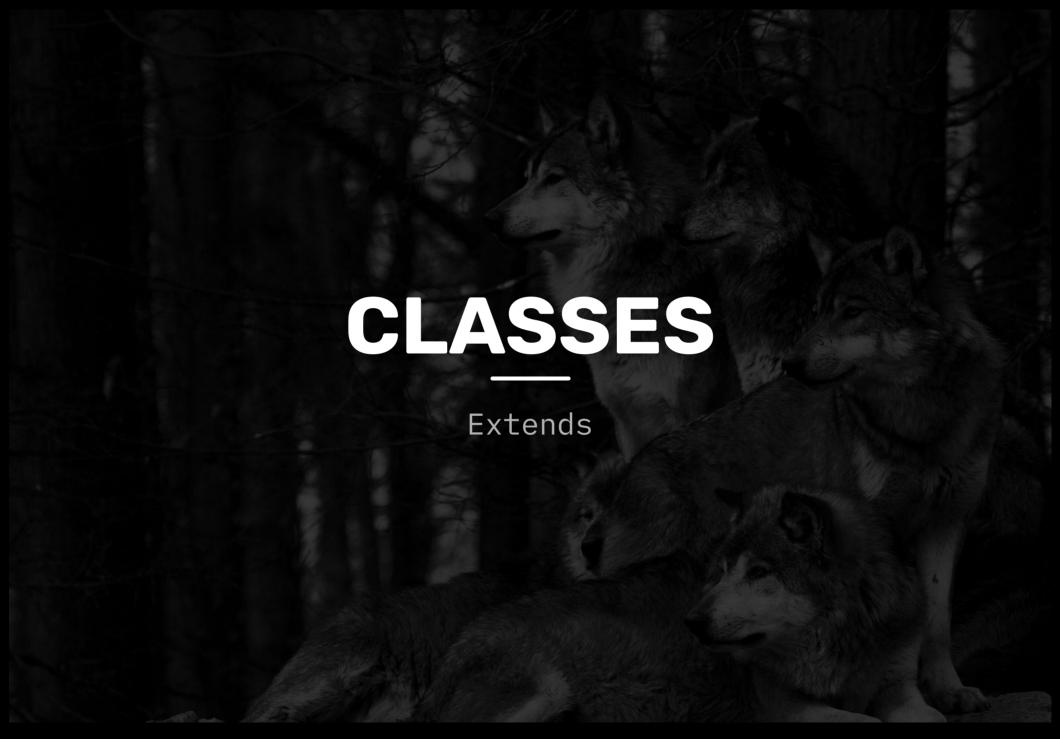
ORIGIMID



Subclasses

É possível criarmos uma subclasse, esta irá ter acesso aos métodos da classe à qual ela estendeu através do seu protótipo.

```
class Veiculo {
  constructor(rodas) {
    this.rodas = rodas;
  acelerar() {
    console.log('Acelerou');
class Moto extends Veiculo {
  empinar() {
    console.log('Empinou com ' + this.rodas + ' rodas');
const honda = new Moto(2);
honda.empinar();
```

Métodos

Podemos escrever por cima de um método herdado.

```
class Veiculo {
  constructor(rodas) {
    this.rodas = rodas;
  acelerar() {
    console.log('Acelerou');
class Moto extends Veiculo {
  acelerar() {
    console.log('Acelerou muito');
const honda = new Moto(2);
honda.acelerar();
```

ORIGIMID

Super

É possível utilizar a palavra chave super para falarmos com a classe que pai e acessarmos os seus métodos e propriedades.

```
class Veiculo {
  constructor(rodas) {
    this.rodas = rodas;
  acelerar() {
    console.log('Acelerou');
class Moto extends Veiculo {
  acelerar() {
    super.acelerar();
    console.log('Muito');
const honda = new Moto(2);
honda.acelerar():
```

Super e Constructor

Podemos utilizar o super para estendermos o método constructor.

```
class Veiculo {
  constructor(rodas, combustivel) {
    this.rodas = rodas;
    this.combustivel = combustivel;
class Moto extends Veiculo {
  constructor(rodas, combustivel, capacete) {
    super(rodas, combustivel);
    this.capacete = capacete;
const honda = new Moto(4, 'Gasolina', true);
```