DESCRIPTION

**Encapsulation via Closures**

Encapsulation means hiding information or data.The simplest a way to create encapsulation in JavaScript is using closures. A closure can be created as a function with private state. When creating many closures sharing the same private state, we create an object.

**Example:**

const Book = function(t, a) {

let title = t;

let author = a;

return {

summary : function() {

console.log(`${title} written by ${author}.`);

}

}

}

const book1 = new Book('Hippie', 'Paulo Coelho');

book1.summary(); // Returns Hippie written by Paulo Coelho.

**Prototypical inheritance**

Object Prototypes (\_\_proto\_\_ )- All JavaScript objects have a prototype. Browsers implement prototypes through the \_\_proto\_\_ property.

Function prototypes (prototype ) - In JavaScript, all functions are also objects, which means that they can have properties.Any time you create a function, it will automatically have a property called prototype.Thus, Functions also have a prototype property.

When we call a function with new, it sets the returned object’s \_\_proto\_\_ property equal to the function’s prototype property. This is the key to inheritance.

Inheritance in JavaScript is implemented through the **prototype chain**. Every normally created object, array, and function has a prototype chain of \_\_proto\_\_ properties ending with Object.prototype at the top.

**Example:** Here, we can say that "animal is the prototype of rabbit" or "rabbit prototypically inherits from animal". The animal properties and methods are become automatically available in rabbit. Such properties are called “inherited”.

let animal = {

eats: true

walk() {

alert("Animal walk");

}

};

let rabbit = {

jumps: true

\_\_proto\_\_: animal // sets animal to be a prototype of rabbit.

};

// we can find both properties in rabbit now:

alert( rabbit.eats ); // true

alert( rabbit.jumps ); // true

// walk is taken from the prototype

rabbit.walk(); // Animal walk

In the above example, when alert tries to read property rabbit.eats, it’s not in rabbit but JavaScript follows the prototype reference and finds it in animal.Also the walk() method is automatically taken from the prototype.