GIT BASH

|  |  |
| --- | --- |
| COMANDOS | **Bash** |
| Para saber dónde estoy | pwd |
|  | /carpeta/ |
| Subir al nivel superior de carpeta (change directory) | cd .. |
| Listar contenido accesible desde mi ubicación | ls |
| Make a directory | mkdir |
| Remove files | rm |
| Remove carpeta |  |
| Carácter comodín o wildcard | \* |
| Crear cualquier tipo de fichero | touch nombre.extension |
| Borrar una carpeta y todos sus ficheros | Rm –r nombreCarpeta |
| Mover: copiar y pegar | cp rutaOrigen rutaDestino/newName |
| Mover una carpeta dentro de otra (cortar pegar) | mv folder1 folder2 |
| Para ver las opciones de cada comando | Comando –help |
| Ejecutar un JS script con Node desde la bash | node fichero.js |

[www.npmjs.com](http://www.npmjs.com) Node Package Manager

Creamos la carpeta del nuevo proyecto, entramos en ella y ejecutamos el primer comando:

1. **mkdir newProject**
2. **cd newProject/**
3. **npm init**
4. **npm install bootstrap**
5. **npm install jquery**
6. **npm install Popper.js**

El archivo package.json tiene una key llamada dependencias con las librerías que queremos adjuntar a nuestro proyecto. Copiamos el archivo package.json en un nuevo proyecto y ejecutamos el comando npm install

Automáticamente se actualizan mis rescursos de terceros a la subversión más avanzada de la versión que se indique en el archivo package.json

Concepto de pipe:

Expresiones regulares

Gestiones de procesos

**ng –v** Ver las versiones del comando ng **ng --version**

Bower: gestor de paquetes específico para cosas de Front. Igual que npm pero con algunos recursos específicos de front que no se pueden encontrar en el repositorio npm

**npm install –g bower** 🡪 Instala bower a nivel global (-g) en la raíz del sistema operativo

Creamos un bower.json con las dependencias que queramos utilizar. Finalmente, igual que con npm ejecutamos el comando **bower install**.

# Crear un servidor node

-------------------

Instala Lite-Server

-------------------

- Abre una consola de comandos

- Instala lite-server:

**npm install -g lite-server**

------------------

Proyecto de prueba

------------------

- Crea un directorio con el nombre "prueba1"

- Crea un proyecto node:

**npm init –y**

Crea una página index.html con un texto "Hola Mundo" en el proyecto

- Modifica el proyecto (package.json) para que en "start" lance lite-server:

"scripts": {

"start": "lite-server"

},

---------------------------

Iniciar y parar el servidor

---------------------------

*- Lanza el proyecto con:*

- En la consola escribe: **npm start**

- Se iniciará el servidor en la dirección:

http://localhost:3000/

- Revisa la página en el navegador

*- Modifica el proyecto:*

- Modifica index.html para que diga "Hola Mundo!!!"

- Observa cómo se recarga el navegador (ÉL SOLITO!!) con la nueva versión

- Para el proyecto con:

- Ve a la consola y pulsa **CTRL+C**. Acepta la finalización pulsando "S". El

Servidor parará

- Verifica el navegador

Lite-server es un paquete que escucha por el puerto 3000 ó 300x.

cat nombreArchivo.extensión 🡪 muestra el contenido en la consola

vim nombreArchivo.extensión 🡪 te muestra el contenido en la consola en formato texto y se puede modificar y editar

crtl + c

:qa!

-g significa global y ya se queda instalado para siempre

Todos los plugins que introduzcamos en el JSON dependencies del JSON package, node lo saca del gestor de paquetes npm.

npm install 🡪 ejecuta el json package e instala todos los plugins del sub JSON dependencies

[**https://docs.npmjs.com/getting-started/semantic-versioning**](https://docs.npmjs.com/getting-started/semantic-versioning)

<https://blog.risingstack.com/your-first-node-js-http-server/>

<https://jesuslc.com/>

<http://blog.koalite.com/2015/06/grunt-o-gulp-que-uso/>

<https://geekytheory.com/node-js-primeros-pasos-y-hola-mundo>

YEOMAN 🡪 scaffolding

Comando: **yo**

Comprar un dominio, configurar las DNS para que apunten al router de la máquina servidor y permitir la entrada por el puerto 80, además de establecer un IP fija.

# Typescript

Optional: Open greeter.ts in Visual Studio, or copy the code into the TypeScript playground. You can hover over identifiers to see their types. Notice that in some cases these types are inferred automatically for you. Re-type the last line, and see completion lists and parameter help based on the types of the DOM elements. Put your cursor on the reference to the greeter function, and hit F12 to go to its definition. Notice, too, that you can right-click on a symbol and use refactoring to rename it.

The type information provided works together with the tools to work with JavaScript at application scale. For more examples of what’s possible in TypeScript, see the Samples section of the website.