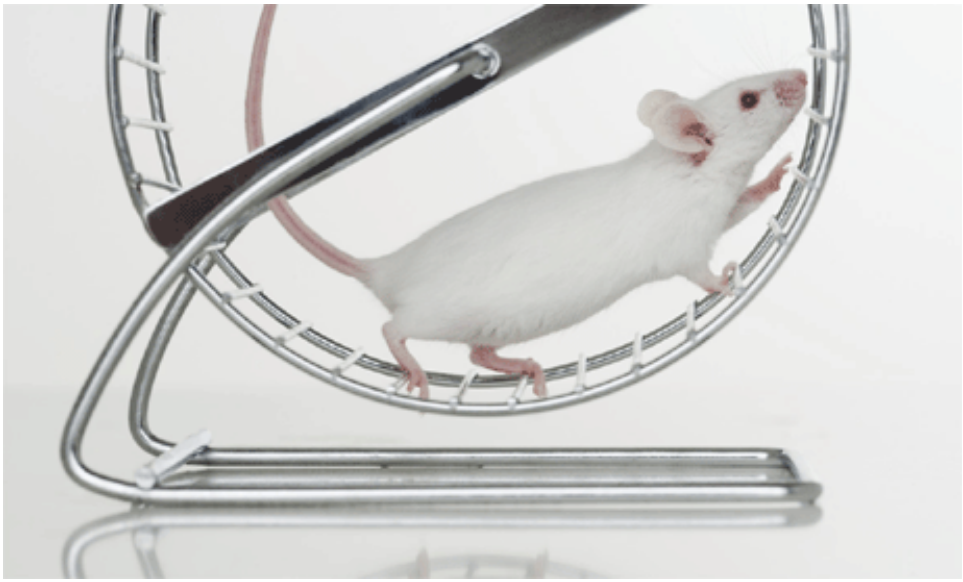
https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Global_Objects/Array

Iterating over Arrays

Iterating over Arrays

A loop is a set of commands that executes repeatedly until a specified condition is met.

Imagine a mouse running on a wheel and the mouse is trained to repeat an action each time he makes one rotation. Before he starts each rotation, he checks if the condition is still true. When the condition ceases to be true he stops running.



FOR Loop 35

```
//A for loop repeats until a specified condition evaluates to false.

//SYNTAX: for ([initialExpression]; [condition]; [incrementExpression]) {statement}

var vegetables = ['Broccoli','Peas','Carrots'];

for (var i = 0; i < vegetables.length; i++) {
    console.log(vegetables[i]);
}

//prints Broccoli, Peas, Carrots
```

Do…while Loop 36

```
//A while statement executes its statements as long as a specified condition evaluates to true.

//SYNTAX: do {statement} while (condition);

var cars = ['Corvette','Mustang','Porsche'];
var i = 0;

do {
    console.log(cars[i]);
    i += 1;
}
while (i < cars.length);    //prints Corvette, Mustang, Porsche
```

While Loop

```
//SYNTAX: while (condition){statement}

var fish = ['Snapper', 'Tuna', 'Salmon'];
var i = 0;

while (i < fish.length) {
    console.log(fish[i]);
    i += 1;
}

//prints Snapper, Tuna, Salmon
```

Discover more on loop statements at:

https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Statements

Does jQuery provide any means to iterate over arrays?

jQUery EACH method 40

```
//jQuery each method

//SYNTAX: .each( function(index, value) ) { statement }

var numbers = [5, 10, 15, 20, 25, 30 , 35, 40, 45, 50];

$.each(numbers, function(i, value){
    console.log(i+' '+value);
});

//'each' iterates over each element in the array and allows us to display or make us of the elements index number and or value. One nice thing about using 'each' is that it automatically calculates the length of the array and the count position for us so there is not need for an 'i' variable or use of '.length' method.
```

Is there a way to filter through an array and only capture results that match our search criteria?

jQuery Grep method 42

```
//jQuery grep method for filtering an array

//SYNTAX: jQuery.grep( array, function(elementOfArray, indexInArray) [, invert] )

var bigNumbers = $.grep(numbers, function(value) {
    return value > 25;
});

console.log(bigNumbers);

//prints only the big numbers 30, 35, 40, 45, 50
```