ES6 cheat cheat

Dr. Sergio Inzunza

Objects

Basic object literal

```
var dog = {

  // object properties
  name: 'Fido',
  age: 1,

  // object methods
  walk: function () { },

  bark() { }
}
```

Objects

Access members

```
// dot notation
console.log(dog.age);

// bracket notation
console.log(dog['age']);
```

Array map

Transform each element

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

JavaScript Demo: Array.map()

```
1 const 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
```

Array filter

Select some elements

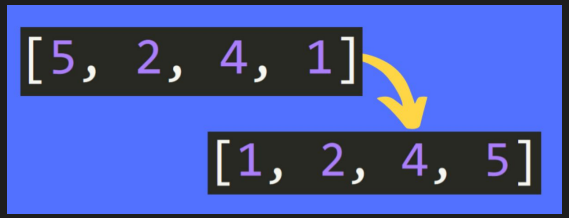
The `filter()` method **creates a new array** with all elements that pass the test implemented by the provided function.

JavaScript Demo: Array.filter()

```
1 const words = ['spray', 'limit', 'elite', 'exuberant', 'destruction', 'present'];
2
3 const result = words.filter(word => word.length > 6);
4
5 console.log(result);
6 // expected output: Array ["exuberant", "destruction", "present"]
7
```

Array sort

Sort the elements



The `sort()` method sorts the elements of an array *in place* and returns the sorted array. The default sort order is ascending, built upon converting the elements into strings, then comparing their sequences of UTF-16 code units values.

The time and space complexity of the sort cannot be guaranteed as it depends on the implementation.

JavaScript Demo: Array.sort()

```
1  const months = ['March', 'Jan', 'Feb', 'Dec'];
2  months.sort();
3  console.log(months);
4  // expected output: Array ["Dec", "Feb", "Jan", "March"]
5
6  const array1 = [1, 30, 4, 21, 100000];
7  array1.sort();
8  console.log(array1);
9  // expected output: Array [1, 100000, 21, 30, 4]
10
```

Object destructuring

Unpack array/object

```
const { prop } = object;
```

```
const { prop1, prop2, ..., propN }
   = object;
```

Spread operator

Clone array/object