Journée 00 – Outillage Typescript & NodeJS

Programmation Applicative Web

Auteur·e : Élise C. Philippe

Édition : B2T3 2022-04 S

Avant-propos

Détails administratifs

Votre travail à rendre sur la forge logicielle. Le nom de votre dépôt doit être `ts_api0`. Les droits en lecture doivent être donnés à `delivery-collector`.

Vous devez être seul·e auteur·e de votre travail. Toutefois, vous êtes encouragé·e·s à échanger avec vos camarades. Utiliser le code d'un·e autre, c'est tricher. Et tricher annule toutes les médailles que vous avez reçues sur l'activité.

Propreté de votre rendu

Votre rendu ne doit comporter que les fichiers nécessaires à la construction de vos livrables. Sont interdits les fichiers suivants :

- les exécutables ;
- les repertoires `node_modules` et `dist` ;
- les fichiers compilés `.js` et `.map` ;
- les fichiers tampons `*.~` et `#*#`.

La présence d'un fichier interdit mettra immédiatement fin à la correction.

Présentation du projet

Lors de cet hyperespace vous allez travailler à la mise en place d'une API Web en utilisant les technologies suivantes:

- NodeJS ;
- Yarn ;
- Typescript ;
- expressjs ; et
- jest.

Votre objectif est de découvrir l'outillage nécessaire à cette *stack* logicielle, de découvrir les principes derrière une API Web REST et de le faire avec rigueur.

Vous allez découvrir comment :

- installer et configurer les différents paquets imposés (Typescript, express, jest) ;
- écrire des tests unitaires en NodeJS + Typescript ;
- designer une API Web ;
- implémenter les terminaisons d'une API Web (les *endpoints*) ;
- respecter un standard de l'industrie ;
- sécuriser une API Web et gérer ses permissions.

En bonus, vous serez encouragés à faire fonctionner les outils que vous utilisez avec votre IDE. Visual Studio Code est un IDE qui supporte Typescript et NodeJS avec peu de configuration.

00. Environnement de travail

Installation des programmes

Commencez par installer les programmes primordiaux à notre stack, qui sont :

- NodeJS ; et
- Yarn.

NodeJS est l'environnement nécessaire à l'exécution de JavaScript côté serveur. La pluparts des paquets NPM sont écrits pour fonctionner spécifiquement dans un environnement **NodeJS**. **Yarn** est une alternative à **NPM**, qui lui est le gestionnaire de paquets par défaut de **NodeJS**. L'intérêt de **Yarn** par rapport à **NPM** est qu'il est plus rapide et permet une résolution des dépendance qui est la même que vous installiez votre *stack* sur Windows, Linux ou MacOS, et que vous utilisiez un processeur ARM ou x86_64.

Notez qu'il y a quelques différences d'utilisation entre **Yarn** et **NPM**.

Installez au moins la version LTS de **NodeJS**, soit au moins la version 16.14.2. Installez ensuite **Yarn**. Si vous êtes sur Linux, installez-les depuis votre gestionnaire de paquet. Si vous êtes sur MacOS, vous devriez pouvoir les installer depuis Homebrew.

Pour tester votre installation de NodeJS, vous pouvez faire un hello world comme ceci :

```
cat > hello.js <<EOF
console.log("hello world");
EOF
node hello.js
```

Initialisation

Créez-vous un répertoire de travail, pouis initialisez votre projet avec `yarn init`. Appellez votre projet `e89_ts_api`, définissez l'entrypoint de votre projet comme étant `src/index.ts`, remplissez le champ auteur avec votre login 89, ne mettez pas de license et marquez votre projet comme privé. Si lors de l'initialisation un champ license a été ajouté tout de même, retirez-le complètement de votre `package.json`.

Initialisez votre dépôt `git` et poussez-le sur la forge.

Créez un `.gitignore` pour éviter de commit des fichiers et dossiers interdits.

Installation des paquets

Il faut que vous installiez avec yarn les paquets suivants :

- `typescript` , au moins la version 4.6 ;
- `ts-node` (seulement pour le développement).

Configurez typescript en ajoutant un fichier `tsconfig.json` et en lui indiquant de compiler tous les fichiers `.ts` contenus dans le dossier `./src` . Les fichiers compilés doivent être écrits dans un dossier `./dist` .

Configuration recommandée pour Typescript en NodeJS 16

Documentation de `tsconfig.json`

Pour tester votre configuration vous pouvez ensuite créer un fichier `src/index.ts` :

```
let str: string;
str = "hello world";
console.log(str);
```

Lancez `yarn tsc` , puis `node dist/index.js` .

`tsc` est le compilateur Typescript, sans argument, il cherche à compiler votre projets en regardant le contenu de `tsconfig.json` .

Écriture des “scripts”

Un fichier `package.json` peut contenir des petits scripts qui aident à l’utilisation de votre projet. Ajoutez les scripts suivants :

- `start` doit lancer le projet avec `ts-node src/index.ts` ;
- `build` doit construire le projet.

Référence d’utilisation des package.json

01. Votre première route

Pré-requis

Pour cette étape il vous faut installer le paquet `express`. Express est un outil de conception d'API Web pour NodeJS. C'est un module écrit en JS, qui ne propose pas de type exploitables par Typescript. Heureusement pour nous, il existe un projet, DefinitelyTyped, qui propose des définitions de types pour `express` notamment. Quand vous souhaitez avoir des définitions de types pour un paquet, qui n'en propose pas à l'origine, vous pouvez les installer via `yarn add --dev @types/${nom_du_paquet}`. **Installez `@types/express` pour bénéficier des définitions de types sur `express`.**

> Sur le site npm vous pouvez voir quels paquets proposent ou non des types en fonction de la présence d'une icône "DT" ou "TS" à côté de leur nom en haut de la page.

Exercice

Écrivez une route `GET /date` qui renvoie la date et l'heure au format ISO dans la propriété `date` d'un object JSON. Votre programme doit écouter sur le port 3089.

Comportement attendu:

```
$ yarn start &
$ http 127.0.0.1:3089/date
HTTP/1.1 200 OK
Connection: keep-alive
Content-Type: application/json; charset=utf-8
Date: Wed, 06 Apr 2022 12:49:50 GMT
X-Powered-By: Express

{
        "date": "2022-04-06T12:48:38.355Z"
}
```

> Dans l'exemple j'utilise l'outil httpie pour tester ma route. Vous pouvez utiliser l'outil de votre choix, que ce soit cURL, httpie, Postman ou Insomnia.