Contenidos a Trabajar

1. Express, el framework más utilizado de Node.Js
2. Instalación
3. ¡Hola Mundo! en Express
4. Estructura de archivos

Express, el framework más utilizado de Node.Js

Express es el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web de Node populares.

Proporciona mecanismos para:

- Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).
- Integración con motores de renderización de "vistas" para generar respuestas mediante la introducción de datos en plantillas.
- Establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.
- Añadir procesamiento de peticiones "middleware" adicional en cualquier punto dentro de la tubería de manejo de la petición.

A pesar de que Express es en sí mismo bastante minimalista, los desarrolladores han creado paquetes de middleware compatibles para abordar casi cualquier problema de desarrollo web. Hay librerías para trabajar con cookies, sesiones, inicios de sesión de usuario, parámetros URL,

datos POST, cabeceras de seguridad y muchos más. Podés encontrar una lista de paquetes middleware mantenida por el equipo de Express en Express Middleware (junto con una lista de algunos de los paquetes más populares de terceros).

¿Dónde comenzó?

Express fue lanzado inicialmente en Noviembre de 2010 y está ahora en la versión 5.0 de la API.

Podés comprobar en el changelog la información sobre cambios en la versión actual, y en GitHub notas de lanzamiento históricas más detalladas.

¿Qué popularidad tiene Node/Express?

La popularidad de un framework web es importante porque es un indicador de si se continuará manteniendo y qué recursos tienen más probabilidad de estar disponibles en términos de documentación, librerías de extensiones y soporte técnico.

Una pregunta mejor es si Node y Express son lo “suficientemente populares" para evitar los problemas de las plataformas menos populares. ¿Continúan evolucionando? ¿Podés conseguir la ayuda que necesitas? ¿Hay alguna posibilidad de que consigas un trabajo remunerado si aprendes Express?

De acuerdo con el número de compañías de perfil alto que usan Express, el número de gente que contribuye al código base, y el número de gente que proporciona soporte tanto libre como pagado, podemos entonces decir que Express es un framework extremadamente popular.

¿Es Express dogmático?

Los frameworks web frecuentemente se refieren a sí mismos como "dogmáticos" ("opinionated") o "no dogmáticos" ("unopinionated").

Los frameworks dogmáticos son aquellos que opinan acerca de la "manera correcta" de gestionar cualquier tarea en particular. Ofrecen soporte para el desarrollo rápido en un dominio en particular (resolver problemas de un tipo en particular) porque la manera correcta de hacer cualquier cosa está generalmente bien comprendida y bien documentada. Sin embargo, pueden ser menos flexibles para resolver problemas fuera de su dominio principal, y tienden a ofrecer menos opciones para elegir qué componentes y enfoques pueden usarse.

Los frameworks no dogmáticos, en contraposición, tienen muchas menos restricciones sobre el modo mejor de unir componentes para alcanzar un objetivo, o incluso qué componentes deberían usarse. Hacen más fácil para los desarrolladores usar las herramientas más adecuadas para completar una tarea en particular, si bien al costo de que necesitas encontrar esos componentes por tu cuenta.

Express es no dogmático, transigente. Podés insertar casi cualquier middleware compatible que te guste dentro de la cadena de manejo de la petición, en casi cualquier orden que te parezca. Podés estructurar la app en un fichero o múltiples ficheros y usar cualquier estructura de directorios.Es válido también mencionar que algunas veces podés sentir que tenés demasiadas opciones.

¿Cómo es el código para Express?

En sitios web o aplicaciones web dinámicas, que accedan a bases de datos, el servidor espera a recibir peticiones HTTP del navegador (o cliente). Cuando se recibe una petición, la aplicación determina cuál es la acción adecuada correspondiente, de acuerdo a la estructura de la URL y a la información (opcional) indicada en la petición con los métodos POST o GET. Dependiendo de la acción a realizar, puede que se necesite leer o escribir en la base de datos, o realizar otras acciones necesarias para atender la petición

correctamente. La aplicación ha de responder al navegador, normalmente, creando una página HTML dinámicamente para él, en la que se muestre la información pedida, usualmente dentro de un elemento específico para este fin, en una plantilla HTML.

Express posee métodos para especificar qué función ha de ser llamada dependiendo del verbo HTTP usado en la petición (GET, POST, PUT, etc.) y la estructura de la URL ("ruta"). También tiene los métodos para especificar qué plantilla ("view") o gestor de visualización utilizar, donde están guardadas las plantillas de HTML que han de usarse y cómo generar la visualización adecuada para cada caso.

El middleware de Express, puede usarse también para añadir la gestión de cookies, sesiones y usuarios, mediante el uso de parámetros, en los métodos POST/GET.

Puede utilizarse además cualquier sistema de trabajo con bases de datos, que sea soportado por Node (Express no especifica ningún método preferido para trabajar con bases de datos).

Instalando el módulo Express

Instalación

Suponiendo que ya has instalado Node.js, crea un directorio para contener la aplicación y conviértelo en el directorio de trabajo.

En la terminal escribirnos:

```
mkdir myapp
cd myapp
```

o bien podemos realizar este paso creando una carpeta manualmente con el nombre del proyecto.

Luego utilizamos el comando npm init para crear un archivo package.json para la aplicación.

```
npm init
```

Este comando solicitará varios elementos, como el nombre y la versión de la aplicación. Por ahora, simplemente presiona ENTER para aceptar los valores predeterminados para la mayoría de ellos, excepto para el siguiente:

```
entry point: (index.js)
```

Especifica `app.js` o el nombre que desees para el archivo principal. Si deseas que sea `index.js`, simplemente presiona ENTER para aceptar el nombre de archivo predeterminado recomendado.

A continuación, instala Express en el directorio `myapp` y guárdalo en la lista de dependencias. De esta manera:

```
npm install express --save
```

Los módulos de Node que se instalan con la opción `--save` se añaden a la lista de dependencias en el archivo `package.json`. Posteriormente, si ejecutas `npm install` en el directorio `app`, los módulos se instalarán automáticamente en la lista de dependencias.

De esta manera logramos tener Express instalado listo para utilizar.

Hola Mundo! - en Express

Primero consideremos el tradicional ejemplo de ¡Hola Mundo!

```
var express = require('express');

var app = express();

app.get('/', function(req, res) {

  res.send('Hola Mundo!');

});

app.listen(3000, function() {

  console.log('Aplicación ejemplo, escuchando el puerto 3000!');

});
```

Las primeras dos líneas incluyen (mediante la orden require()) el módulo de Express y crean una aplicación de Express. Este elemento se denomina comúnmente app, y posee métodos para el enrutamiento de las peticiones HTTP, configuración del 'middleware', y visualización de las vistas de HTML, uso del motores de 'templates', y gestión de las configuraciones de las aplicaciones que controlan la aplicación (por ejemplo el entorno, las rutas, entre otras).

Las líneas que siguen en el código (las tres líneas que comienzan con app.get) muestran una definición de ruta que se llamará cuando se reciba una petición HTTP GET con una dirección ('/') relativa al directorio raíz. La función 'callback' toma una petición y una respuesta como argumentos, y ejecuta un send() en la respuesta, para enviar la cadena de caracteres: "Hola Mundo!".

El bloque final de código, define y crea el servidor, escuchando el puerto 3000 e imprime un comentario en la consola. Cuando se está ejecutando el servidor, es posible ir hasta la dirección localhost:3000 en un navegador, y ver como el servidor de este ejemplo devuelve el mensaje de respuesta.

Estructura de Archivos

Express no hace las cargas de estructura o de componentes utilizados.

Rutas, vistas, archivos estáticos, y otras lógicas de aplicación específica pueden vivir en cualquier número de archivos con cualquier estructura de directorio.

Mientras que esto es perfectamente posible, se puede tener toda la aplicación en un solo archivo. Es decir, dividir nuestro código en infinita cantidad de módulos según diferentes necesidades o desarrollar toda nuestra aplicación directamente en el entry point (lo cual no es lo más recomendable).

Esto es posible ya que normalmente en Express los desarrollos son basados en funciones sin embargo es común ver este tipo de aplicaciones organizadas mediante el patrón de arquitectura MVC.