Typescript 개발환경설정

Typescript 설정

```
npm init -y
npm install -D typescript ts-node @types/node
```

Typescript 사용하기

index.ts

```
const str:string = 'hello typescript'
console.log(str)
```

실행해보기

```
node index.ts

# **Error**
# SyntaxError: Missing initializer in const
```

node는 typescript 를 실행할수있는 기능이없다.

node는 `javascript runtime` 이기 때문.

그래서 javascript로 변환한 다음에 사용해야함.

```
npx tsc index.ts
```

이랬더니

```
┌ [프로젝트명]
├ index.js
└ index.ts
```

같은 디렉토리에 생성되었다.

내가 원하는 구조는.

```
┌ [프로젝트명]
├ /dist
├─ index.js
└ index.ts
```

dist 폴더 안에 넣고싶다.

그러면 `webpack` 의 `webpack.config.js` 파일처럼.

tsc 명령 실행을 하기전에 읽는 파일이 있습니다.

바로 `tsconfig.json` 입니다.

tsconfig.json

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

이렇게 작성후 명령어를 써봅시다.

```
$ npx tsc --build
```

그렇다면 원하는대로 실행이됩니다.

이후 `tsconfig.json`에는 다양항 설정이 가능합니다.

> Typescriptlang.org/tsconfig#allowJs

tsconfig.json

```
{
```

```
    "compilerOptions": {
        "outDir": "./dist/",
        "esModuleInterop": true,
        "moduleResolution": "node",
        "resolveJsonModule": true,
        "baseUrl": ".",
        "typeRoots": ["./node_modules/@types", "./@types"],
        "strict": true,
        "declaration": true,
        "paths": {
            "@core/*": ["core/*"],
            "*": ["@types/*"]
        }
    }
}
```

typescript 빌드하기

```
tsc index.ts
```

외부라이브러리 declara

```
$ npm install hex-to-bianry
$ npm install crpyto-js
```

- main.ts 참조

eslint / prettier 설정

eslint 는 문법적인 컨벤션 역활을함.

prettier 는 간단한 코드컨벤션을 맞추기 위해서 사용함

```
npm install -D eslint
npm install -D prettier eslint-plugin-prettier eslint-config-prettier
```

.eslintrc

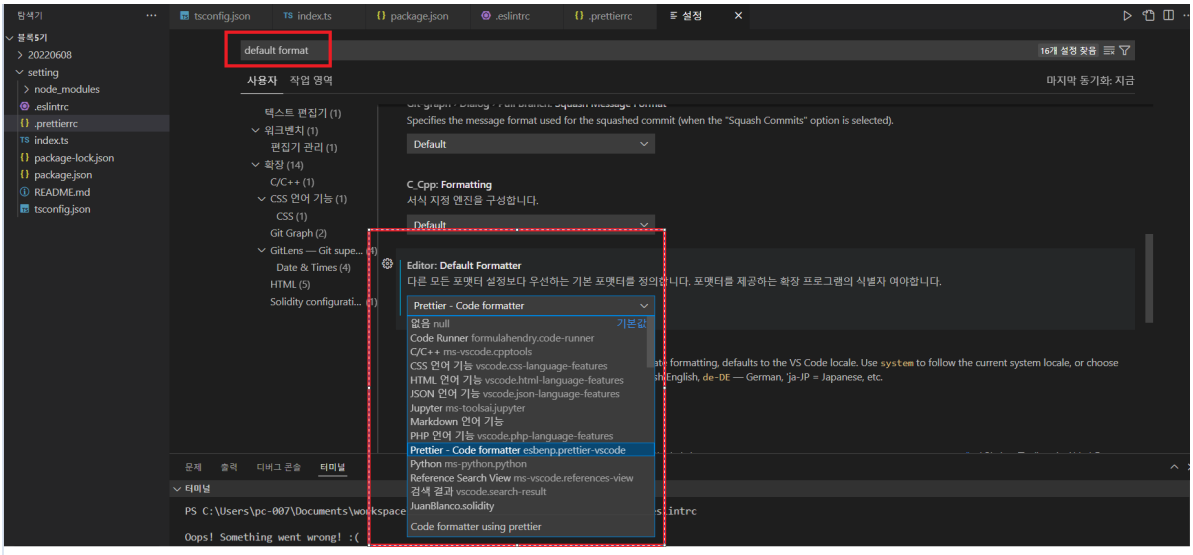
```
{
    "extends": ["plugin:prettier/recommended"]
}
```

.prettierrc

```
{
    "printWidth": 160,
    "tabWidth": 4,
    "singleQuote": true,
    "trailingComma": "all",
    "semi": false
}
```

이후 `vscode` 설정

Ctrl + , 누른뒤



Default Formatter 라고 검색한뒤,

Editor:Default Formatter 부분에 prettier - code formatter 내용으로 수정

Typescript 중요 키워드

instanceof

```
instanceof는 해당하는 변수가 사용하고 있는 prototype의 chain을 2번째 인자와 쭉 비교해서 true/false 값을 리턴한다.
```

type / interface

기본적으로 둘다같다.

개인적으로 Typescript 에서는 둘중 뭘선택해도 좋긴하나.

Generic 에 내용을 선언할시는 interface보단 조금 더 좋아보인다.

declare

npx ts-node --files [디렉토리]

혹은

```
npm install -D tsconfig-paths
```

ts-node -r tsconfig-paths/register

접근자

public

private

https://tuhbm.github.io/2019/02/27/accessModifier/