

Ejercicio 1. Ámbito de las variables en Javascript

1. ¿Cuál es la diferencia entre global scope, function scope y block scope en Javascript? Compruébalo con código de ejemplo como el de este enlace: https://www.w3schools.com/js/js_let.asp

Global scope son variables declaradas globalmente que se pueden acceder fuera de la función.

Function scope son variables que se declaran en local y solo se accede dentro de la función.

Block scope no pueden ser accedidas fuera de las llaves de la función.

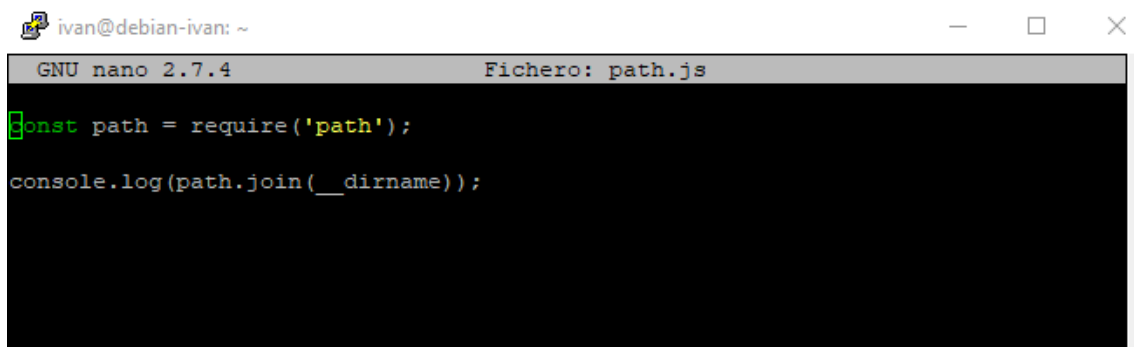
2. ¿Cuál es la diferencia entre let y var?

Let son variables que tienen ámbito en el bloque, no pudiendo acceder globalmente.

Var son variables que tienen un ámbito global pudiendo acceder desde fuera.

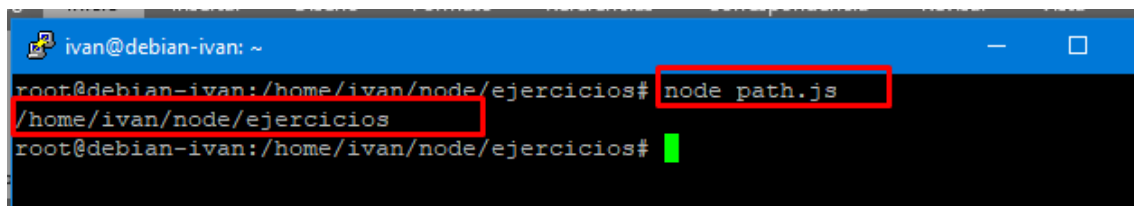
Ejercicio 2.

Crea un pequeño programa que muestre el directorio de trabajo actual, similar al comando pwd de Linux. Consulta la API: <https://nodejs.org/api/path.html>



A terminal window titled 'ivan@debian-ivan: ~' shows the GNU nano 2.7.4 editor editing 'Fichero: path.js'. The code in the editor is:

```
const path = require('path');  
console.log(path.join(__dirname));
```



A terminal window titled 'ivan@debian-ivan: ~' shows the execution of the script. The prompt is 'root@debian-ivan:/home/ivan/node/ejercicios#'. The command 'node path.js' is entered and executed, resulting in the output '/home/ivan/node/ejercicios'.

Ejercicio 3.

1. Crea un servidor web en Node que lea un fichero HTML (about.html) y devuelva el contenido al navegador. El archivo HTML debe incluir tu nombre y apellidos entre etiquetas <h1>.

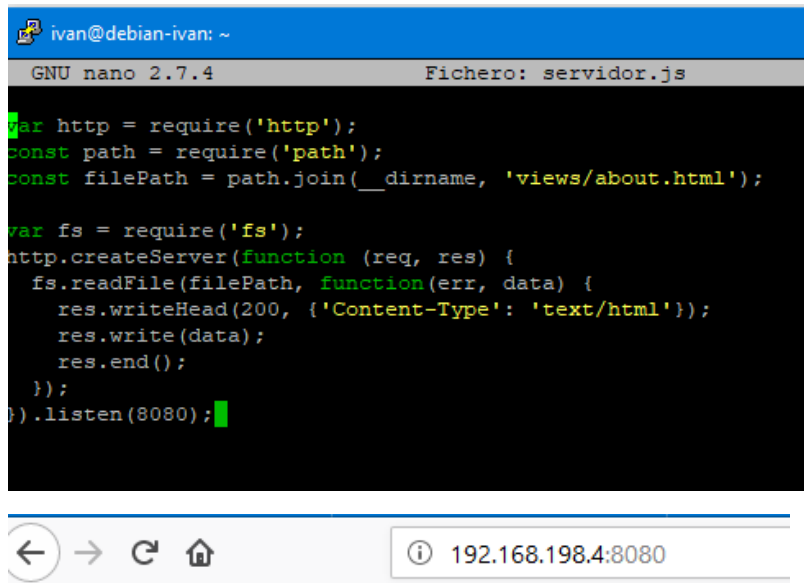
```
ivan@debian-ivan: ~  
GNU nano 2.7.4 Fichero: servidor.js  
var http = require('http');  
var fs = require('fs');  
http.createServer(function (req, res) {  
  fs.readFile('about.html', function(err, data) {  
    res.writeHead(200, {'Content-Type': 'text/html'});  
    res.write(data);  
    res.end();  
  });  
}).listen(8080);
```



Iván Hernández Fuentes

```
ivan@debian-ivan: ~  
GNU nano 2.7.4 Fichero: about.html  
<html>  
<head>  
<meta charset='utf-8'>  
</head>  
<body>  
<h1>Iván Hernández Fuentes</h1>  
</body>  
</html>
```

- 2 Ahora crea una carpeta que se llame views que almacenará el fichero html anterior. La carpeta views es sólo un nombre común para almacenar vistas en el patrón de diseño MVC. Modifica el programa para que lea el fichero de la carpeta.



```
ivan@debian-ivan: ~
GNU nano 2.7.4 Fichero: servidor.js

var http = require('http');
const path = require('path');
const filePath = path.join(__dirname, 'views/about.html');

var fs = require('fs');
http.createServer(function (req, res) {
  fs.readFile(filePath, function(err, data) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    res.end();
  });
}).listen(8080);
```

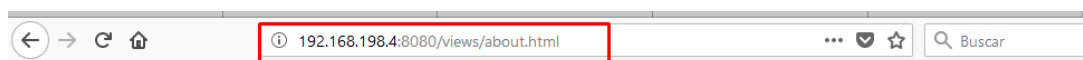
Iván Hernández Fuentes

Ejercicio 4.

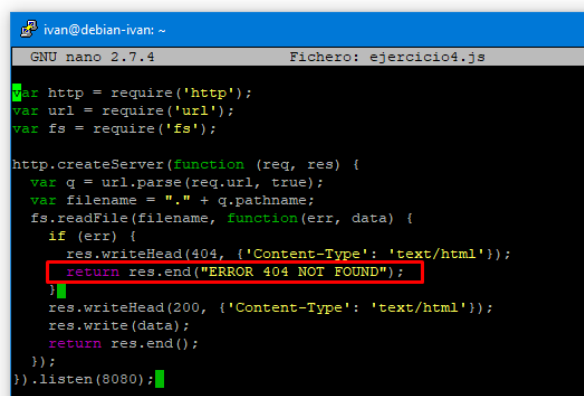
Crea un programa que utilice la parte de la URL que hace referencia a la página web para mostrar el fichero HTML correspondiente.

Si el fichero no existe el objeto err no será nulo. En este caso hay que mostrar una página 404 de error.

Este ejercicio es, de hecho, un Apache rudimentario. Por supuesto no soporta tipos mime y la seguridad es inexistente. Cuando veamos el framework express veremos cómo ampliar sus funcionalidades.



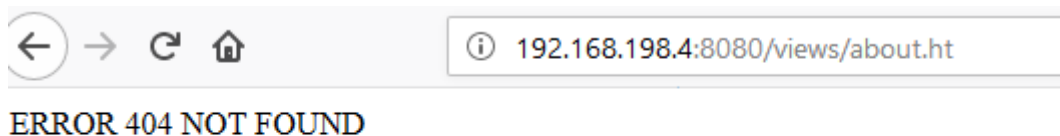
Iván Hernández Fuentes



```
ivan@debian-ivan: ~
GNU nano 2.7.4 Fichero: ejercicio4.js

var http = require('http');
var url = require('url');
var fs = require('fs');

http.createServer(function (req, res) {
  var q = url.parse(req.url, true);
  var filename = "." + q.pathname;
  fs.readFile(filename, function(err, data) {
    if (err) {
      res.writeHead(404, {'Content-Type': 'text/html'});
      return res.end("ERROR 404 NOT FOUND");
    }
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
  });
}).listen(8080);
```



Ejercicio 5.

1. Averigua cómo se desinstala un paquete con npm. ¿Cómo puedes ver qué módulos están instalados localmente para un proyecto? ¿Y Globalmente?

Para desinstalar un paquete utilizamos `npm uninstall (nombre)`

Localmente con `npm list`

```
root@debian-ivan:/# npm list
/
└─ (empty)
```

Globalmente con `npm list -g`

```
root@debian-ivan:/# npm list -g
/usr/lib
└─ npm@6.4.1
   └─ abbrev@1.1.1
      └─ ansicolors@0.3.2
         └─ ansistyles@0.1.3
            └─ aproba@1.2.0
               └─ archy@1.0.0
                  └─ bin-links@1.1.2
                     └─ bluebird@3.5.1 deduped
                        └─ cmd-shim@2.0.2 deduped
                           └─ gentle-fs@2.0.1 deduped
                              └─ graceful-fs@4.1.11 deduped
                                 └─ write-file-atomic@2.3.0 deduped
                                    └─ bluebird@3.5.1
                                       └─ byte-size@4.0.3
                                          └─ cacache@11.2.0
                                             └─ bluebird@3.5.1 deduped
                                                └─ chownr@1.0.1 deduped
                                                   └─ figgy-pudding@3.4.1 deduped
                                                      └─ glob@7.1.2 deduped
                                                         └─ graceful-fs@4.1.11 deduped
                                                            └─ lru-cache@4.1.3 deduped
                                                               └─ mississippi@3.0.0 deduped
                                                                  └─ mkdirp@0.5.1 deduped
                                                                     └─ move-concurrently@1.0.1 deduped
                                                                        └─ promise-inflight@1.0.1 deduped
                                                                           └─ rimraf@2.6.2 deduped
                                                                              └─ ssri@6.0.0 deduped
                                                                                 └─ unique-filename@1.1.0 deduped
                                                                                    └─ y18n@4.0.0
                                                                                       └─ call-limit@1.1.0
                                                                                          └─ chownr@1.0.1
                                                                                             └─ ci-info@1.4.0
                                                                                                └─ cli-columns@3.1.2
                                                                                                   └─ string-width@2.1.1
                                                                                                      └─ is-fullwidth-code-point@2.0.0
                                                                                                         └─ strip-ansi@4.0.0
                                                                                                            └─ ansi-regex@3.0.0
```

2. Crea una carpeta para un nuevo proyecto. Ejecuta ahora `npm init -y` para inicializar el proyecto. Dentro de tu carpeta verás un fichero `package.json`.

¿Cuál crees que es su funcionalidad? Para ver la configuración del proyecto nombre del autor, versión, etc...

```
root@debian-ivan:/home/ivan/node/ejercicios/proyecto# npm init -y
Wrote to /home/ivan/node/ejercicios/proyecto/package.json:

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

3. Instala el paquete upper-case. Vuelve a mirar el fichero package.json. ¿Ves alguna diferencia? ¿Dónde se ha instalado el paquete?

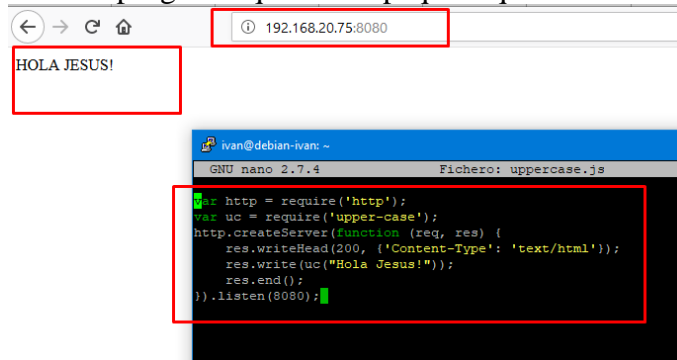
Se ha creado una carpeta node_modules y un archivo package-lock.json

```
root@debian-ivan:/home/ivan/node/ejercicios/proyecto# ls
node_modules package.json package-lock.json
```

Entramos en package.json y vemos como se han añadido dependencias nuevas (upper-case)

```
{
  "name": "proyecto",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "upper-case": "^1.1.3"
  }
}
```

4. Crea un programa que use el paquete que acabas de instalar.



The screenshot shows a web browser window with the address bar displaying '192.168.20.75:8080' and the page content 'HOLA JESUS!'. Below the browser, a terminal window shows the code for 'uppercase.js' using the 'http' and 'upper-case' packages to create a simple web server that responds with 'HOLA JESUS!'.

```
var http = require('http');
var uc = require('upper-case');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write(uc("Hola Jesus!"));
  res.end();
}).listen(8080);
```

- 5.

5 Desinstala el paquete upper-case.

```
root@debian-ivan:/home/ivan/node/ejercicios/proyecto# npm uninstall upper-case
npm WARN proyecto@1.0.0 No description
npm WARN proyecto@1.0.0 No repository field.

removed 1 package in 1.457s
found 0 vulnerabilities
```