Desarrollo Web En Entorno Cliente

TEMA 4

Objetos definidos por el usuario

Daniel López Lozano

Escuela Arte Granada

Índice

- Programación orientada a objetos clásica.
- ECMA Script 6.

Programación orientada a objetos clásica

Daniel López Lozano

Escuela Arte Granada

- JavaScript no es un lenguaje orientado a objetos tan puro como es el caso de Java, C++,Python,Ruby….
- JavaScript realmente está basado en prototipos.
- Un prototipo es una función que hace las veces de constructor y de la clase en los lenguajes orientados a objetos.
- En esencia, cualquier clase, objeto, prototipo o propiedad en JavaScript, realmente son funciones.

- Por eso, es más correcto decir que, JavaScript es un lenguaje basado en funciones o prototipos.
- Para crear un objeto en JavaScript, tenemos que crear su constructor, que contendrá la definición de todos sus atributos, métodos, e indicará que datos son necesarios para su creación.
- A diferencia de otros lenguajes, en JavaScript no existe la sobrecarga de métodos, por tanto sólo es posible definir un único constructor para cada clase de objetos.

Definición de una clase

```
function Carta(n, p){
    var numero=n;
    var palo=p;
    this.getNumero=function(){
        return numero;
    }

    this.getPalo=function(){
        return palo;
    }

    this.setNumero=function(nuevo){
        numero=nuevo;
    }

    this.setPalo=function(nuevo){
        palo=nuevo;
    }
```

Definición y uso de una clase

```
    this.muestraTodo = function(){ //método que muestra los atributos
            var todo="";
            for(i in this){
                todo=todo+i+" "+this[i]+" ";
            }
            return todo;
    }

    function oculto(){ //método privado
            return Math.random();
    }
}
var unaCarta=new Carta(10,"copas");
unaCarta.setPalo("bastos");
alert(unaCarta.getNumero()+" "+unaCarta.getPalo()); //muestra número y palo
```

- La POO en JavaScript no es tan potente como en otros lenguajes orientados a objetos pero a pesar de esto, es posible heredar de otras clases para reutilizar el código
- Para esto debemos utilizar la palabra reservada prototype para indicar que una clase hereda de otra.
- Es necesario llamar al constructor de la clase padre en el constructor de la clase hija de la siguiente forma:

clasePadre.call(this , atributosPadre);

Implementación de la herencia

```
function Persona(nom) {
    var nombre=nom;
    this.saluda=function(){
        return "Me llamo "+nombre+" ";
    }
}

function Alumno(nom, est, not) {
    Persona.call(this,nom);
    var estudios=est;
    var expediente=not;
    this.mostrarEx=function(){
        return "estudié "+estudios+" con "+expediente+" de expediente";
    }
}
Alumno.prototype = new Persona;
var yo=new Alumno("Jesús", "Informatica", 8);
alert(yo.saluda()+yo.mostrarEx());
```

- EJERCICIO 1: Crea una clase para almacenar las reservas de un hotel, para una reserva se desea almacenar el dni del cliente, su fecha de entrada y salida y el número de personas de la reserva.
- Realiza métodos get y set para el dni del cliente, las fechas de entrada y salida, además de otro get y set para el número de personas de la reserva.
- Crear un método privado llamado DiasReserva que devuelva el numero de días entre las fecha de entrada y salida

- Crea también un método coste que indique el precio de la reserva según sean de 1 a 4 personas: 30, 50, 65, 75 euros respectivamente por día.
- Crear una reserva para 3 personas , mostrar el coste, ampliarla 3 días más y mostrar el nuevo coste.

- EJERCICIO 2: Crear la clase alojamiento que herede de reserva, esta clase tendrá como atributos adicionales desayuno que valdrá verdadero o falso para indicar si incluye el desayuno y pensión que podrá valer (1,2) que significa media pensión o pensión completa.
- Calcula el coste total (con un método), que es el coste anterior más 10 por desayuno, 20 por media pensión y 30 por pensión completa cada día.
- ImprimirFactura será un método que muestre todos los datos de una reserva.

ECMA Script 6

Daniel López Lozano

Escuela Arte Granada

- La programación orientada a prototipos fue siempre la forma de hacer POO en JS.
- Aunque los prototipos de JS son muy poderoso para los programadores que vienen de lenguajes orientados a objetos como Java o PHP les resulta extraña ya que no son propiamente objetos, o al menos no como el resto de lenguajes considera los objetos.
- Por ejemplos herencia múltiple, implementación de interfaces, sobrecarga no existen como tal y no permiten seguir el autentico patrón del desarrollo orientado a objetos.

- Para facilitar esto en ECMAScript 6 se agregaron una forma de crear estas clases, como en los demás lenguajes.
- Aunque al final esta nueva sintaxis para hacer POO es simplemente una capa sobre la sintaxis actual mediante prototipos, solo que más simple de entender.
- Incluyen constructores, herencia explicita, métodos get y set, etc.

Definición de clases ES6

```
class Persona {
  constructor(nombre, edad) {
    this.nombre = nombre;
    this.edad   = edad;
  }
  get verNombre() {
    return this.nombre;
  }
  set nuevoNombre(nuevo) {
    this.nombre = nuevo;
  }

  presentarse() {
    return 'Hola me llamo ' + this.nombre +
           ' y tengo ' + this.edad + ' años';
  }
}
var Sergio = new Persona('Sergio', 22);
alert(Sergio.verNombre); // devuelve Sergio
Sergio.nuevoNombre('Daniel'); //cambia el a Daniel
alert(Sergio.verNombre); // devuelve Daniel
alert(Sergio.presentarse());
```

Herencia usando ES6

```
class Desarrollador extends Persona {
  constructor(nombre, edad, cargo) {
    super(nombre, edad);
    this.cargo = cargo;
  }

  presentarse() {
    return super.presentarse() + ' y soy desarrollador ' + this.cargo;
  }
}

var Sergio = new Desarrollador('Sergio', 22, 'Frontend');
Sergio.presentarse();
//'Hola me llamo Sergio y tengo 22 años y soy desarrollador Frontend'
```

- EJERCICIO 3 y 4: Realizar la clase hotel y la clase alojamiento usando ES6 en lugar de prototipos.

Fin del Tema 3

Bibliografía

- Gauchat, Juan Diego: **“El gran libro de HTML5, CSS3 y Javascript”**.Editorial Marcombo. 2012
- Vara, J.M. y otros: **“Desarrollo Web en Entorno Cliente. CFGS”**.Editorial Ra-Ma. 2012
- **“Programación en Javascript”.** Colección de artículos disponibles en la url. http://www.desarrolloweb.com/manuales/ Última visita: Septiembe 2017.
- **W3SCHOOL “Manual de referencia y Tutoriales”** http://www.w3schools.com/ Última visita Septiembre 2017
- **Comesaña, J.L. : Técnico en Desarrollo de Aplicaciones Web.** http://www.sitiolibre.com/daw.php Última visita Septiembre 2017