

ITEM 60 EFFECTIVE TS

ITEM 60. allowJS로 TS와 JS 같이 사용하기



```
tsconfig.json

{
  allowJs: true
}
```

소규모 프로젝트는 한번에 TS 프로젝트로 변환할 수 있으나, 대규모 프로젝트는 점진적으로 전환할 수 밖에 없습니다. 이 때 allowJs 컴파일러 옵션을 설정하면, TS파일과 JS파일을 서로 임포트 할 수 있도록 해줍니다.

타입체크랑 상관은 없지만, 기존 빌드과정에 TS 컴파일러를 추가하기 위해서라도 allowJs 옵션을 설정해야 합니다.



browerify 는 모듈 번들러 라이브러리로, webpack과 유사합니다

```
swarv > npm install --save-dev tsify_
```

in pwsh at 14:13:17

```
swarv > browserify index.ts -p [ tsify --allowJs ] > bundle.js_
```

in pwsh at 14:13:17

번들러에 타입스크립트가 통합되어 있거나, 플러그인 방식으로 통합이 가능하다면 allowJs 를 쉽게 적용할 수 있습니다.

```
module.exports = {  
  transform: {  
    '^.+\\.tsx?$': 'ts-jest',  
  },  
};
```

대부분의 테스트 도구에는 동일한 역할의 옵션이 있습니다.
예를 들어, jest를 사용할 때 ts-jest를 설치하고 jest.config.js
에 전달할 타입스크립트 소스를 지정하는 것이 있습니다.

```
// "outDir": "./dist",  
/* 출력될 디렉토리 설정 */
```

프레임워크 없이 빌드 체인을 직접 구성했다면 복잡한 작업이 필요할 것입니다. 한 가지 방법으로는 outDir 옵션입니다.

outDir 옵션을 사용하면 TS가 outDir에 지정된 디렉토리에 소스 디렉토리와 비슷한 구조로 JS 코드를 생성하므로, outDir로 지정된 디렉토리를 대상으로 빌드체인을 진행하면 됩니다

allowJs 유의사항

기존 JS 코드에 특별한 규칙이 있었다면, TS가 생성한 코드가 JS의 규칙을 따르도록 출력 옵션을 조정해야할 수도 있습니다.

TS로 마이그레이션하며 빌드와 테스트가 동작하게 하는 것은 어렵지만, 제대로 된 점진적 마이그레이션을 위해 필요합니다



THANK YOU

끝까지 봐주셔서 감사합니다 :)