Contents

[**1.** **Prototype** 2](#_Toc66984153)

[**2.** **\_\_proto\_\_** 2](#_Toc66984154)

[**3.** **Reference Vs Value** 2](#_Toc66984155)

1. **Prototype**

Dùng trong **function constructor**

function Dog(name) { // the usual constructor function

this.name = name

}

Dog.prototype = ...

**prototype** is the object that is used to build \_\_proto\_\_ when you create an object with new.

**prototype** is **not available** on the instances themselves (or other objects), but only on the constructor functions.

Add method/property vào function constructor

1. **\_\_proto\_\_**

Dùng trong **instance** của function constructor

function Dog(name) { // the usual constructor function

this.name = name

}

Let a = new Dog(‘Hector’);

a.\_\_proto\_\_ = ....

Add method/property vào function constructor

**\_\_proto\_\_** is available everywhere.

1. **Reference Vs Value**



Khi gán d = c thì nó chỉ gán địa chỉ/address của c thôi chứ ko gán value của c

Khi ==, === thì nó chỉ so sánh địa chỉ/address thôi chứ ko so sánh value

function add(arr, el) { arr.push(el)} =>Khi truyền input as array thì nó cũng sẽ lấy địa chỉ/address rồi implement chứ ko lấy giá trị của array

function add(arr, el) { arr = [el] } => Nhưng khi gán nó = 1 array khác thì nó sẽ có địa chỉ khác

nếu declare **const arr = [1,2,3]**

thì **vẫn modified được bằng** các phương thức của arr(chứ **ko phải phép gán**). Vì nó ko làm thay đổi địa chỉ/address của const này. Và const obj cũng chỉ lưu địa chỉ/address

this

this (aka "the context") is a special keyword inside each function: giá trị của this chỉ **phụ thuộc** cách function được **gọi/called** (runtime binding)chứ **ko** phải cách nó được định **nghĩa/defined**