

# Sequelize y su configuración

# Índice

1. [Instalación](#)
2. [Rutas y directorios](#)
3. [Configuración](#)
4. [Objeto DB](#)

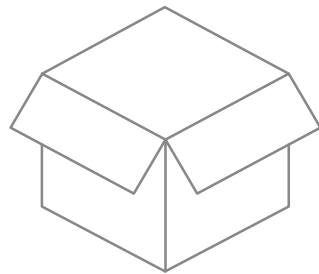


Sequelize es un ORM que nos ayuda a conectarnos e interactuar con bases de datos como Postgres, MySQL, MariaDB, SQLite, Microsoft SQL Server y más.



# Preparando el **proyecto**

Antes de instalar **Sequelize**, debemos tener en cuenta que al ser un paquete utilizado por **Node.js** vamos a tener que utilizar **npm**.



# 1 | Instalación

# Instalación

Dentro de la carpeta del proyecto de Node.js, hay que ejecutar los siguientes comandos:

```
npm install sequelize-cli -g  
  
>_ npm install sequelize  
  
npm install mysql2
```

# 2 | Rutas y directorios

# Rutas y directorios

Una vez instalados los paquetes que necesitamos, debemos establecer las rutas y directorios. Para ello, debemos crear un archivo llamado `.sequelizerc` en la raíz del proyecto y, dentro de este, escribir lo siguiente:

```
const path = require('path')

module.exports = {
  config: path.resolve('./database/config', 'config.js'),
  'models-path': path.resolve('./database/models'),
  'seeders-path': path.resolve('./database/seeders'),
  'migrations-path': path.resolve('./database/migrations'),
}
```



# Iniciar **Sequelize** en el proyecto

Para que Sequelize cree todas las carpetas y archivos que necesitamos para comenzar a trabajar con él, debemos correr el siguiente comando:

```
>_  sequelize init
```

# 3 | Configuración

# Configuración

Por último, debemos configurar la conexión con la base de datos. En las carpetas que creó Sequelize, encontraremos el archivo **config.js** en la ruta **/database/config/config.js**. Dentro de este, encontramos un JSON con credenciales por defecto que debemos reemplazar por las nuestras.

```
{  
  "development": {  
    "username": "root",  
    "password": "Monito123!",  
    "database": "movies_db",  
    "host": "127.0.0.1",  
    "dialect": "mysql",  
    "operatorsAliases": false  
  }  
}
```

# ¡ATENCIÓN!



Necesitamos agregar un pequeño detalle para que no nos encontremos con un problema a la hora de requerir Sequelize. Para esto, en el archivo `config.js` que editamos anteriormente debemos asignar todo el JSON que modificamos a **`module.exports`**.

```
JS config.js ×
database > config > JS config.js > [?] <unknown>
1  module.exports = {
2    "development": {
3      "username": "root",
4      "password": "Monito123!",
5      "database": "movies_db",
6      "host": "127.0.0.1",
7      "dialect": "mysql",
8      "operatorsAliases": false
9    },|
```

# 4 | Objeto DB

# Objeto **DB**

Como dato curioso, les contamos que al final del archivo **index.js**, ubicado en **/database/models/index.js**, encontramos la exportación del objeto **DB**. Este será al que llamaremos cada vez que queramos utilizar Sequelize para realizar consultas a nuestra base de datos.



# Documentación



**Para saber más podemos acceder a la documentación oficial de Sequelize haciendo click en el siguiente link:**

<https://sequelize.org/>

DigitalHouse>  
Coding School