

Universidad Gerardo Barrios



Facultad:

Ciencia y tecnología

Materia:

Programación Computacional IV

Docente:

Ing. Gisela Espinoza

Estudiante:

Edwin Wilfredo Rodriguez Cabrera

Carrera:

Ingeniería en sistemas y redes informáticas

Fecha de entrega:


Domingo 23 de mayo

Módulos en NodeJs

En Node.js, un **módulo** es un conjunto de funciones y objetos de JavaScript que las aplicaciones externas pueden usar.


¿Cómo heredar un módulo en NodeJs?

Partiremos del siguiente ejemplo, crearemos un constructor de que reciba un número y un Código



```
1 function Number(number, code){
2   this.number=number,
3   this.code=code
4 }
```

Luego crearemos un constructor llamado persona, para agregar un nombre al constructor anteriormente creado haciendo referencia a él con la función "call".



```
1 function Persona(name, number, code) {
2   this.name = name;
3   Number.call(this, number, code)
4
5
6 }
7
```

Esta función básicamente permite llamar a una función definida en otro lugar, pero con el contexto actual, el primer parámetro indica que desea utilizar al ejecutar la función y los otros son aquellos que deben pasarse cuando la función se invoca.

Por último, creamos un nuevo objeto con el valor de Persona, este nuevo objeto tiene Number como prototipo por lo tanto lo heredará

```
1 Persona.prototype = Object.create(Number.prototype);
2 Persona.prototype.constructor = Persona;
3
4 var profesor1 = new Persona('Wilfredo', '7', 'G')
5 const nameP = profesor1
6
7 exports.nameP = nameP;
```

Luego podemos probar el resultado de nuestro código

```
1 const test = require('./example');
2
3 console.log(test.nameP);
```

Haciendo uso de require en nuestro archivo y haciendo uso de la constante creada para ver el resultado de la variable profesor1 que hemos creado.

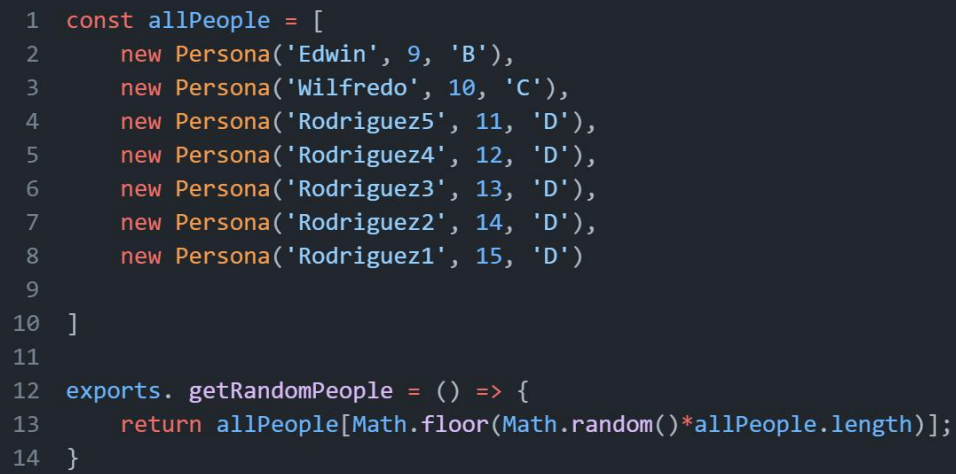
Obteniendo lo siguiente:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\numbers> node test.js
{ name: 'Wilfredo', number: '7', code: 'G' }
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\numbers> █
```

También podemos crear una dependencia en nuestro proyecto de nuestros módulos de la siguiente manera si nosotros lo necesitamos.

Para el ejemplo crearemos lo siguiente



```
1  const allPeople = [  
2    new Persona('Edwin', 9, 'B'),  
3    new Persona('Wilfredo', 10, 'C'),  
4    new Persona('Rodriguez5', 11, 'D'),  
5    new Persona('Rodriguez4', 12, 'D'),  
6    new Persona('Rodriguez3', 13, 'D'),  
7    new Persona('Rodriguez2', 14, 'D'),  
8    new Persona('Rodriguez1', 15, 'D')  
9  ]  
10 ]  
11  
12 exports.getRandomPeople = () => {  
13   return allPeople[Math.floor(Math.random()*allPeople.length)];  
14 }
```

La función junto con el export obtendrá una persona del arreglo creado anteriormente de manera aleatoria.

Iremos a nuestro proyecto y crearemos una nueva carpeta e iniciaremos el package con el siguiente comando npm init -y

```
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\numbers> cd..
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18> mkdir exampleWork

Directorio: E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18

Mode                LastWriteTime         Length Name
----                -
d-----          23/5/2021   19:08                exampleWork

PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18> cd exampleWork
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> npm init -y
Wrote to E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork\package.json:

{
  "name": "exampleWork",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
```

E instalaremos como dependencia nuestro archivo creado anteriormente, navegando hacia la carpeta en donde se encuentra el archivo

```
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> npm install --save ../numbers
```

Ahora navegamos hacia nuestro package.json

```
1 {
2   "name": "exampleWork",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "dependencies": {
13    "numbers": "file:../numbers"
14  }
15 }
16
```

Y veremos que las dependencias se han modificado
Crearemos un archivo index.js y lo probaremos

```
1 const test = require('numbers');
2 const randomPeople = test.getRandomPeople()
3 console.log(randomPeople);
```

Obteniendo como resultado lo siguiente

```
Persona { name: 'Rodriguez5', number: 11, code: 'D' }  
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> node index.js  
Persona { name: 'Rodriguez4', number: 12, code: 'D' }  
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> node index.js  
Persona { name: 'Rodriguez1', number: 15, code: 'D' }  
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> node index.js  
Persona { name: 'Rodriguez3', number: 13, code: 'D' }  
PS E:\Wilfredo\Documents\ProgramacionIII\NodeJS\practicaSemana18\exampleWork> 
```