JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

FROM ZERO TO EXPERT!

@JONASSCHMEDTMAN

SECTION

OBJECT ORIENTED PROGRAMMING (OOP) WITH JAVASCRIPT

LECTURE

OOP IN JAVASCRIPT

OOP IN JAVASCRIPT: PROTOTYPES

"CLASSICAL OOP": CLASSES

Class

INSTANTIATION

Instance

- Objects (instances) are **instantiated** from a class, which functions like a blueprint;
- Behavior (methods) is **copied** from class to all instances.

OOP IN JS: PROTOTYPES

Prototype ← Contains methods

PROTOTYPAL INHERITANCE/ DELEGATION

Object ← Can access methods

- Objects are **linked** to a prototype object;
- **Prototypal inheritance:** The prototype contains methods (behavior) that are **accessible to all objects linked to that prototype;**
- Behavior is **delegated** to the linked prototype object.

👉 **Example: Array**

```
const num = [1, 2, 3];
num.map(v => v * 2);
```

MDN web docs
moz://a

```
Array.prototype.keys()
Array.prototype.lastIndexOf()
Array.prototype.map()
```

`Array.prototype` is the **prototype** of all array objects we create in JavaScript

Therefore, **all** arrays have access to the `map` method!

```
ƒ Array()
  arguments: (...)
  caller: (...)
  length: 1
  name: "Array"
  prototype: Array(0)
    unique: ƒ ()
    length: 0
    constructor: ƒ Array()
    concat: ƒ concat()
    map: ƒ map()
```

3 WAYS OF IMPLEMENTING PROTOTYPAL INHERITANCE IN JAVASCRIPT

🤔 *“How do we actually create prototypes? And how do we link objects to prototypes? How can we create new objects, without having classes?”*

☝️ **The 4 pillars of OOP are still valid!**

- 👉 Abstraction
- 👉 Encapsulation
- 👉 Inheritance
- 👉 Polymorphism

1 Constructor functions

- 👉 Technique to create objects from a function;
- 👉 This is how built-in objects like Arrays, Maps or Sets are actually implemented.

2 ES6 Classes

- 👉 Modern alternative to constructor function syntax;
- 👉 “Syntactic sugar”: behind the scenes, ES6 classes work **exactly** like constructor functions;
- 👉 ES6 classes do **NOT** behave like classes in “classical OOP” (last lecture).

3 `Object.create()`

- 👉 The easiest and most straightforward way of linking an object to a prototype object.