NPM
(Node Package Manager)

DEV.F::
DESARROLLAMOS(PERSONAS);

Elaborado por: César Guerra

¿Qué es NPM?

NPM son las siglas de Node Package Manager, es decir, **gestor de paquetes de Node.js.**

Este gestor de paquetes (muy similar al concepto de apt-get en GNU/Linux), nos permitirá instalar de forma muy sencilla y automática paquetes Javascript.

Estos paquetes nos brindan superpoderes para realizar tareas muy específicas sin que nosotros tengamos que hacer su programación de 0.

DEV.F::

Podemos ejecutar el comando:
npm --version ó
npm -v

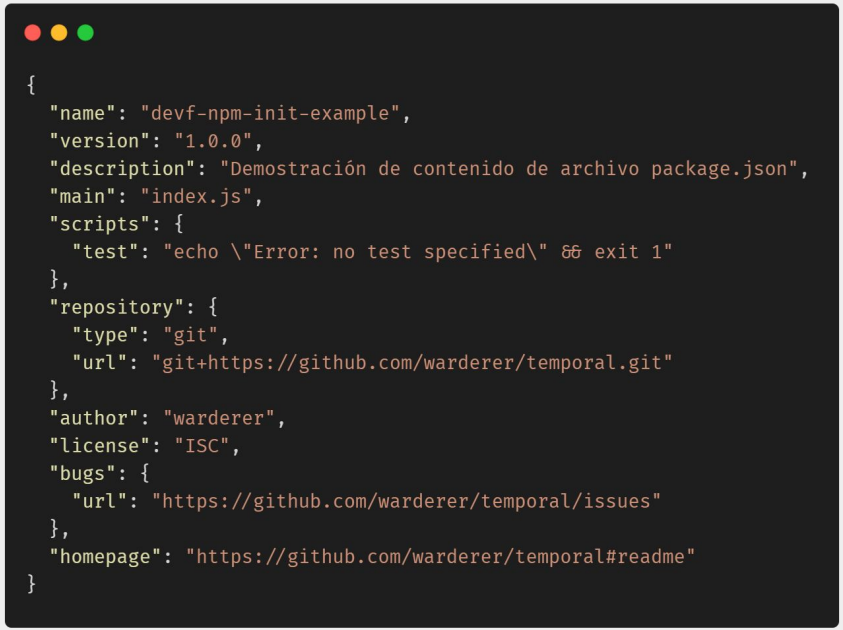
Para averiguar si tenemos npm instalado, de ser así nos regresará un número de versión

¿Tengo NPM instalado?

Al instalar Node.js, este instalará también consigo una serie de utilidades, incluidos su gestor de paquetes: npm.

Por lo que **si ya instalaste Node.js, ya debes tener npm**.

DEV.F::



```
{
  "name": "devf-npm-init-example",
  "version": "1.0.0",
  "description": "Demostración de contenido de archivo package.json",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/warderer/temporal.git"
  },
  "author": "warderer",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/warderer/temporal/issues"
  },
  "homepage": "https://github.com/warderer/temporal#readme"
}
```

Ejemplo de archivo package.json

package.json

Para usar paquetes, tu proyecto debe contener un archivo llamado package.json. Dentro de ese archivo, encontrarás metadatos específicos para los proyectos, como nombre del proyecto, autor, versión, licencia, etc.

Dentro de este archivo también se crea una lista de dependencias (paquetes de npm) y scripts que se pueden ejecutar en el proyecto.

Este archivo normalmente se crea en la raíz del proyecto.

DEV.F::

npm init

```
D:\warde\Documents\GitHub\2021\temporal>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (temporal) DevF NPM Init Example
Sorry, name can only contain URL-friendly characters and name can no longer contain capital letters.
package name: (temporal) devf-npm-init-example
version: (1.0.0)
description: Demostración de contenido de archivo package.json
entry point: (index.js)
test command:
git repository: (https://github.com/warderer/temporal.git)
keywords:
author:
license: (ISC)
About to write to D:\warde\Documents\GitHub\2021\temporal\package.json:

{
  "name": "devf-npm-init-example",
  "version": "1.0.0",
  "description": "Demostración de contenido de archivo package.json",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/warderer/temporal.git"
  },
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/warderer/temporal/issues"
  },
  "homepage": "https://github.com/warderer/temporal#readme"
}


Is this OK? (yes) yes

D:\warde\Documents\GitHub\2021\temporal>
```

Inicializando npm: Creando un archivo package.json

Afortunadamente npm viene con una herramienta que nos ayuda a crear un archivo package.json en vez de tener que hacerlo nosotros de forma manual.

Para ello debemos usar el comando:

npm init

DEV.F::

DEV.F::

PRO TIP

Si usamos el argumento -y al ejecutar npm init, este saltara todo el proceso de preguntas y creará un archivo package.json con la información mínima colocada, es decir:

npm init -y

Modos de Trabajo de NPM

DEV.F:
DESARROLLAMOS(PERSONAS);

Casos de uso de NPM

A NIVEL PROYECTO

1. Es la modalidad más usada.
2. Gestiona las dependencias usadas por proyecto (instalar, actualizar, eliminar).
3. Las dependencias son asociadas solo a la carpeta de cada proyecto.
4. Facilita compartir el proyecto con otras personas e inicializar el proyecto.

A NIVEL GLOBAL: -g

1. Es usada en situaciones específicas.
2. Se instalan a nivel sistema y no por proyecto, por lo que están disponibles siempre en cualquier lugar.
3. Normalmente se trata de aplicaciones de línea de comandos (CLI) qué se usan desde terminal.
4. Solo es necesario instalar 1 vez.

Gestión de Dependencias con NPM

DEV.F:
DESARROLLAMOS(PERSONAS);

Resumen de Comandos NPM (1)

Inicializar

npm init

npm init -y //crear rápido

Nos ayuda a crear el archivo de configuración de npm: package.json en la carpeta del proyecto.

$ npm init

Instalar un paquete

npm install <paquete>

npm i <paquete>

Este comando instala un paquete o dependencia a nivel proyecto. Ejemplo:

$ npm i request

Descargar paquetes/dependencias

npm install

Cuando existe un archivo package.json y queramos descargar e instalar las dependencias en nuestro proyecto, usamos este comando. Generalmente lo hacemos cuando descargamos el proyecto de otra persona.

$ npm install

Instalar un paquete (global)

npm install -g <paquete>

npm i -g <paquete>

Este comando instala un paquete o dependencia a nivel global. Ejemplo:

*$ npm i **-g** nodemon*

DEV.F

PRO TIP

Si estas en entorno Mac, es posible que al ejecutar npm para instalar un paquete marque un error de permisos, si es tu caso, usa el comando ***sudo,*** *por ejemplo:*

sudo npm i <paquete>

Resumen de Comandos NPM (2)

Actualizar

npm update <paquete>

npm update -g <paquete> //global

Busca las últimas versiones de paquetes y actualiza contra las versiones instaladas actualmente

$ npm update express

Desinstalar

npm uninstall <paquete>

Desinstala completamente un paquete, es decir borra los archivos y su entrada en el archivo package.json

$ npm uninstall express

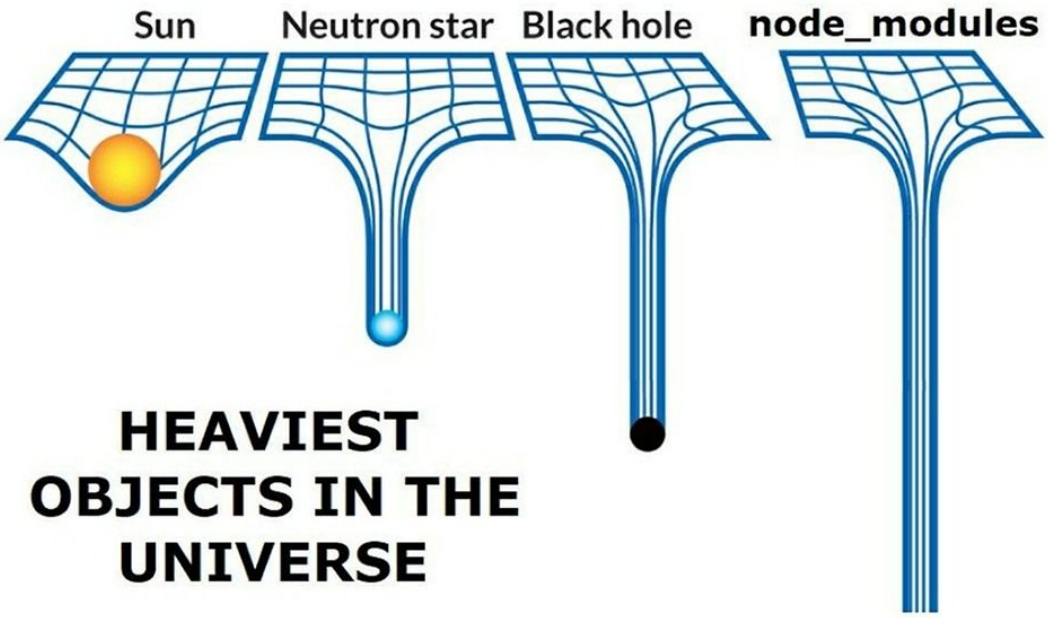
Qué son los paquetes?

Un paquete es uno o más archivos .js (módulos) agrupados (o empaquetados) juntos. Los archivos en un paquete son código reutilizable que realiza una función específica para tu aplicación Node.js

node_modules

DEV.F::
DESARROLLAMOS(PERSONAS);

Acerca de la carpeta node_modules



- Es la **carpeta donde npm descarga y almacena todas las dependencias** usadas a nivel proyecto.
- Es una carpeta muy pesada.
- Esta se crea específicamente para cada equipo por lo que **no debe ser compartida ni subida a un repositorio**. Ya qué se puede regenerar con el comando npm install siempre y cuando exista el archivo package.json

PRO TIP

Siempre que exista una carpeta node_modules es buena idea crear un archivo ***.gitignore*** *en la raíz del proyecto con contenido* ***node_modules***

Con el fin de que cuando subamos el proyecto al repositorio, no se suba node_modules.