Capitulo 2 – Orientação a Objetos em JavaScript

Aula 3 – Protótipos e Classes

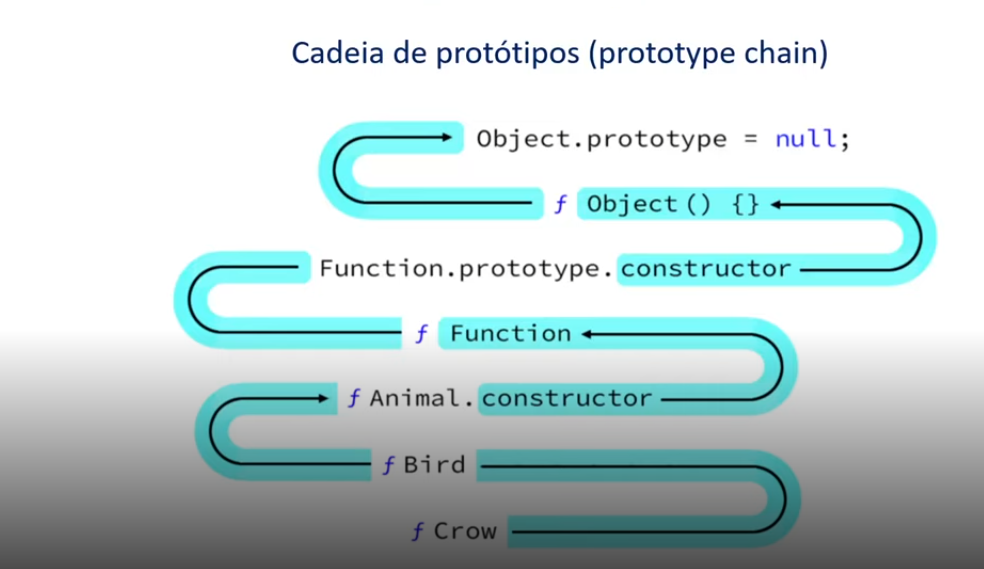
Protótipos

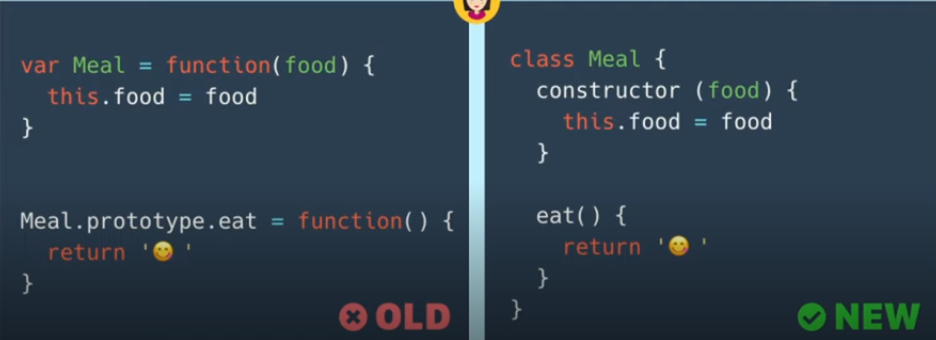
Os protótipos no JavaScript são basicamente o esqueleto de todos os objetos, então Todos os objetos JavaScript vão herdar propriedades e métodos de um prototype. E O OBJETO Object.prototype esta no topo desta cadeia.

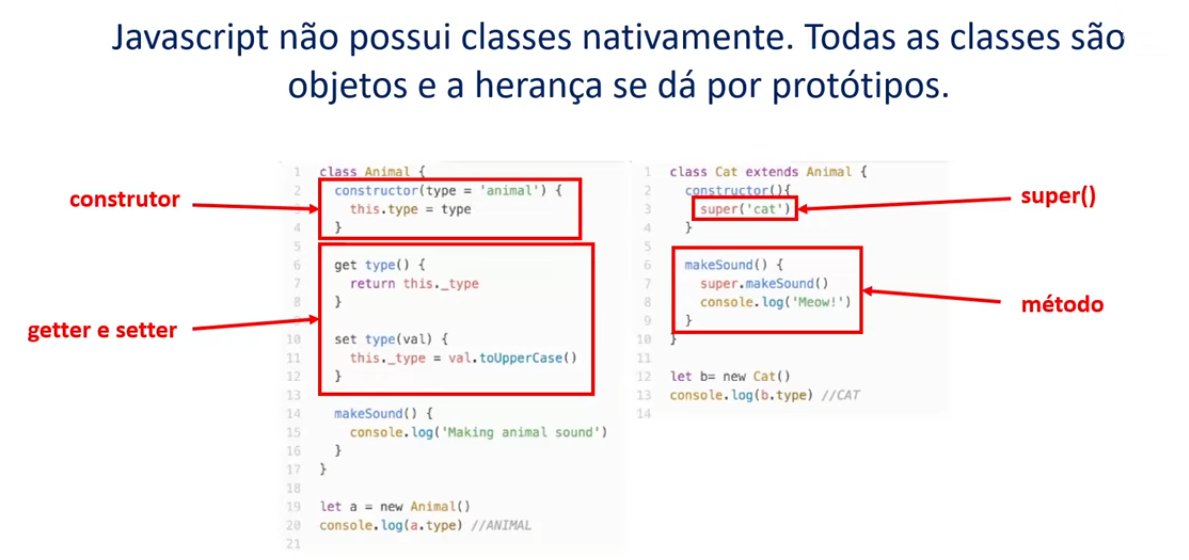
**Sempre que nós temos um objeto no JavaScript que é de um tipo complexo/ composto, ele vai ter essa propriedade “\_proto” que vai ter uma serie de métodos e propriedades, então nós conseguimos utilizar:**

Objeto. hasOwnProperty () = Para ver se um objeto tem alguma chave com aquele nome por conta do prototype, por isso está na classe pai dele, mesmo que nós não tínhamos escrito essa função.

Podemos converter um objeto para um “. toString ()”.

Classes

**Classes no JavaScript não existem nativamente**, ela é o açúcar sintático uma coisa que chama: **Syntatic sugar** que é basicamente uma sintaxe feita para facilitar a escrita. Mas o que acontece e que agente sempre utiliza objetos, e objetos que tem protótipos. **Todas as classes são objetos e a herança se da por protótipos.**



Aqui temos um print de uma classe “**animal**” e uma classe filha dela “**cat**” e tem uma anatomia de uma classe em JavaScript. Então a classe em JavaScript sempre tem um “**construtor**”. O construtor ele vai construir a classe e se você passa parâmetros ele vai atribuir esses parâmetros a certos valores dentro dessa classe. O construtor, o tipo e “animal”, tem um “**parâmetro padrão**”. Também temos “**get e se**t” para nos termos acesso a propriedade de um objeto. Então o “**get typer**” que é o tipo de animal, ele vai retornar o tipo e o “**set**” e para determinar o tipo que tem o outro valor. As propriedades elas são realmente são apenas dessa classe.

Na classe filha “**cat**”, ela tem um método chamado “**super**”, o super ele vai fazer e mandar pra cima a função pai os parâmetros que estão ali. Então quando falo “**super(‘cat’)**” o que esta fazendo e : esta construindo cat e no momento que ele for construído nos vamos utilizar essa classe, o tipo da classe pai vai ser “**cat**”. E também tem o método “**makeSound ()**” que nos sempre podemos sobre escrever métodos.

Extend nomeDaClasse = Para referenciar a classe pai.