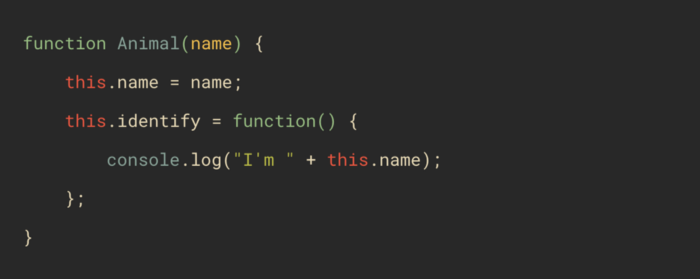
Pesquise sobre padrões de projeto e escolha um para apresentar e descrever o seu funcionamento. Além disso, explique quais as vantagens e desvantagens comparados a outros e mostre suas referências.

# Constructor pattern (Padrão do construtor)

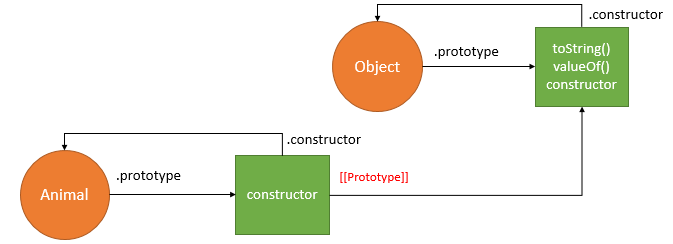
O JavaScript permite criar uma função de construtor customizada que define as propriedades e métodos de objetos definidos pelo usuário. Por convenção o nome de uma função de construtor em JavaScript começa com uma letra maiúscula.

O próximo exemplo reescreve o objeto animal do exemplo anterior:

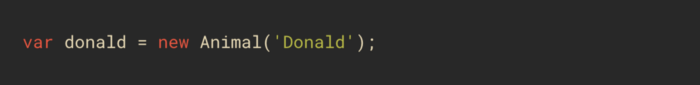


Diferente do padrão de fábrica, as propriedades e métodos do objeto animal são atribuídos diretamente ao objeto this dentro da função construtora.

Neste ponto, o motor JavaScript cria a função Animal() e um objeto anônimo. A função Animal() tem a propriedade prototype apontando para o objeto anônimo e o objeto anônimo tem a propriedade construtora apontando para a função Animal(). Além disso o motor JavaScript vincula o objeto anônimo ao objeto global.



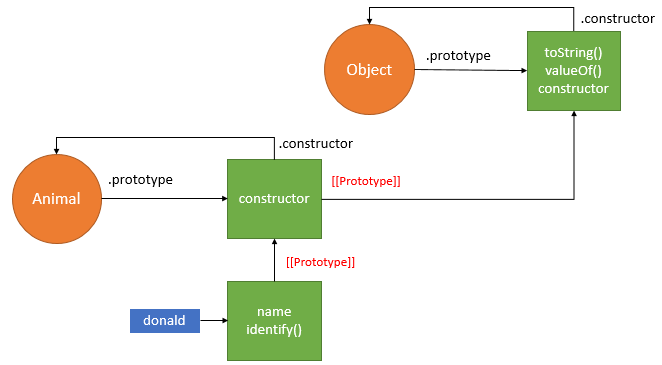
Para criar uma nova instância do Animal, use o operador new. Por exemplo:



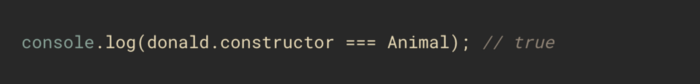
Nos bastidores, o JavaScript executa quatro etapas:

1. Crie um novo objeto.
2. Defina o valor de construtor this ao novo objeto.
3. Execute o código dentro do construtor, ou seja, adicionando propriedades ao novo objeto.
4. Retorne o novo objeto.

A figura a seguir ilustra o relacionamento entre o objeto donald e outros objetos.

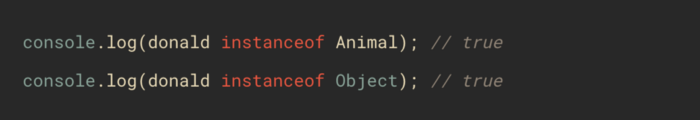


Veja o seguinte exemplo:

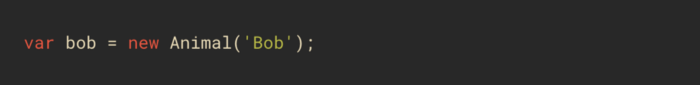


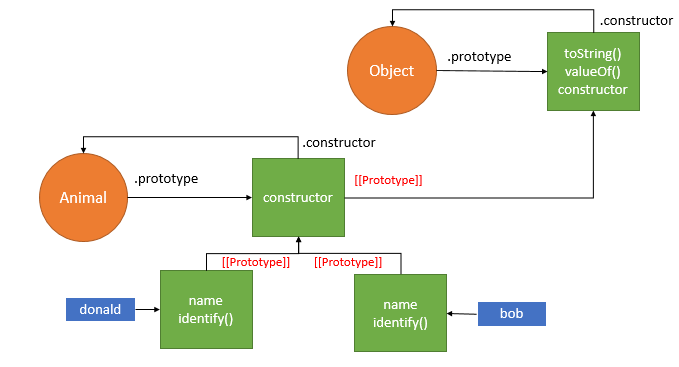
Neste exemplo, como o objeto donald não possui a propriedade constructor, o motor JavaScript segue a cadeia de protótipos para encontrá-lo no objeto Animal.prototype. Ele encontra a propriedade constructor no objeto Animal.prototype e nesse caso a propriedade constructor aponta para a função Animal(), portanto a instrução acima retorna true.

O objeto donald também é uma instância de Animal e Object como ilustrado abaixo:



A desvantagem do padrão construtor é que o mesmo método identify() é duplicado em cada instância. O código a seguir cria um novo objeto Animal chamado bob.





Como você vê, o método identi() é duplicado nos objetos donald e bob. Para resolver esse problema, você usa o **Padrão de protótipo**.