## Console clear

console.clear() para poder limpiar la consola

## File system:

Se requiere importar el paquete de file system: const fs = require('fs');

Crear archivo desde el propio sistema:

### fs.writeFile(file, data[, options], callback):

Texto

Descripción generada automáticamente

Cuando se necesita utilizer una función de un archivo en especifico, en otro archivo, se de exportar:

Texto

Descripción generada automáticamente

Como se exporto un objeto, desde donde se llama, se puede desestructurar el mismo:



Colocando un async y un try-catch para manejar el error

Texto

Descripción generada automáticamente

Se genera una función tipo promesa:

Texto

Descripción generada automáticamente

### Recibir argumentos mediante la consola:

En el ts: 

En la consola: 

Salida:

Texto

Descripción generada automáticamente

Donde esta node, path donde esta la aplicación, argumento que se envio

Texto

Descripción generada automáticamente

Mediante consola se envia el comando: node app --base=9 y crea la tabla del 9

El anterior procedimiento es de manera educativa, no se recomienda

## Creación del package.json

De manera manual:

Se coloca en la consola npm init, siguiendo los pasos del vídeo, se crea el siguiente archivo:

Texto

Descripción generada automáticamente

El package.json es el punto inicial de cualquier proyecto de Node, nos ayuda y ayuda a Node a saber que comandos se ejecutan para ejecutar su aplicación.

Cuando se instala algún paquete desde la consola por npm, en el package:

Texto

Descripción generada automáticamente

Se crean las dependencias, y las dependencias son paquetes necesarios para que se ejecute la aplicación.

Luego de que se instalo esta dependencia, se creo un package-lock.json, y ahí se especifica como están construidas las dependencias que están en el package.json

Y también se crearon los módulos de node.

También se pueden instalar paquetes solo para un ambiente en desarrollo, se puede colocar, npm i nodemon --save-dev, en el package.json crea lo siguiente:

Texto

Descripción generada automáticamente

En el node modules están las dependencias, lo que se necesita para que se ejecute algo, como nodemon o los colores

Si se quiere instalar una versión en específico, se coloca: npm i [colors@1.0.0](mailto:colors@1.0.0)

Con npm update actualiza las dependencias

## Yargs

Paquete para entrada de comandos por consola:



Argv viene con lo de yarg, por lo tanto la salida de consola:

Texto

Descripción generada automáticamente

Mucho mas fácil de leer

## Configuración yargs

Crear opciones: Es decir, cuando se coloque b, la consola va a saber que b también es la base



La anterior salida de consola demuestra que como sabe que b también es base, le asigna también el valor de la base, también se puede colocar demandoption: true para que el valor de la base se coloque obligatorio cuando se escribe el comando

Pantalla de un video juego

Descripción generada automáticamente con confianza media

También se puede hacer un check de lo que se envia por la terminal, se verifica las condiciones que se envían y se puede colocar lógica.

En este caso, lo que se envia en la base tiene que ser un número, por lo tanto:

Texto

Descripción generada automáticamente

Con el isNaN se puede validar si no es un número y mandar ese error por consola