Codesmith

CSPrep

Day 4

Overview

POD

Review a complex higher order function (that makes use of another higher order function)

- map with forEach

Understanding docs on mdn - map

Project: drawing insights from big data using higher order functions

Using map with forEach

```
91  function map(array, instructions) {
92    const output = [];
93    function runOnEachElement(el) {
94      output.push(instructions(el));
95    }
96    forEach(array, runOnEachElement);
97    return output;
98  }
99
100 function forEach(list, instructionsForEach) {
101   for (let i = 0; i < list.length; i++) {
102     instructionsForEach(list[i]);
103   }
104 }
105
106 function multiplyByTwo(num) {
107   return num * 2;
108 }
109
110 const doubles = map([1,2,3], multiplyByTwo);
111
```

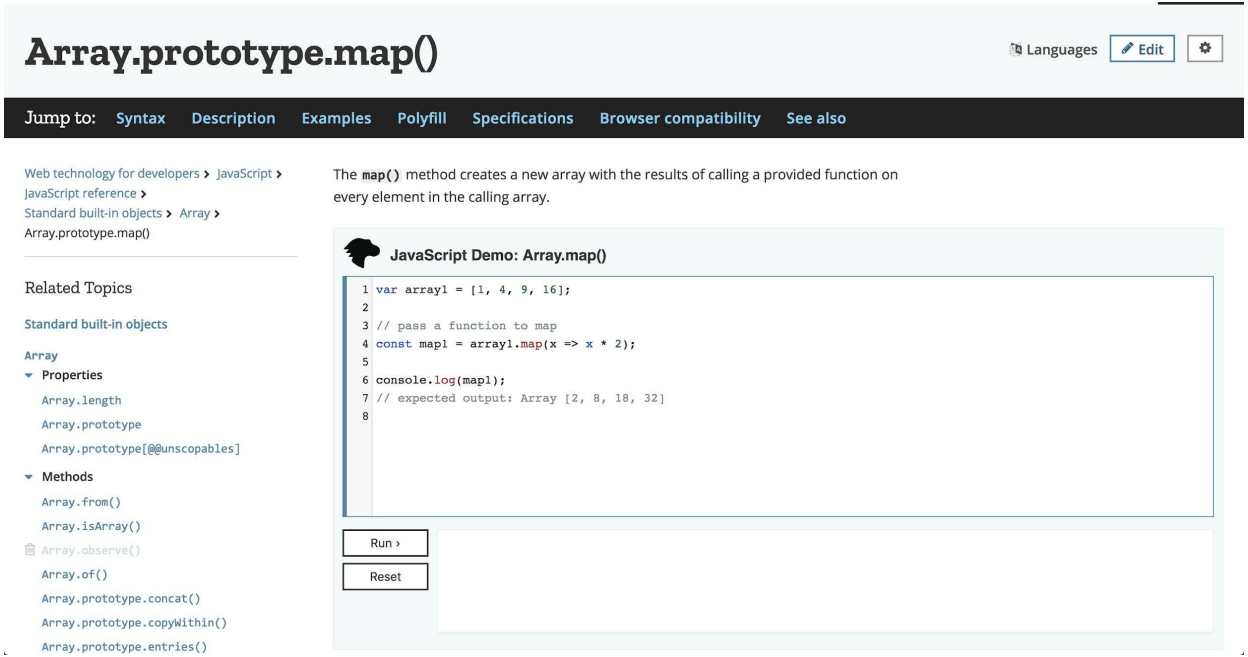
```
function map(array, instructions) {
  const output = [];
  function runOnEachElement(el) {
    output.push(instructions(el));
  }
  forEach(array, runOnEachElement);
  return output;
}

function forEach(list, instructionsForEach) {
  for (let i = 0; i < list.length; i++) {
    instructionsForEach(list[i]);
  }
}

function multiplyByTwo(num) {
  return num * 2;
}

const doubles = map([1,2,3], multiplyByTwo);
```

Understanding docs: .map on MDN



Array.prototype.map()

Languages | Edit

Jump to: Syntax | Description | Examples | Polyfill | Specifications | Browser compatibility | See also

Web technology for developers › JavaScript › JavaScript reference › Standard built-in objects › Array › Array.prototype.map()

The `map()` method creates a new array with the results of calling a provided function on every element in the calling array.

Related Topics

Standard built-in objects

Array

- Properties
  - Array.length
  - Array.prototype
  - Array.prototype[@@unscopables]
- Methods
  - Array.from()
  - Array.isArray()
  - Array.observe()
  - Array.of()
  - Array.prototype.concat()
  - Array.prototype.copyWithin()
  - Array.prototype.entries()

JavaScript Demo: Array.map()

```
1 var array1 = [1, 4, 9, 16];
2
3 // pass a function to map
4 const map1 = array1.map(x => x * 2);
5
6 console.log(map1);
7 // expected output: Array [2, 8, 18, 32]
8
```

Run ›

Reset

Project: Big data with higher order functions

The data you will be working with is historical bitcoin data from 2013 to 2015

Work through the initial prompts in CSBin - try to use higher order functions like forEach, map, filter, reduce…. When in doubt, use a for loop first then refactor!

When you're ready, ask for the bonus prompts.

http://csbin.io/cs-prep-big-data