# TypeScript Configuration

# TypeScript

## TypeScript란?

### 1. 타입스크립트

TypeScript 는 Microsoft에서 개발 및 유지 관리 하며 오픈 소스 프로그래밍 언어이다.

TypeScript는 Microsoft에 의해 개발/관리되고 있는 오픈소스 프로그래밍 언어이다. 대규모 애플리케이션을 개발하는데 자바스크립트가 어렵고 불편하다는 불만에 대응하기 위해 개발되었다.

TypeScript는 스크립트 언어의 표준인 ECMA Script (줄여서 ES)의 표준을 따르기 때문에 JavaScript 영역을 침범하지 않고 최신 ES를 지원한다. 새로운 ES가 나올 때 마다, TypeScript 역시 버전 업을 하여 최신 ES의 기능을 지원한다.

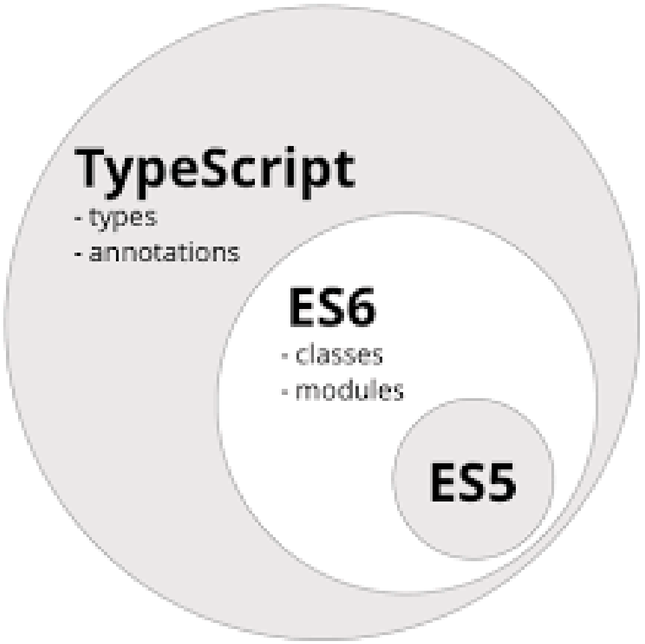
JavaScript 의 모든 구문과 의미를 지원 하는 동시에 정적 유형 지정 및 ECMAScript의 상위 집합 인 더 풍부한 구문과 같은 몇 가지 추가 기능을 제공한다.

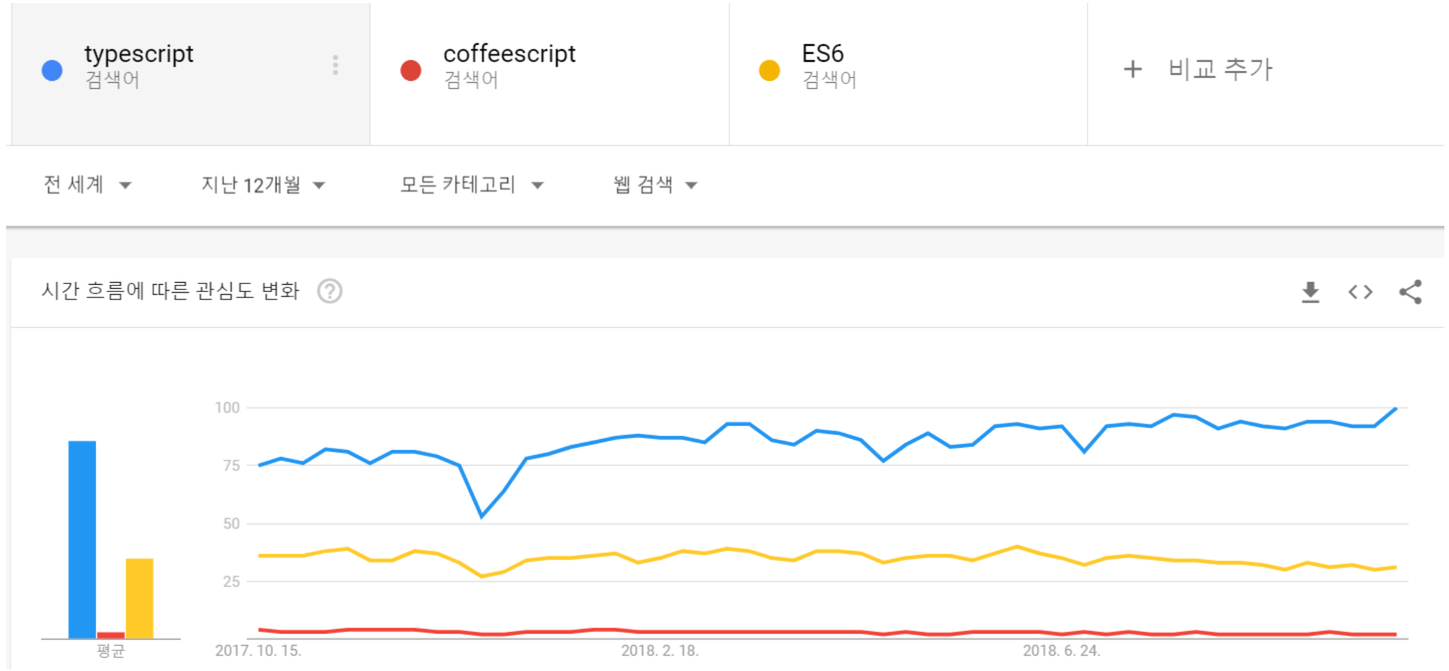
# TypeScript

## TypeScript란?

### 1. 타입스크립트

다음 다이어그램은 TypeScript와 ES5, ES2015, ES2016 간의 관계를 보여준다.



# TypeScript

## TypeScript 특징

### 2. 타입스크립트 특징

TypeScript의 가장 큰 특징은 type을 지정하는 것이다. 따라서 쉽게 보자면 이렇게 볼 수도 있다.

**TypeScript = JavaScript + Type**

자바스크립트의 var 와 같은 자료 형 대신, string, number와 같은 자료 형을 지정함으로써 안정성을 확보한다.
TypeScript를 설치할 때 같이 설치되는 tsc(TypeScript Compiler)는 컴파일 과정에서, 타입 검사를 통해 에러 없이 안정성이 확보되면 타입들을 제거하고 최종적으로 자바스크립트 코드를 생성한다.

자바스크립트에 없는 type이 지원되면서 몇 가지 장점이 생기는데, 우선 변수에 type이 추가되어 안정성이 확보된다.
두 번째로 type에 대한 예외 처리를 하지 않아도 된다

```
if (typeof name !== "string") {
        // Type Guard 처리
}
```

# TypeScript

## TypeScript 특징

### 2. 타입스크립트 특징

마지막 장점은 JavaScript 엔진의 최적화를 돕는다.

```
var onAdd = function(x, y) {
    return x + y;
}
```

JavaScript 엔진은 위 함수가 자주 호출되며, 인자 x, y가 정 수형으로 들어오는 것으로 생각한다. 하지만 이는 100% 보장할 수 없으며 어느 순간 정수가 아닌 string이 들어올 경우, 이 최적화 된 코드는 string 처리를 할 수 없어 최적화가 해제된다.

최적화와 최적화 해제가 지속적으로 발생하면 오히려 성능이 더 저하된다. typescript에서는 type을 지정하고 컴파일 과정에서 이를 검사하기 때문에 최적화 해제가 발생할 상황이 더 적어진다.

또한 ES6이 지원하지 않는 namespace를 도입하여 전역 이름공간(global namespace)이 더러워지는 것을 방지한다. 또한 ES6의 클래스 특징을 따르며, 여기에 ES6에 없는 인터페이스 특징을 지원함으로 더 나은 OOP 환경을 제공한다.

# TypeScript

## TypeScript 개발환경

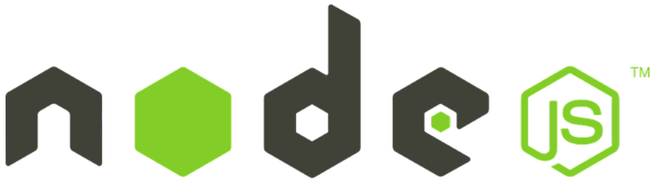
### 3. 개발 환경

1. NodeJS 설치.

   https://nodejs.org/ko/

   설치 확인 - Command로 확인

   ```
   node -v
   npm -v
   ```



2. VSCode 설치

   https://code.visualstudio.com/

# TypeScript

## TypeScript 개발환경

### 3. 개발 환경

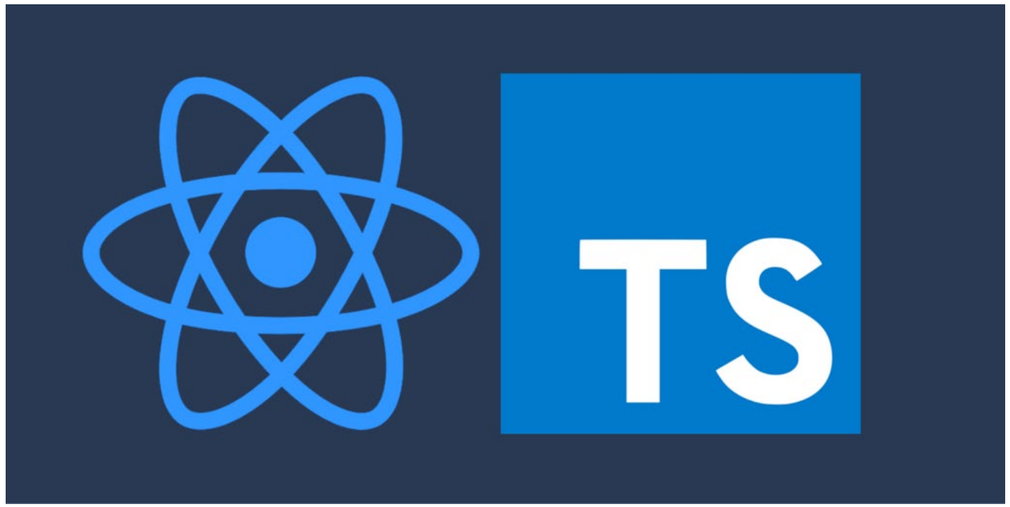
3. TypeScript 설치.

command로 설치

```
npm install -g typescript
```

설치 확인

```
tsc -version
```



# TypeScript
## VSCode Setting

## 4. VSCode Setting

1. 확장 프로그램 설치. [ 보기 - 확장 ]

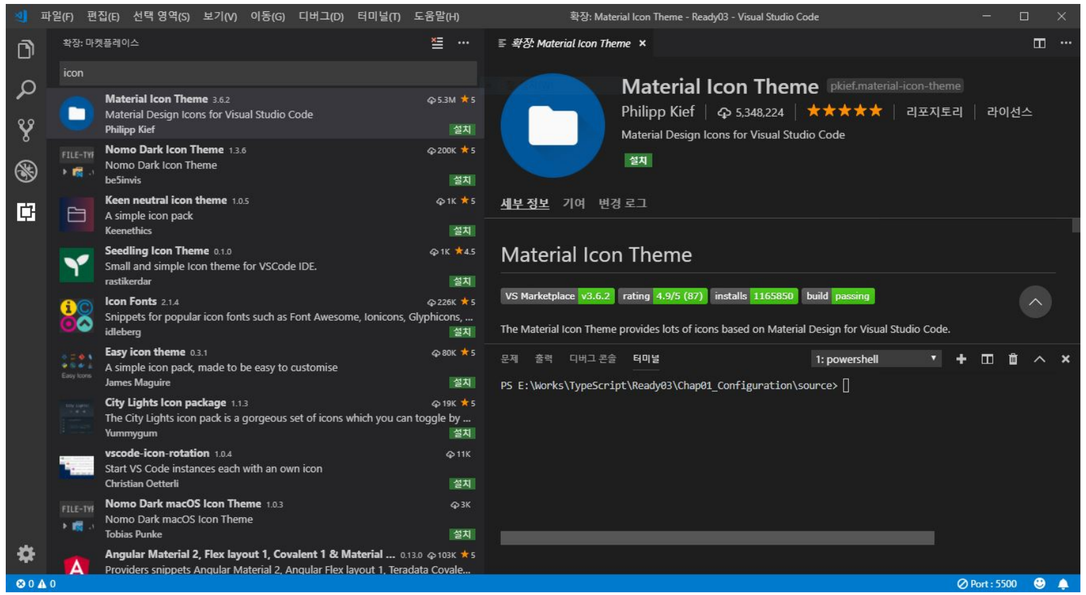
검색 키워드를 이용하여 검색 후 설치한다

**Korean Language Pack for Visual Studio Code**

**Live Server**

**JavaScript (ES6) code snippets**

**Material Icon Theme**




**angular2-inline**

**Angular 7 Snippets - TypeScript, Html, Angular Material, ngRx, RxJS & Flex Layout**

# TypeScript
## VSCode Setting

### 4. VSCode Setting

2. 기본 설정. [ 파일 - 기본 설정 - 설정 ]
    상단의 검색을 이용하여 빠르게 설정할 수 있다.

3. 단축키 설정. [ 파일 -기본 설정 - 바로 가기 키 ]
    변경하고자 하는 항목을 더블클릭 후 변경.

| | |
|---|---|
| Shift + Alt +F 또는 Ctrl + K Ctrl + F | 코드 정리 |
| Shift + Alt + A | 코드 블록 주석 처리 |
| Ctrl + / | 코드 한 줄 주석 처리 |

4. 테마 변경 [ 파일 - 기본 설정 - 색 테마 ]
    확장 프로그램을 이용하여 다양한 테마를 사용 가능 - theme로 검색 후 설치

5. 아이콘 변경 [ 파일 - 기본 설정 - 파일 아이콘 테마 ]

# TypeScript

## TypeScript Compile

### 5. TypeScript Compile

1. JS 파일로 변경.
    tsc fileName
        tsc sample.ts

2. 위치 지정
    tsc fileName --outDir 경로
        tsc sample.ts --outDir ./dist

3. JS 형식 지정
    tsc fileName --target 형식
        tsc sample.ts --target es5

4. JS Module 지정
    tsc fileName --module 형식
        tsc sample.ts --module commonjs

```
namespace sample {
    class Jumsu {
        constructor(private name: string, private kor: number, private
        eng: number){}

        public getTotal(): number {
            return this.kor + this.eng;
        }

        private getAvg(): number {
            return this.getTotal() / 2;
        }

        public display(): void {
            console.log(`${this.name}님의 총점은 ${this.getTotal()} /
            ${this.getAvg()} 입니다`);
        }
    }

    let hong: Jumsu = new Jumsu("홍길동", 100, 80);
    hong.display();
}
```

# TypeScript
## VSCode Compile

### 5. TypeScript Compile

5. tsconfig.json 파일 작성

tsc -init

파일 수정 - 다음 항목 주석을 해제한다

```
"sourceMap": true,
"outDir": "./dist",
```

[참조]

https://www.typescriptlang.org/docs/handbook/tsconfig-json.html

https://www.typescriptlang.org/docs/handbook/compiler-options.html

[ 한글 참조 사이트 ]
https://vomvoru.github.io/blog/tsconfig-compiler-options-kr/

6. tsconfig.json 설정에 의한 Compile

tsc

tsc -p ./

# TypeScript

## VSCode Compile

### 6. tasks.json 파일을 이용한 build

1. tasks.json 파일 작성

메뉴의 "보기 - 명령 팔레트"

> 기본 빌드 작업 구성 (입력)

tsc: 감시 tsconfig.json (선택)

2. 감시 모드 실행

메뉴의 "터미널 - 빌드 작업 실행"

tasks.json 파일

```
{
	"version": "2.0.0",
	"tasks": [
		{
			"type": "typescript",
			"tsconfig": "tsconfig.json",
			"option": "watch",
			"problemMatcher": [
				"$tsc-watch"
			],
			"group": {
				"kind": "build",
				"isDefault": true
			}
		}
	]
}
```