**Sabendo ainda mais sobre herança**

[Sugerir melhoria](https://cursos.alura.com.br/course/javascript-es6-orientacao-a-objetos-parte-1/section/6/task/7)

Durante esta aula, aprendemos a utilizar herança através do uso da palavra chave extends. Contudo, este instrutor que preza sempre por um código menos verboso e enxuto deixou passar um detalhe, que pode enxugar ainda mais nosso código. Primeiro, vamos lembrar o uso de extends:

class Funcionario {

constructor(nome) {

this.\_nome = nome;

}

get nome() {

return this.\_nome;

}

set nome(nome) {

this.\_nome = nome;

}

}

class Secretaria extends Funcionario {

constructor(nome) {

super(nome);

}

atenderTelefone() {

console.log(`${this.\_nome} atendendo telefone` );

}

}

Veja que no construtor de Secretaria eu recebo o nome da secretária pelo construtor e passamos esse parâmetro para a classe pai. Contudo, essa solução neste cenário é um tanto verbosa. Qual motivo? **Por padrão, quando uma classe herda outra, ela também herda seu construtor**. Dessa maneira, podemos reescrever a classe Secretaria como:

class Secretaria extends Funcionario {

// não precisei adicionar constructor e nem chamar super!

atendeTelefone() {

console.log(`${this.\_nome} atendendo telefone` );

}

}

Criando uma instância de Secretaria:

let secretaria = new Secretaria('Suzete');

O JavaScript automaticamente considera o construtor da classe pai Funcionario, que recebe um parâmetro.

Contudo, apareceu um novo requisito na classe Secretaria. Toda secretária deve ter, além de um nome, um outro funcionário ao qual está subordinada:

class Secretaria extends Funcionario {

constructor(nome, funcionario) {

this.\_nome = nome;

this.\_funcionario = funcionario;

}

atendeTelefone() {

console.log(`${this.\_nome} atendendo telefone` );

}

get funcionario() {

return this.\_funcionario;

}

}

No exemplo anterior, foi necessário adicionar o construtor porque a propriedade \_funcionario só existe em Secretaria. O problema é que nosso código não funciona! Se tentarmos fazer:

let secretaria = new Secretaria('Suzete', new Funcionario('Barney'));

Recebemos o erro:

Uncaught ReferenceError: this is not defined

Quando temos um construtor na classe filha que recebe uma quantidade de parâmetros diferentes do construtor da classe pai, para que o this seja inicializado com um valor, precisamos chamar o construtor da classe pai, passando os parâmetros que ela precisa. Corrigindo nosso código:

class Secretaria extends Funcionario {

constructor(nome, funcionario) {

super(nome); // cuidado, tem que ser a primeira instrução!

this.\_funcionario = funcionario;

}

atenderTelefone() {

console.log(`${this.\_nome} atendendo telefone` );

}

}

A palavra super, como já vimos, nos dá acesso à **superclasse**, ou seja, a classe que foi herdada. Em nosso caso, estamos passando para o construtor de Funcionario o nome recebido pelo construtor de Secretaria. O segundo parâmetro, funcionario, só diz respeito à secretária, por isso a propriedade foi adicionada em this.\_funcionario.

Mas atenção! A chamada do construtor da classe pai deve ser a primeira instrução no construtor da classe filha. Se por acaso tivéssemos o construtor assim:

constructor(nome, funcionario) {

this.\_funcionario = funcionario; // this ainda não foi inicializado

super(nome);

}

Teremos um erro, porque estamos tentando acessar um this que ainda não foi inicializado.