# JavaScript OO

Uma referência ao tutorial da Mozilla:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript

# Orientar a Objetos é...

Uma abstração da realidade em termos computacionais permitindo a representação de objetos do nosso dia a dia através de tecnologias de desenvolvimento, isto é utilizando estruturas de dados disponíveis.

# Me empresta uma caneta?

```
var Caneta = function (cor) {
  this.cor = cor;
  console.log('Caneta inicializada');
};

var canetaVermelha = new Caneta(‘vermelha’);
var canetaAmarela = new Caneta('amarela');
```

# Objetos possuem ações (methods)

```
var Caneta = function (cor) {
  this.cor = cor;
  console.log('Caneta inicializada');
};

Caneta.prototype.pintar = function() {
  console.log(“Estou pintando”);
};
var canetaVermelha = new Caneta(‘vermelha’);
canetaVermelha.pintar();
```

# O que são Prototypes?

Every JavaScript object has a prototype. The prototype is also an object.

All JavaScript objects inherit their properties and methods from their prototype.

# Herança (Com construtor)

```
function Graph() {
  this.vertices = [];
  this.edges = [];
}

Graph.prototype = {
  addVertex: function(v){
    this.vertices.push(v);
  }
};

var g = new Graph();
```

# Herança (Object.create introduzido pelo ECMA5)

```
var a = {a: 1};
// a ---> Object.prototype ---> null

var b = Object.create(a);
// b ---> a ---> Object.prototype ---> null
console.log(b.a); // 1 (inherited)

var c = Object.create(b);
// c ---> b ---> a ---> Object.prototype ---> null

var d = Object.create(null);
// d ---> null
console.log(d.hasOwnProperty);
// undefined, because d doesn't inherit from Object.prototype
```

# Herança Extends (ECMA 6)

```
"use strict";

class Polygon {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
}
```

```
class Square extends Polygon {
  constructor(sideLength) {
    super(sideLength, sideLength);
  }
  get area() {
    return this.height * this.width;
  }
  set sideLength(newLength) {
    this.height = newLength;
    this.width = newLength;
  }
}

var square = new Square(2);
```