
SENPAI
ACADEMY

Express

SENPAI
ACADEMY

Node.js - Express

Express es una librería rápida y minimalista para el desarrollo de aplicaciones web Node.js.

Nos proporciona un conjunto sólido de características que nos facilitan el desarrollo de servidores en NodeJS.

Permite usar casi cualquier template engine, sumando además utilidades para responder con varios formatos de datos, transferir archivos, ruteo de URLs y demás.

SENPAI
ACADEMY

Node.js - Express

Express dispone de utilidades HTTP que permiten crear APIs de manera fácil y rápida. (API === Application programming interface)

Es una pequeña capa, con mucha funcionalidad, lo que acelera el desarrollo. También tiene una alta performance.

Express no soluciona ni nos provee todo, seguramente vamos a necesitar utilizar algunas otras librerías para complementar.

Para utilizarlo es necesario instalarlo en nuestro proyecto, osea, desde una terminal parados en la raíz de nuestro proyecto:

```
npm install express
```

SENPAI ACADEMY

Express

```
// En App.js
const express = require('express');

const app = express();

app.get('/', function(req, res) {
  res.send('Hello');
});

app.listen(4000);
```

Una aplicación en Node.js con express puede ser tan pequeña como esto.

1. Requerimos la librería
2. Inicializamos la “app”
3. Definimos un get en la ruta “/” y le damos un callback para que ejecute cuando alguien la visite
4. Le decimos que escuche en el puerto 4000.

SENPAI ACADEMY

Express - Scaffolding (Estructura)

Express no nos fuerza a usar una estructura predefinida. Podemos dejar las rutas y assets en donde queramos.

Para mejorar la mantenibilidad de nuestra aplicación, se recomienda crear al menos las siguientes 3 carpetas:

- **/public**: Para mantener los recursos como imágenes, hojas de estilo, y JavaScript del cliente.
- **/routes**: Donde se definirán los manejadores de rutas de nuestra aplicación.
- **/views**: Donde estarán las vistas de nuestra aplicación, que dependen del template engine que utilicemos (React, Vue, etc)

SENPAI ACADEMY

Express - Routing

Cuando creamos una app con express, nos proporciona todos los verbos HTTP:

- *app.get()*
- *app.post()*
- *app.put()*
- *app.delete()*

También podríamos interceptar cualquier verbo utilizando:

- *app.all()*

SENPAI ACADEMY

Express - Routing

El primer parámetro es la URL que vamos a servir:

```
app.get('/',   );
app.get('/acerca_de',   );
app.get('/contacto',   );
app.post('/form',   );
```

Podemos también definir parámetros en la URL:

```
app.get('/book/:bookId',   );
app.get('/user/:userId/publications/:pubId',   );
```

SENPAI ACADEMY

Express - Routing

El segundo parámetro al definir una ruta es la función callback que queremos que Express ejecute cuando alguien visite esa ruta.

Este callback, Express lo va a invocar siempre con dos parámetros: **Request y Response**

```
app.get('/contacto', function (request, response) {

  response.statusCode = 200;
  response.setHeader("Content-Type", "text/html");
  response.end("<div> <h3>Pagina de Contacto</h3> </div>");

});
```

SENPAI ACADEMY

Express - Routing

Desde el objeto **request** podemos acceder a varios datos de la solicitud, entre ellos los parámetros:

```
app.get('/user/:userId/publications/:pubId', function (req, res) {
  res.send(`Recibi el userId: ${req.params.userId} y el pubId: ${req.params.pubId}`);
});
```

SENPAI ACADEMY

Express - Routing

Es importante el orden en que definimos los manejadores de rutas. Por ejemplo, acá definimos una ruta que engloba todas las requests que ya no tengan una ruta específica con **'/*'**

```
// En App.js
const express = require('express');
const app = express();

app.get('/users', function(req, res) {
  res.send('Llego a ' + req.originalUrl);
});

app.get('/*', function (request, response) {
  response.send("Mi aplicación esta corriendo");
})

app.listen(4000, function () {
  console.log('Callback del listen. La app quedó corriendo en el puerto 4000');
});
```

SENPAI ACADEMY

Express - Request y Response

Desde el parámetro ***request***, podemos acceder a:

- hostname
- body
- method
- baseUrl
- cookies
- params
- path
- ip
- query

Documentación de request

Desde el parámetro ***response***, podemos utilizar los siguientes métodos:

- append (<header-key>, <header-value>)
- cookie (<cookie-name>, <cookie-value>, options>)
- clearCookie (<cookie-name>)
- status (<statusCode>)
- send (<Buffer> | <String> | <Objeto> | <Array>)
- json (<jsonObject>)
- redirect (<statusCode>, <newUrl>)

Documentación de response

SENPAI ACADEMY

Express - Nodemon

Esta librería se instala a través de npm y nos sirve para estar escuchando cambios en nuestro código de Node.js, y que reinicie automáticamente el servidor.

Es una herramienta que nos va a ser muy útil durante el desarrollo.

La instalamos de forma global:

```
npm install -g nodemon
```

Y para correr nuestra aplicación con esta herramienta, lo hacemos:

```
nodemon app.js
```

Hagamos ahora un ejemplo, que pueden encontrar en Github

SENPAI ACADEMY

SENPAI
ACADEMY

gustavguez

gustavguez

GUSTAVO RODRIGUEZ

FULL STACK DEVELOPER
SOLCRE