

Diagnóstico

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Diagnóstico

- Crearse una cuenta en [Quizizz](#).
- Realizar el quiz de node, npm y teoría de APIs.



Npm

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Node Package Manager es un manejador de paquetes de node (el más popular de JavaScript). Permite para compartir e instalar paquetes.

Se compone de 2 partes:

- Un **repositorio online** para **publicar y descargar paquetes** de software libre.
- Una **CLI** para la terminal **que permite interactuar** instalar, gestionar y publicar paquetes.

NOTA: Se puede considerar un gestor de dependencias de proyectos de tipo npm.

Comandos de Npm

Inicialización de un proyecto npm

- **npm init:** Inicializa una carpeta como un proyecto de npm.

Levantar un proyecto npm

- **npm start:** Permite “iniciar” un proyecto de node. Es el único comando por defecto que no requiere la palabra run (**npm run other-command**).

Comandos de Npm

Instalar/desinstalar dependencias

- **npm i**: Instala todas las dependencias del package.json
- **npm install -g <package-name>**: Instala un paquete globalmente.
- **npm install <package-name>**: Instala un paquete y lo agrega al package.json
- **npm install --save <package-name>**: Instala un paquete y lo agrega al package.json
- **npm install -D <package-name>**: Instala un paquete y lo agrega al package.json en la parte de devDependencies.
- **npm install --save-dev <package-name>**: Instala un paquete y lo agrega al package.json en la parte de devDependencies.

Comandos de Npm

Gestión de dependencias

- `npm uninstall <package-name>`
- `npm -g uninstall <package-name>`
- `npm search <package-name>`
- `npm ls`
- `npm update -save`
- `npm list`
- `npm list -g --depth 0`
- `npm outdated`

Paquetes

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Paquetes

Son módulos distribuidos en forma de librerías que resuelven alguna necesidad de desarrollo.

A continuación se listan algunos de los módulos:

- npm.
- expressjs.
- socket.
- mongoose.
- mocha.
- create-react-app.
- vite.

NOTA: Tu también puedes desarrollar tus propios paquetes y publicarlos.

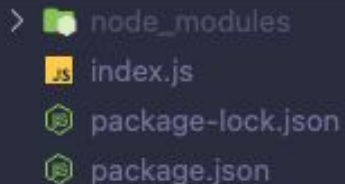
Scaffold Npm

DEV.FX
DESARROLLAMOS(PERSONAS);

dev

Estructura de proyecto npm

- **node_modules:** Carpeta donde se instalan las dependencias de un proyecto npm, normalmente esta carpeta se agrega al .gitignore.
- **package.json:** Guardan las dependencias y los comandos de node.
- **package-lock.json:** Guarda un snapshot de las dependencias que se instalaron en un determinado momento.



A screenshot of a file explorer window with a dark background. It shows a directory structure with the following items: a folder named 'node_modules' with a green folder icon, a file named 'index.js' with a yellow JavaScript icon, and two files named 'package-lock.json' and 'package.json' both with green JSON icons.

```
> node_modules  
index.js  
package-lock.json  
package.json
```

package.json

Este archivo guarda las dependencias y los comandos de node.

- **name:** Nombre que se le asigna al proyecto, generalmente es el mismo nombre de la carpeta y el repositorio.
- **versión:** Se utiliza para conocer la versión del proyecto, el estándar para versionar es SEMVER.
- **description:** Meta data sobre lo que trata el proyecto.
- **license:** Tipo de licencia de software del proyecto.

package.json

Este archivo guarda las dependencias y los comandos de node.

- **scripts:** Para declarar comandos largos que después se pueda ejecutar con `npm run <commande>`.
- **devDependencies:** Son dependencias que sólo se instalan en el entorno local.
- **dependencies:** Son dependencias que se instalan en cualquier entorno (local, test, qa, producción y pro).

package-lock.json

- Da visibilidad de los cambios en el árbol de dependencias.
- Garantiza la conciliación de dependencias.
- Se autogenera cada que un comando modifica el `node_modules` o el `package.json`.
- Es usualmente generado por el comando `npm install`.
- Este archivo tiene las versiones exactas de las dependencias utilizadas por un proyecto npm.
- Debe subirse al repositorio.

Scripts

DEV.F
DESARROLLAMOS(PERSONAS);

Scripts

Son comandos propios que se pueden agregar al package.json para poderlos ejecutar con **npm run <my-comand>**.

```
"scripts": {  
  "test": "echo \"Error: no test specified\" && exit 1",  
  "start": "node index.js",  
  "dev": "nodemon index.js"  
},
```