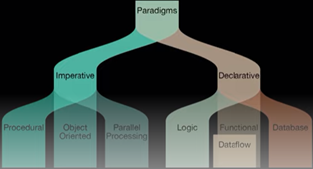
Este curso, obviamente, trata curtamente da orientação ao objeto em javascript.

A programação tem dois paradigmas, o **imperativo** que foca em **como você vai resolver os problemas apresentados** e o **declarativo foca no que você vai fazer**. São essencialmente ‘formas de programar’.

Na Programação Orientada a Objetos, apelidada POO, os **programas são objetos que possuem uma série de objetos** naturalmente isso significa que ela foca em trabalhar utilizando as variáveis do tipo objeto que temos visto. Se baseia em quatro pilares principais:

**Herança**: consiste na herança de características de objetos filhos de seus “pais”.

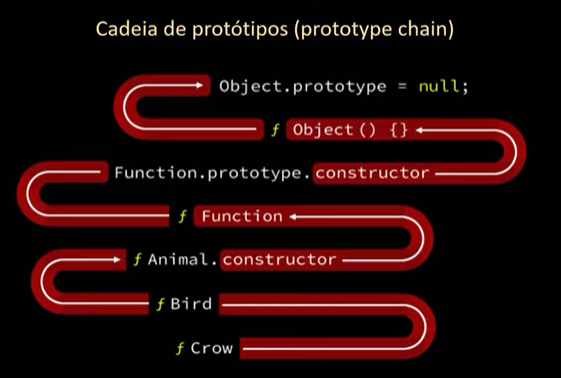
**Polimorfismo**: se refere a objetos que herdam o mesmo comportamento do pai, mas ao invoca-los dentro de seus métodos se comportam de maneira diferente um bom exemplo dado é pensar na característica “falar” de um objeto pai “animal”, todos animais podem falar, mas ao falar um objeto “gato” irá miar e um objeto “cachorro” latir.

**Encapsulamento**: diz respeito ao fato de cada classe possuir propriedades e métodos independentes do restante do código.

**Abstração**: abstração em sua definição é o ‘processo mental que consiste em isolar um aspecto determinado de um objeto, a fim de simplificar a sua avaliação, classificação ou para permitir a comunicação do mesmo’, ou seja, é pegar um problema complexo e torna-lo cada vez mais simples a fim de entender suas diferentes partes, **é um processo de generalização de um objeto complexo**.

**OOJS** é o Orientação ao Objeto em JavaScript.

**Protótipos** são esqueletos de todos os objetos no JS, assim todos os objetos do JS vão herdar características de um protótipo, um **prototype**, assim, **o objeto Object.prototype está no topo desta cadeia, desta árvore genealógica de objetos**.

A imagem ao lado mostra um exemplo de cadeia de objetos que leva ao protótipo, ao executar um comando em um objeto filho, o JS vai procurar o comando de baixo para cima, ou seja, vai procurar primeiro as caracteristicas dele, depois do pai, depois do avo, e assim por diante até achar a caracteristica.

**Classes** não existem nativamente no JS, são um **“açucar sintático**”, sintax sugar, um sintaxe feita apenas para facilitar a escrita, na prática, utilizamos sempre os Objetos, com a nova versão que temos agora, nós podemos fazer a sintaxe de class mais parecida com outras linguagens que são feitas para acomodar o paradigma de POO no JS, porêm o **class, em sua definição aqui, não é diferente de objetos com protótipos**.

A class em JS sempre possui um **constructor** que irá construir a classe, e parâmetros que são atribuidos a valores dentro da classe.

O **get** é um subcomando que pega o valor requisitado dentro da classe, enquanto o **set** determina que tipo tem outro valor.

O método **super()** passa argumentos enviados para as classes ascendentes.