Setting up Express with TypeScript

Summary: in this tutorial, you will learn how to set up an Express application with TypeScript.

Prerequisites

- Node.js – version 20 or later to support `.env` file and the `--watch` flag. You don't need to install `dotenv` for reading .env file and `nodemon` package for automatically restarting the server whenever you change the source code.
- TypeScript compiler (ts-node) is a Node.js module that compiles TypeScript code to JavaScript.
- TypeScript executes (TSX) package that runs TypeScript directly within the Node.js environment.

If you don't have these installed, you can check how to set up the TypeScript development environment.

Setting up an Express application with TypeScript

Step 1. Create an Node.js project:

```
mkdir express-ts
cd express ts
npm init -y
```

This command creates a new project directory express-ts, navigates inside the directory, and creates a new package.json file.

Step 2. Install the `express` package:

```
npm install --save express@next
```

This command will install Express 5 or higher.

Step 3. Install development dependencies:

```
npm i -D @types/express @types/node
```

This command installs type definitions for `express` and Node.js as development dependencies:

- `@types/express` – type definitions for Express.
- `@type/node` – type definitions for Node.js.

The `-D` flag adds these packages as development dependencies. It'll add the following section to the `package.json` file:

```
  "devDependencies": {
    "@types/express": "^5.0.0",
    "@types/node": "^22.7.6"
  },
```

Note that you may have higher versions.

Step 4. Add an entry to the type module in the package.json file:

```
"type": "module",
```

This will allow you to use the ES module instead of the CommonJS module.

Additionally, change the scripts section to the following:

```
 "scripts": {
    "start": "npx tsx --env-file=.env --watch  src/index.ts"
  },
```

This command starts the TypeScript app defined in the `src/index.ts` , loads the environment variables defined in the `.env` file, and automatically restarts the server whenever you change the files:

- `npx tsx` uses the TSX runner for Node.js.
- `--env-file=.env` loads environment variables from a `.env` file.
- `--watch` watches for file changes and automatically restarts the server when detecting a file change.
- `src/index.ts` is the entry point file for the Express TypeScript application.

Step 5. Create an `.env` file in the project directory and set the port to `3000` :

```
PORT=3000
```

Step 6. Create a new directory `src` in the project directory and the `index.ts` file with the following code:

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

// Create a new express application instance
const app = express();

// Set the network port
const port = process.env.PORT || 3000;

// Define the root path with a greeting message
app.get("/", (req: Request, res: Response) => {
    res.json({ message: "Welcome to the Express + TypeScript Server!" });
});

// Start the Express server
app.listen(port, () => {
    console.log(`The server is running at http://localhost:${port}`);
});
```

Step 7. Run the npm start on your terminal:

```
npm start
```

It'll show the following output:

```
> express-ts@1.0.0 start
> npx tsx --env-file=.env --watch  src/index.ts

The server is running at http://localhost:3000
```

When you change the .ts code, the server will automatically restart.

This step requires VS Code and the Http Rest Client extension.

Step 8. Making an HTTP request to the root route by creating an `http` directory and `api.http` file within the `http` directory with the following content:

```
GET http://localhost:3000/
```

When you click the **Send Request** link, it'll make an HTTP request to `http://localhost:3000/` and return the following output:

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: application/json; charset=utf-8
Content-Length: 57
ETag: W/"39-6zzG7WgJ4cOX2hxA+l9Gi7ltUyc"
Date: ...
Connection: close

{
  "message": "Welcome to the Express + TypeScript Server!"
}
```