ARRAYS

LEVEL UP YOUR JAVASCRIPT

BY CODY BARRUS & RYAN EWING

WITH ES2015 CAME 10 NEW ARRAY METHODS!

STATIC METHODS

- `Array.from`
- `Array.of`

Array.from

```
Array.from(arguments)
```

RETURNS AN ARRAY FROM AN ARRAY LIKE OBJECT. SUCH AS:

```
> arguments
> document.querySelectorAll('div')
> $('div')
```

ESSENTIALLY REPLACES THE NEED FOR:

```
function cast ()
  return Array.prototype.slice.call(arguments)
}
cast('a', 'b')
```

OR

```
function cast ()
  return [].slice.call(arguments)
}
```

Array.from CNT'D

THREE ARGUMENTS

- input - AN ARRAYLIKE OR ITTERABLE
- map - A MAPPING FUNCTION EXECUTED ON EACH ITEM OF input
- context - this BINDING WHEN CALLING map

```
function typesOf () {
  return Array.from(arguments, value => typeof value)
}
typesOf(null, [], NaN)
// <- ['object', 'object', 'number']
```

ES2015 ALTERNATIVE

- FOR ARGUMENTS. YOU COULD ALSO USE REST PARAMETERS WHICH MIGHT BE EASIER TO READ.
- HOWEVER. FOR OTHER USES. LIKE JQUERY. `Array.from` IS A LOGICAL CHOICE

```
function typesOf (...all) {
  return all.map(value => typeof value)
}
typesOf(null, [], NaN)
// ['object', 'object', 'number']
```

```
new Array()
// []

Array.of()
// []

new Array(1)
// [undefined x 1]

Array.of(1)
// [1]

new Array(3)
// [undefined x 3]

Array.of(3)
// [3]
```

```
new Array(1, 2)
// [1, 2]

Array.of(1, 2)
// [1, 2]

new Array(-1)
// RangeError: Invalid array length

Array.of(-1)
// [-1]
```

PROTOTYPES

- ARRAY.PROTOTYPE.COPYWITHIN*
- ARRAY.PROTOTYPE.FILL
- ARRAY.PROTOTYPE.FIND
- ARRAY.PROTOTYPE.FINDINDEX
- ARRAY.PROTOTYPE.KEYS
- ARRAY.PROTOTYPE.VALUES

Array.prototype.fill

- FILLS ALL THINGS IN AN ARRAY WITH THE VALUE PROVIDED

```
[1, 2, 3].fill('tacos')
// ['tacos', 'tacos', 'tacos']
```

- CAN SPECIFY A START AND END INDEX

```
new Array(5).fill(0, 0, 3)
// [0, 0, 0, undefined x 2]
```

Array.prototype.find

- FINDS AND RETURNS THE FIRST ITEM THAT MATCHES `callback(item, i, array)` FOR AN ARRAY
  - RETURNS `undefined` IF NOTHING IS FOUND

```
[1, 2, 3, 4, 5].find(item => item > 2)
// 3
[1, 2, 3, 4, 5].find((item, i) => i === 3)
// 4
```

Array.prototype.findIndex

- LIKE FIND, BUT RETURNS THE INDEX OF MATCHING ELEMENT
  - RETUNS -1 IF NOTHING IS FOUND

```
[1, 2, 3, 4, 5].find(item => item > 2)
// <- 2
[1, 2, 3, 4, 5].find((item, i) => i === 3)
// <- 3
```

Array.prototype.keys

- RETURNS AN ITERATOR THAT YIELDS A SEQUENCE HOLDING THE KEYS OF THE ARRAY.
- CAN USE WITH ES6 SPREAD OPERATOR, `for..of`, OR BY CALLING `.next()`

```
for (let key of [1, 2, 3].keys()) {
  console.log(key)
  // 0
  // 1
  // 2
}
```

Array.prototype.values

- LIKE `keys`, BUT RETURNS AN ITERATOR WITH A SEQUENCE OF VALUES

```
for (let key of [1, 2, 3].values()) {
  console.log(key)
  // 1
  // 2
  // 3
}
```

Array.prototype.entries

- LIKE `keys` AND `values`. BUT RETURNS AN ITERATOR WITH A SEQUENCE OF KEY, VALUE PAIRS

```
for (let key of [1, 2, 3].values()) {
  console.log(key)
  // [0, 1]
  // [1, 2]
  // [2, 3]
}
```

Array.prototype[Symbol.iterator]

- EXACTLY THE SAME AS `Array.prototype.values`