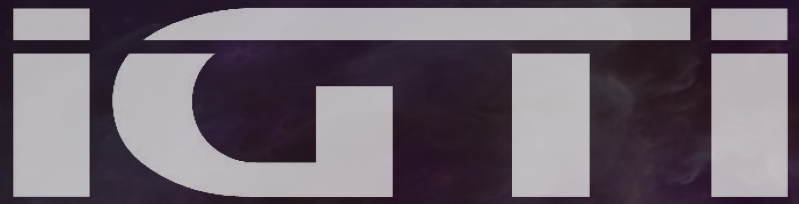


A aula interativa do **Desenvolvimento Avançado Javascript** começará em breve!

Atenção:

1) Acesse a aula com NOME COMPLETO, para que sua frequência seja computada.
2) Mantenha o microfone DESLIGADO, abrindo-o apenas em momentos de interatividade.
3) Mantenha seu vídeo sempre ATIVADO.

Desenvolvimento Avançado Javascript

PRIMEIRA AULA INTERATIVA

PROF. BRUNO TEIXEIRA

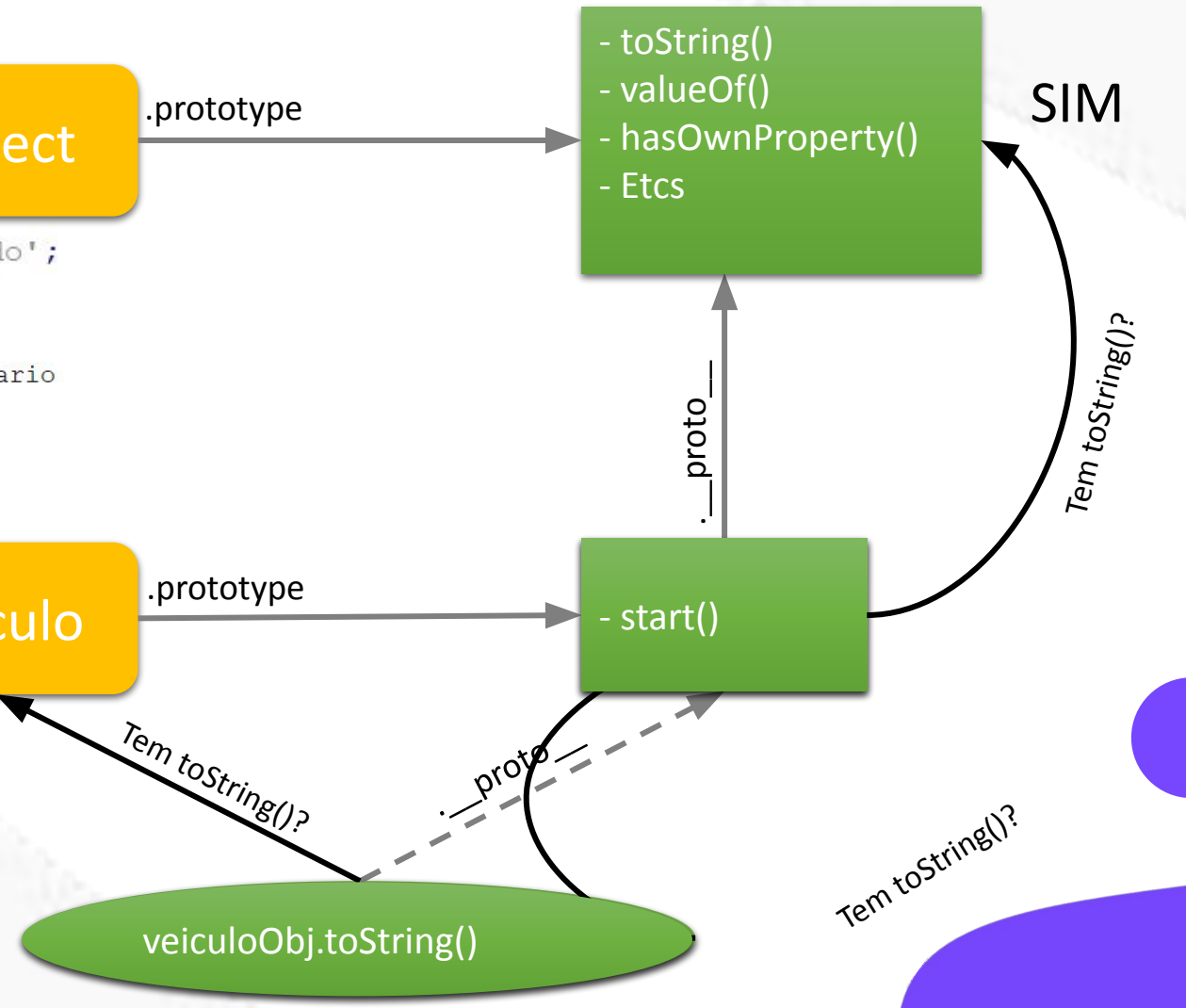
Nesta aula

- Temas do fórum.
- Tópicos da disciplina.
- Atividade Prévia.

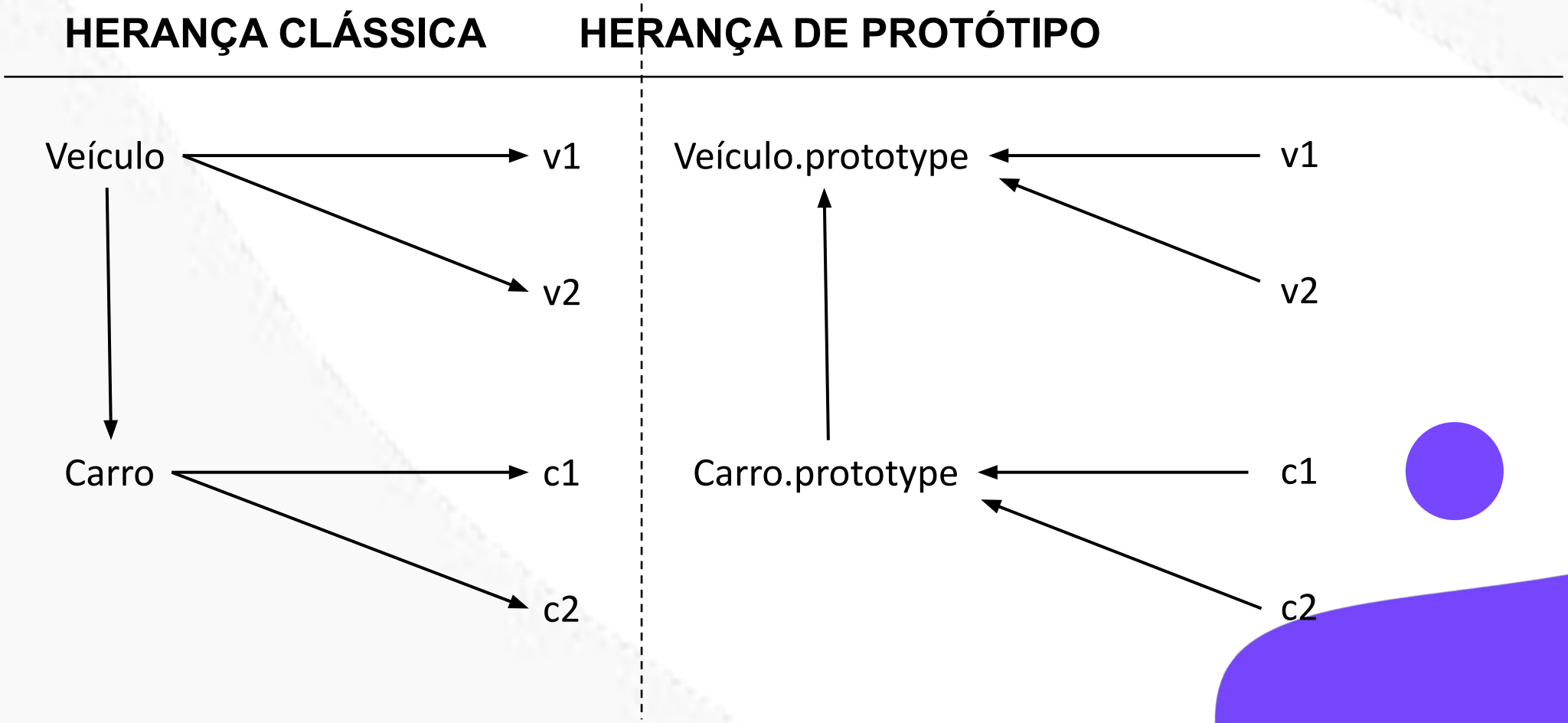
Protótipos

- *Prototype* é um objeto que existe como propriedade em cada função do JavaScript.
- Aplicações:
  - Adição e busca por propriedades e métodos num objeto.
  - Implementação de herança em JavaScript.

```
function Veiculo() {
    this.proprietario = 'Fernando';
}

Veiculo.prototype.start = function() {
    return "Veiculo do "+this.proprietario
           +" iniciando..."
}

var veiculoObj = new Veiculo();

veiculoObj.toString();
```



Protótipos - Herança



Fonte: You Don’t Know JS Book Series.

Protótipos - Herança

```
function Veiculo() {
    this.proprietario = 'Fernando';
}

Veiculo.prototype.start = function() {
    return "Veiculo do "+this.proprietario
            +" iniciando..."
}

function Carro(proprietario) {
  Veiculo.call(this,proprietario);
}

Carro.prototype = Object.create(Veiculo.prototype);


Carro.prototype.acelera = function() {
  console.log("Olá, "+ this.start());
};

var c1 = new Carro("Maria");
var c2 = new Carro("Lucas");

c1.acelera();
c2.acelera();
```

Protótipos - Herança

```
function Veiculo() {
    this.proprietario = 'Fernando';
}

Veiculo.prototype.start = function() {
    return "Veiculo do "+this.proprietario
           +" iniciando..."
}

function Carro(proprietario) {
  Veiculo.call(this,proprietario);
}

Carro.prototype = Object.create(Veiculo.prototype);

Carro.prototype.acelera = function() {
  console.log("Olá, "+ this.start());
};

var c1 = new Carro("Maria");
var c2 = new Carro("Lucas");

c1.acelera();
c2.acelera();
```

superclass - Veiculo

Método - superclass

subclass - Carro

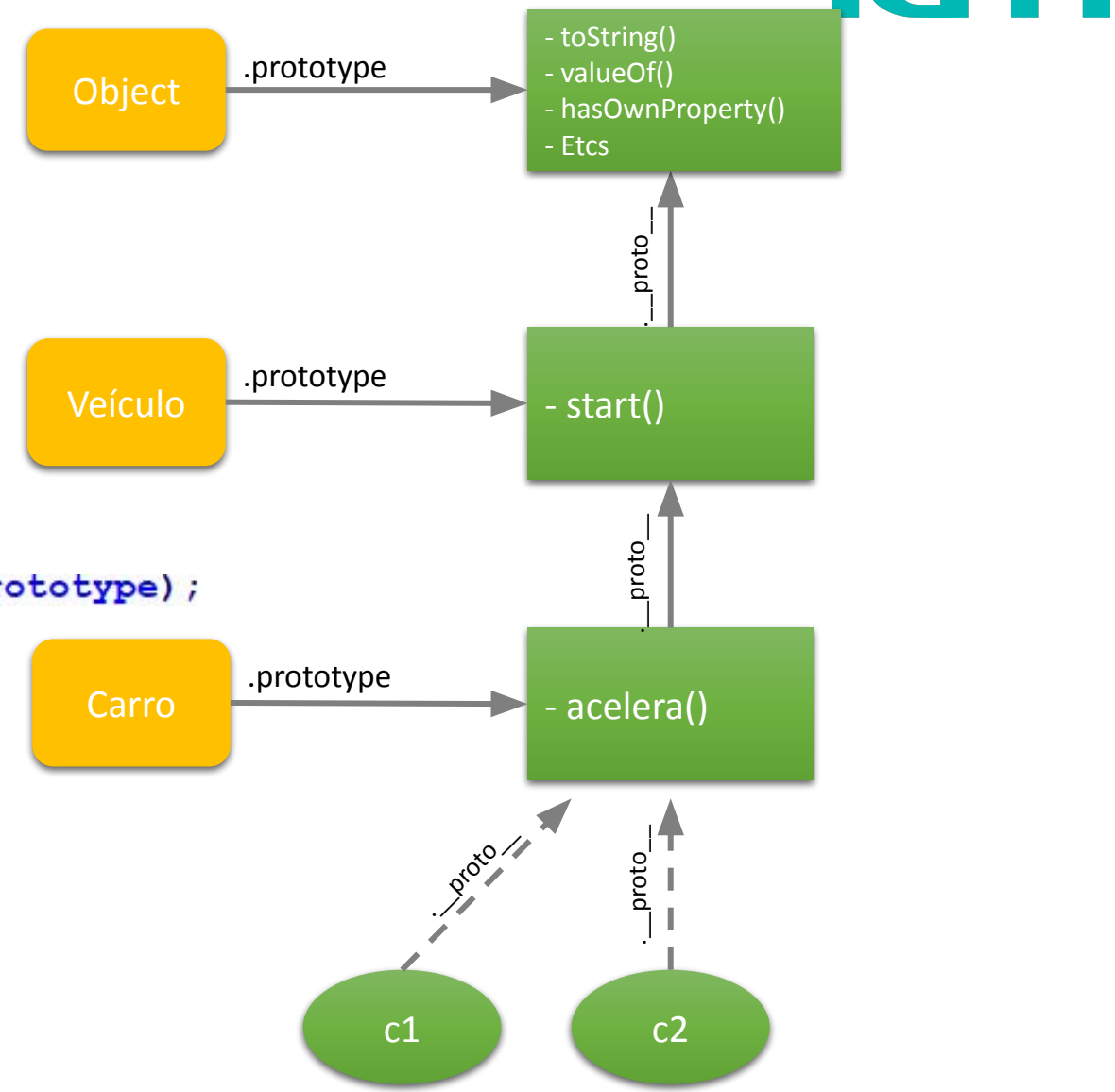
subclass extends super class

Método - subclass

Instância da subclass

Protótipos - Herança

```
function Veiculo() {
    this.proprietario = 'Fernando';
}

 Veiculo.prototype.start = function() {
    return "Veiculo do "+this.proprietario
           +" iniciando..."
 }

function Carro(proprietario) {
  Veiculo.call(this,proprietario);
}

Carro.prototype = Object.create(Veiculo.prototype);


Carro.prototype.acelera = function() {
  console.log("Olá, "+ this.start());
};

var c1 = new Carro("Maria");
var c2 = new Carro("Lucas");

c1.acelera();
c2.acelera();
```



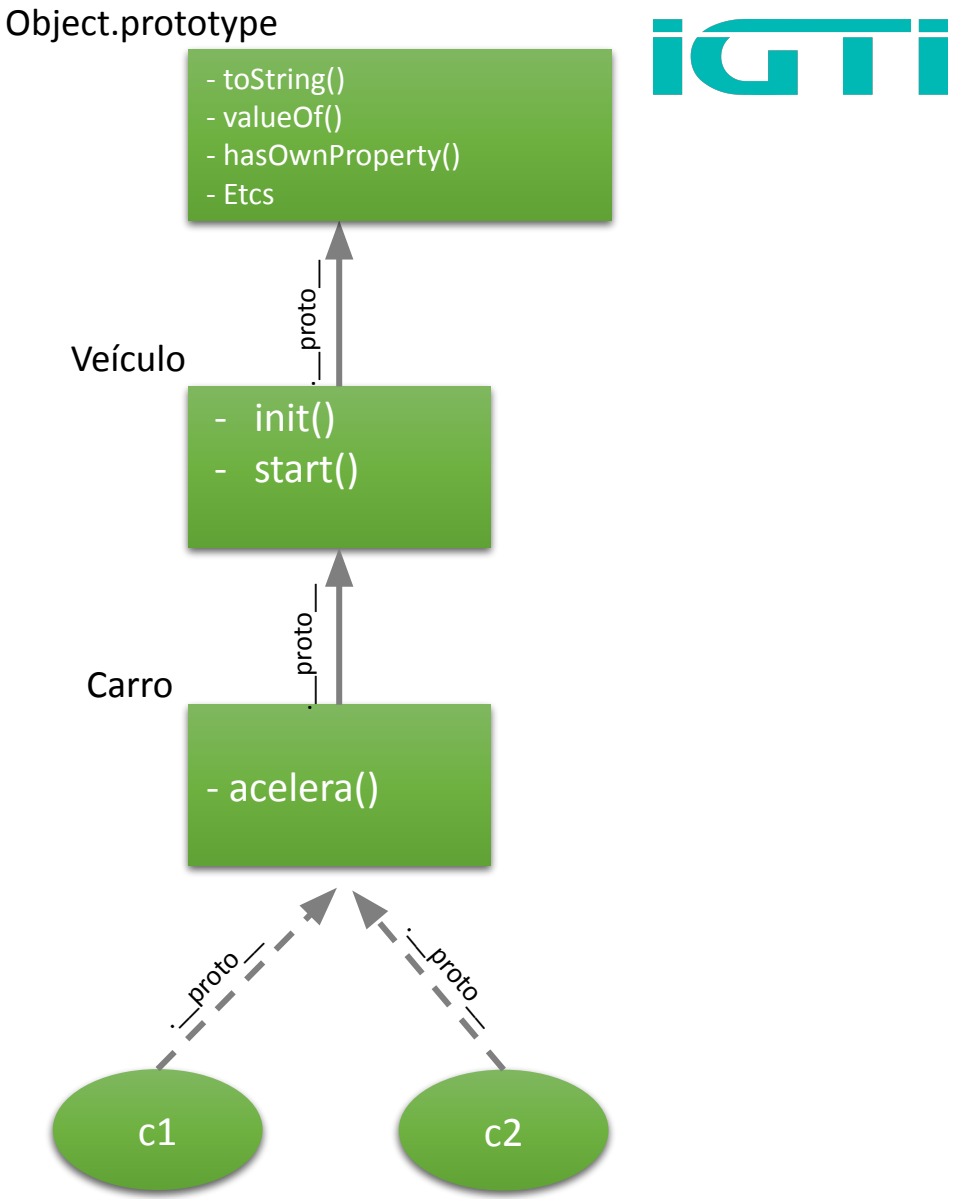
Protótipos - Herança

```
function Veiculo() {
    this.proprietario = 'Fernando';
}

Veiculo.prototype.start = function() {
    return "Veiculo do "+this.proprietario
           +" iniciando..."
}

function Carro(proprietario) {
  Veiculo.call(this,proprietario);
}

Carro.prototype = Object.create(Veiculo.prototype);


Carro.prototype.acelera = function() {
  console.log("Olá, "+ this.start());
};

var c1 = new Carro("Maria");
var c2 = new Carro("Lucas");

c1.acelera();
c2.acelera();
```

```
var Veiculo = {
  init: function(proprietario) {
    this.proprietario = proprietario;
  },
  start: function() {
    return "veiculo do "+this.name +
           " iniciando...";
  }
}

var Carro = Object.create(Veiculo);


Carro.acelera = function() {
  console.log("Olá, "+ this.start());
};

var c1 = Object.create(Car);
c1.init('Maria');
var c2 = Object.create(Car);
c2.init('Lucas');
c1.acelera();
c2.acelera();
```

Protótipos - Herança

```
var Veiculo = {
  init: function(proprietario) {
    this.proprietario = proprietario;
  },
  start: function() {
    return "veiculo do "+this.name +
           " iniciando...";
  }
}

var Carro = Object.create(Veiculo);

Carro.acelera = function() {
  console.log("Olá, "+ this.start());
};


var c1 = Object.create(Car);
c1.init('Maria');
var c2 = Object.create(Car);
c2.init('Lucas');
c1.acelera();
c2.acelera();
```

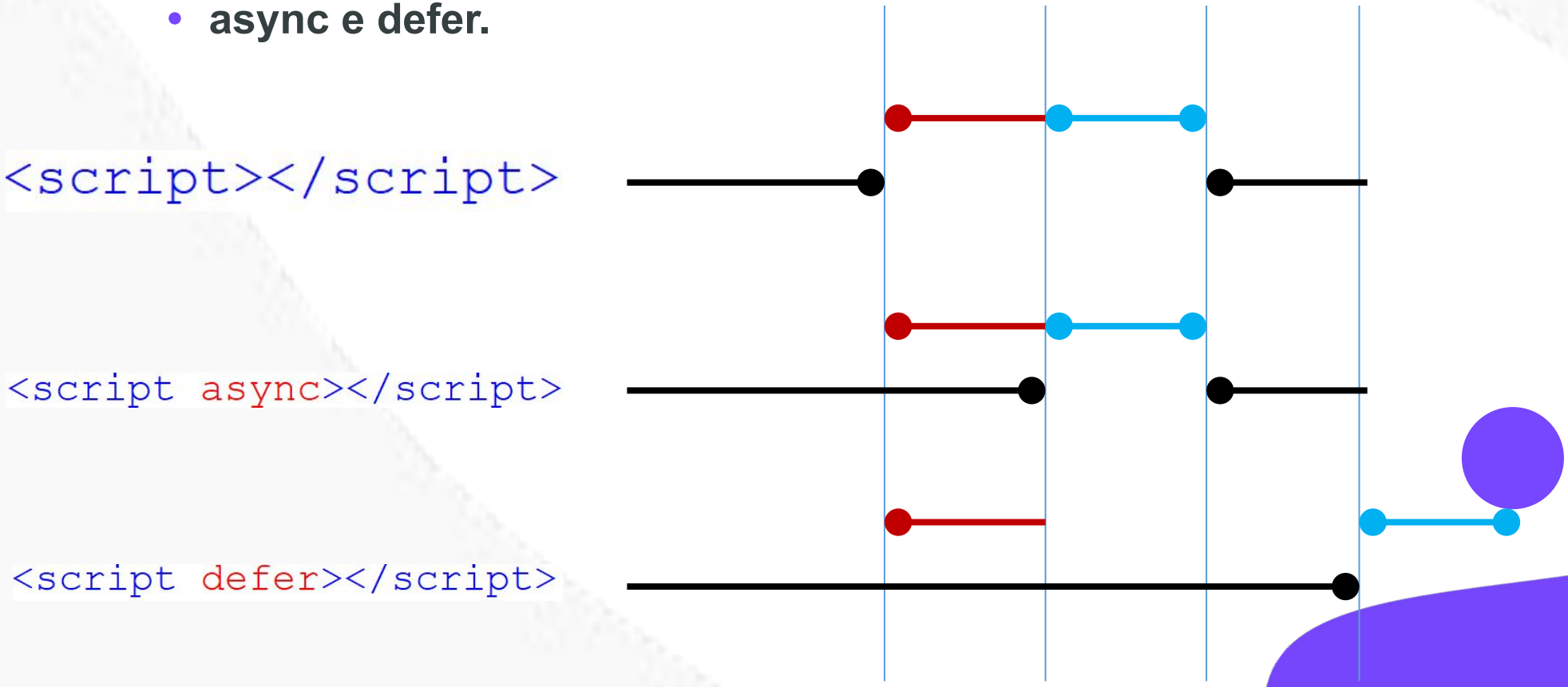


Atividade Prévia

- Dicas para a atividade prévia.

JavaScript Assíncrono

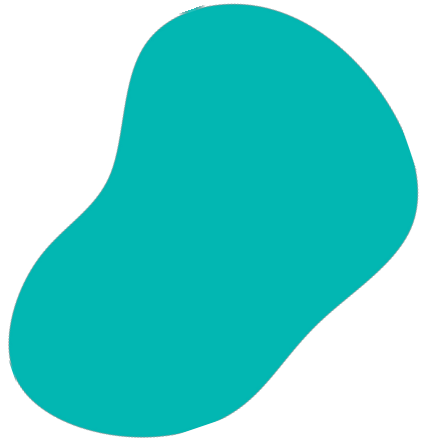
- async e defer.



Funções Construtoras

Funções Fábrica

Classes



Classes

```
class Carro {
 constructor(prop) {
     this.proprietario = prop
 }

 acelera() {
     console.log("Acelerando.....")
 }
}

const c1 = new Carro("Marcia")

c1.acelera()
```

Funções Construtoras

```
function Carro (prop) {
    this.proprietario = prop
}

const c1 = new Carro(prop);
```

Funções Fábrica

```
function carro(prop) {
    return {
        proprietario = prop
    }
}

console.log(carro("Maria"))
```

Conclusão

- ✔ Atividade Prévia.
- ✔ Protótipos.
- ✔ JavaScript Assíncrono.
- ✔ Funções Construtoras e Funções Fábrica.