Destructuring

Destructuring allows us to extract values from arrays and objects (even deeply nested) and store them in variables with a more convenient syntax.

Destructuring Arrays

```
var arr = [1, 2, 3, 4];
var a = arr[0];
var b = arr[1];
var c = arr[2];
var d = arr[3];
```

```
let [a, b, c, d] = [1, 2, 3, 4];

console.log(a); // 1
console.log(b); // 2
```

Destructuring Objects

```
var luke = { occupation: 'jedi', father: 'anakin' };
var occupation = luke.occupation; // 'jedi'
var father = luke.father; // 'anakin'
```

```
let luke = { occupation: 'jedi', father: 'anakin' };
let {occupation, father} = luke;

console.log(occupation); // 'jedi'
console.log(father); // 'anakin'
```

Maps

Maps is a much needed data structure in JavaScript. Prior to ES6, we created **hash** maps through objects:

```
var map = new Object();
map[key1] = 'value1';
map[key2] = 'value2';
```

However, this does not protect us from accidentally overriding functions with specific property names:

```
> getOwnProperty({ hasOwnProperty: 'Hah, overwritten'}, 'Pwned');
> TypeError: Property 'hasOwnProperty' is not a function
```

Actual **Maps** allow us to `set`, `get` and `search` for values (and much more).

```
let map = new Map();
> map.set('name', 'david');
> map.get('name'); // david
> map.has('name'); // true
```

The most amazing part of Maps is that we are no longer limited to just using strings. We can now use any type as a key, and it will not be type-cast to a string.

```
let map = new Map([
    ['name', 'david'],
    [true, 'false'],
    [1, 'one'],
    [{}, 'object'],
    [function () {}, 'function']
]);

for (let key of map.keys()) {
    console.log(typeof key);
    // > string, boolean, number, object, function
}
```

> **Note**: Using non-primitive values such as functions or objects won't work when testing equality using methods such as `map.get()`. As such, stick to primitive values such as Strings, Booleans and Numbers.

We can also iterate over maps using `.entries()`:

```
for (let [key, value] of map.entries()) {
    console.log(key, value);
}
```