```c
int main() {
  printf("hello, world");
  return 0;
}
```

Typescript compilation et exécution ES2020

Farouk Touil

Jeudi le 9 juillet 2020

Salut, à la fin de ce document vous allez apprendre à mettre en place l'environnement de base pour les applications cotée serveur dit (Backend Applications) en anglais, aussi nommée API (Application Programming Interface) et automatiser la compilation de Typescript dans Visual Code Studio. Donc je vous invite a gardé précieusement ce document dans votre caisse a outil de développeur.

Etape N° 1 => Création du fichier package.json

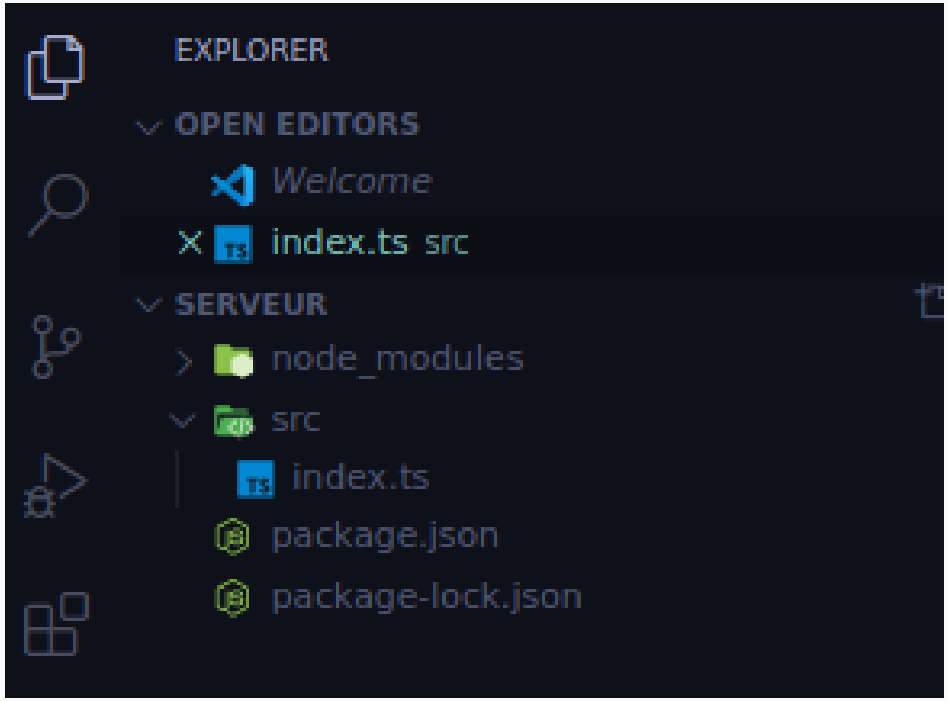
```
# Note: La taille totale du projet fini est de : 3,0 Mo
# Ctrl-Shift-p et choisir (Create New Intergrated Terminal)
# ouvrir le cmd est créer un dossier intitulé serveur
~$ mkdir serveur/
# naviguer dans le dossier
~$ cd serveur/
# exécuter la commande npm init --yes
~$ npm init --yes
# Remarque un fichier intitulé package.json est créer
# Voire Aperçu suivant:
```

Etape N° 1 => Aperçu du fichier package.json

```
0  {
1    "name": "serveur",
2    "version": "1.0.0",
3    "description": "",
4    "main": "index.js",
5    "scripts": {
6      "test": "echo \"Error: no test specified\" && exit 1"
7    },
8    "keywords": [],
9    "author": "",
10   "license": "ISC"
11 }
```

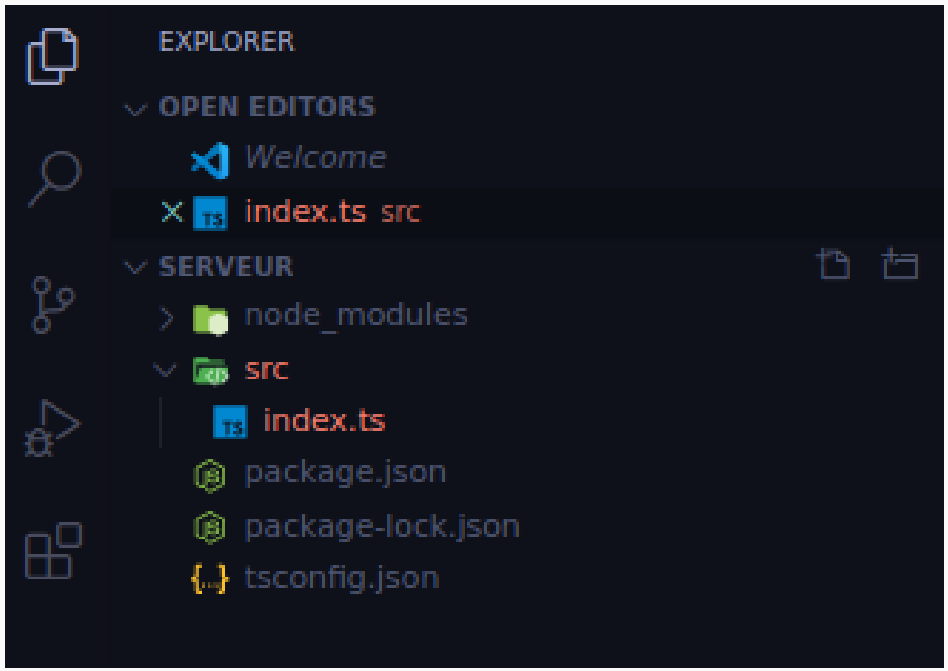
Etape N° 1 => Création de dossier (src)

```
# Créer un dossier intitulé (src)
~$ mkdir src/
# créer aussi un fichier (index.ts) dans le répertoire src/
~$ touch src/index.ts # écrire dans le fichier index.ts
~$ echo "console.log('IT WORKS!')" >> src/index.ts
# Voire Aperçu suivant :
```



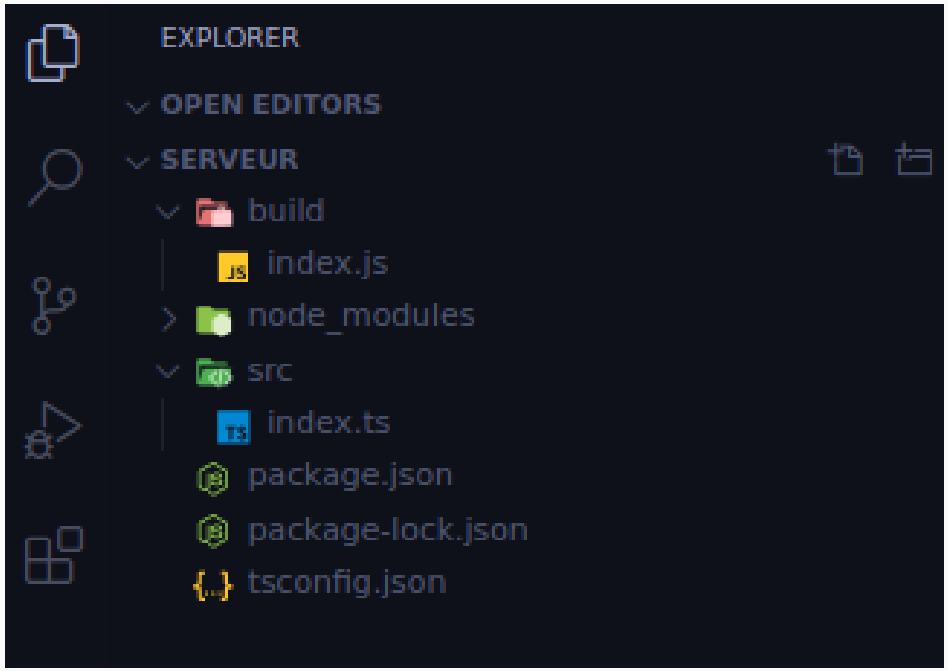
Etape N° 2 => Compilation de fichier typescript avec (tsc)

```
# Pour obtenir la commande (tsc) installer typescript
# ouvrir le cmd est tapé (npm install g typescript) "g" pour global
~$ npm install g typescript
# En suite en en exécute (tsc --init)
# Cette commande ajoute un fichier intitulé (tsconfig.json)
# Ouvrir ce dernier pour édition
# Chercher la ligne ("target": "es5") remplacé  es5 avec es2020
# Chercher la ligne (//"outDir": "./") remplacé ("outDir": "./build")
# Basculer vers le terminal
# Exécuter la commande (tsc)
```



Etape N° 2 => Résultat de l'execution de la commande (tsc)

Après exécution de la commande (tsc) le Typescript est compiler en JavaScript, remarquer le nouveau dossier (build) et fichier (index.js) créer. Dans la troisième étape nous automatiseront le processus de la compilation par l'ajout d'une ligne de code au fichier package.json.



Etape N° 3 => Automatiser la compilation avec watch mode

Dans cette troisième étape nous automatiseront le processus de la compilation par la modification d'une ligne de code au fichier package.json, à la ligne numéro 6.

Supprimer la ligne `("test":"echo \"Error: no test specified\" && exit 1")` et remplacer avec `("build":"tsc -w")`, prière de ne pas copier les parenthèses.

```
0   {
1     "name": "serveur",
2     "version": "1.0.0",
3     "description": "",
4     "main": "index.js",
5     "scripts": {
6       "test": "echo \"Error: no test specified\" && exit 1"
7     },
8     "keywords": [],
9     "author": "",
10    "license": "ISC"
11  }
```

Etape N° 3 => Résultats et testes des modifications

Maintenant chaque foi que le fichier (index.ts) est modifié, typescript écrit les modifications sur (index.js), donc la compilation est automatique.

```
0  {
1    "name": "serveur",
2    "version": "1.0.0",
3    "description": "",
4    "main": "index.js",
5    "scripts": {
6      "build": "tsc -w"
7    },
8    "keywords": [],
9    "author": "",
10   "license": "ISC"
11 }
```

Etape N° 3 => Exécution

Pour automatiser l'exécution des fichiers JS on installe un package appeler Nodemon avec la commande `$ npm install nodemon -D`

Puis ajouter le code `"dev":"nodemon build/index.js"` comme afficher dans le fichier package.json a la ligne 7:

```
0  {
1    "name": "serveur",
2    "version": "1.0.0",
3    "description": "",
4    "main": "index.js",
5    "scripts": {
6      "build": "tsc -w",
7      "dev": "nodemon build/index.js"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC"
12 }
```

Tester le tous

À la fin pour tester le tous, il va falloir ouvrir deux sessions dans le terminal, dans la première tapée `$ npm run build` , et dans la deuxième tapé `$ npm run dev`, puis juste ajouter du code dans index.ts est enregistré, le tous sera compilé est exécuté automatiquement, comme l'indique le titre du document.

Happy code.