Cómo configurar TypeScript con Express

Conoce la guía paso a paso para configurar TypeScript con un servidor Express. Este artículo cubre todo, desde la configuración inicial hasta el despliegue.

By Jeremy Holcombe

5 minutos de lectura

Ver original

TypeScript es un lenguaje de programación fuertemente tipado que amplía las capacidades de JavaScript. Ofrece una serie de características que te ayudarán a desarrollar aplicaciones escalables con Node.js y Express.

Una de las ventajas fundamentales de TypeScript sobre JavaScript es que proporciona clases de tipos, lo que facilita la escritura de código más predecible y mantenible. Además, TypeScript ofrece seguridad de tipos, lo que garantiza que tu código esté libre de errores en tiempo de ejecución y facilita la detección de fallos en las primeras fases del desarrollo. El lenguaje también incluye herramientas de refactorización y autocompletado, que mejoran la experiencia de los desarrolladores.

Además, Node.js y Express proporcionan un rendimiento excelente para aplicaciones de cualquier escala. El uso de clases en TypeScript también ayuda con la organización y la estructura, lo que contribuye aún más a la escalabilidad. Con estas herramientas, puedes crear aplicaciones robustas y escalables para hacer frente a la creciente demanda.

Este artículo muestra cómo configurar una aplicación Express utilizando TypeScript con un único endpoint. Después, explica cómo desplegar tu aplicación en el alojamiento de aplicaciones de Kinsta.

Cómo crear un servidor Express

Para seguir este tutorial, asegúrate de que tienes Node.js y npm instalados en tu ordenador. Para crear un servidor Express

1. Crea un directorio utilizando el siguiente código:

   ```
   mkdir sample_app && cd sample_app
   ```

2. Inicializa una aplicación Node.js en el directorio ejecutando este comando:

   ```
   npm init -y
   ```

   La bandera -y del comando acepta las indicaciones por defecto al crear un archivo package.json rellenado con el siguiente código:

   ```
   {
     "name": "sample_app",
     "version": "1.0.0",
     "description": "",
     "main": "index.js",
     "scripts": {
       "test": "echo "Error: no test specified" && exit 1"
     },
     "keywords": [],
     "author": "",
     "license": "ISC"
   }
   ```

3. A continuación, instala express para añadir funciones esenciales y dotenv para la gestión de variables de entorno en el directorio que acabas de crear ejecutando este comando:

   ```
   npm i express dotenv
   ```

4. Crea un archivo .env en la raíz del directorio sample_app y rellénalo con la variable que se indica a continuación.

   ```
   PORT=3000
   ```

5. Crea una aplicación express que responda con un texto Hello World cuando los usuarios visiten http://localhost:3000.

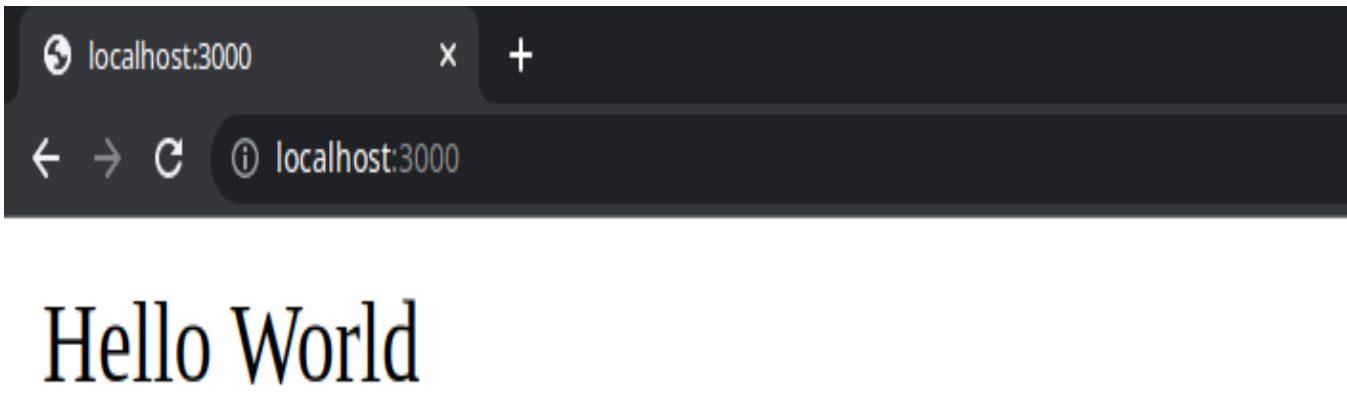
```
const express = require("express");
const dotenv = require("dotenv");

// configures dotenv to work in your application
dotenv.config();
const app = express();

const PORT = process.env.PORT;

app.get("/", (request, response) => {
  response.status(200).send("Hello World");
});

app.listen(PORT, () => {
  console.log("Server running at PORT: ", PORT);
}).on("error", (error) => {
  // gracefully handle error
  throw new Error(error.message);
})
```

dotenv.config() rellena el entorno de proceso de tu aplicación Node (process.env) con variables definidas en un archivo .env.

6. Inicia tu aplicación Node.js ejecutando este comando:

```
node index.js
```

Comprueba si la aplicación funciona visitando http://localhost:3000 en tu navegador. Deberías obtener una respuesta similar a ésta.



Hello World en http:localhost:3000.

Habilitar TypeScript en una aplicación Express

Sigue los pasos que se indican a continuación para utilizar TypeScript en una aplicación Express:

1. Instala TypeScript ejecutando este comando:

   ```
   npm i -D typescript
   ```

   La opción -D permite a npm instalar paquetes como dependencias dev. Puedes utilizar los paquetes que instales con esta opción en la fase de desarrollo.

2. Uno de los puntos fuertes de la comunidad TypeScript es el repositorio DefinitelyTyped de GitHub. Almacena documentación de definiciones de tipos para varios paquetes npm. Los usuarios pueden integrar rápidamente los paquetes npm en sus proyectos sin preocuparse de las dificultades relacionadas con los tipos,

instalando únicamente la definición de tipos para esos paquetes con npm. DefinitelyTyped es una herramienta indispensable para los desarrolladores de TypeScript. Les permite escribir código más limpio y eficiente y reducir la probabilidad de errores. Instala las definiciones de tipos tanto de express como de dotenv ejecutando este comando:

```
npm install -D @types/express @types/dotenv
```

3. Para inicializar TypeScript, ejecuta este comando.

   ```
   npx tsc --init
   ```

   El archivo tsconfig.json generado indica el directorio raíz de tu aplicación TypeScript. Proporciona opciones de configuración para definir cómo deben trabajar los compiladores de TypeScript. Incluye una serie de opciones de config deshabilitadas o habilitadas, con comentarios que explican cada opción.

4. Añade una propiedad outDir al objeto config para definir el directorio de salida.

   ```
   {
     "compilerOptions": {
       // …
       "outDir": "./dist"
       // …
     }
   }
   ```

Cómo crear un servidor TypeScript

Para crear un servidor TypeScript, cambia la extensión .js por .ts y actualiza el código con estas definiciones de tipo:

```
import express, { Request, Response } from "express";
import dotenv from "dotenv";

// configures dotenv to work in your application
dotenv.config();
const app = express();

const PORT = process.env.PORT;
```

```
app.get("/", (request: Request, response: Response) => {
  response.status(200).send("Hello World");
});

app.listen(PORT, () => {
  console.log("Server running at PORT: ", PORT);
}).on("error", (error) => {
  // gracefully handle error
  throw new Error(error.message);
});
```

Para utilizar el paquete compilador y compilar el archivo TypeScript en JavaScript, ejecuta el siguiente comando en el directorio raíz de tu aplicación.

```
npx tsc
```

A continuación, inicia tu aplicación ejecutando el comando

```
node dist/index.js
```

Al visitar http://localhost:3000 en tu navegador debería aparecer la respuesta «Hello World».

Cómo desplegar tu servidor TypeScript en Kinsta

Ahora, estás listo para desplegar tu aplicación en la web. Puedes desplegar tu aplicación en muchas plataformas, incluyendo el alojamiento de aplicaciones de Kinsta.

Antes de enviar tu aplicación a un repositorio Git, no es aconsejable utilizar TypeScript y enviar el archivo JavaScript compilado a Git. Incluye un script start en el archivo package.json.

```
{
  // …
  "script": {
    "start": "npx tsc && node dist/index.js",
  }
  // …
```

```
}
```

Además, crea un archivo .gitignore en el directorio raíz de tu aplicación e incluye node_modules y .env para evitar que estos archivos se envíen a tu proveedor de Git.

Una vez configurado tu repositorio, sigue estos pasos para desplegar tu aplicación en Kinsta:

1. Inicia sesión o crea una cuenta para ver tu panel MyKinsta.
2. Autoriza a Kinsta con tu proveedor Git.
3. Haz clic en Aplicaciones en la barra lateral izquierda, y luego en Añadir aplicación.
4. Selecciona el repositorio y la rama desde la que deseas desplegar.
5. Asigna un nombre único a tu aplicación y elige una Ubicación para el centro de datos.
6. Utiliza todas las configuraciones por defecto. MyKinsta utiliza npm start como punto de entrada para desplegar tu aplicación. Si quieres utilizar otro comando, puedes ajustar el proceso de ejecución en MyKinsta.
7. Haz clic en Crear aplicación.

Tras el despliegue, MyKinsta proporciona una URL para acceder públicamente al despliegue de tu aplicación. Puedes visitar la página para confirmar que muestra «Hello World»

Resumen

Esta guía ha demostrado cómo desarrollar y configurar una Aplicación Express utilizando TypeScript y desplegar la aplicación con Kinsta. TypeScript tiene capacidades adicionales que JavaScript no tiene — incluyendo clases de tipos, seguridad de tipos, herramientas de refactorización y autocompletado — para ayudarte a construir aplicaciones escalables y detectar errores durante el desarrollo.

Kinsta te ayuda a desplegar tu aplicación rápidamente con mayor seguridad y estabilidad. Con 21 centros de datos que ofrecen la máquina C2 de Google, que se ejecuta en la red de nivel premium de Google.

¿Has utilizado TypeScript en el pasado? ¿Qué opinas de utilizarlo con un servidor Express?

Jeremy Holcombe Kinsta

Editor de Contenidos y Marketing en Kinsta, Desarrollador Web de WordPress y Redactor de Contenidos. Aparte de todo lo relacionado con WordPress, me gusta la playa, el golf y el cine. También tengo problemas con la gente alta ;).