Ionic Study

Day 7

오늘 할 것들

• 타입스크립트 알아보기

- Typescript
 - Ionic3은 Angular와 타입스크립트에 기반을 두고 있음.
 - 타입스크립트는 컴파일러에 의해 자바스크립트로 변환되어 실제 동작.

- Typescript 컴파일러 설치
 - npm install –g typescript

```
- - X
MINGW64:/
Ayami@Ayami-PC MINGW64 /
$ npm install -g typescript
C:\Users\Ayami\AppData\Roaming\npm\tsserver -> C:\Users\Ayami\AppData\Roaming\np
m\node_modules\typescript\bin\tsserver
C:\Users\Ayami\AppData\Roaming\npm\tsc -> C:\Users\Ayami\AppData\Roaming\npm\nod
e_modules\typescript\bin\tsc
+ typescript@2.9.2
updated 1 package in 1.985s
$ tsc --help
Version 2.9.2
Syntax: tsc [options] [file ...]
Examples: tsc hello.ts
              tsc --outFile file.js file.ts
              tsc @args.txt
Options:
 -h, --help
                                                                         Print this message.
                                                                         Show all compiler options.
      --version
                                                                         Print the compiler's version
```

- tsc 명령
 - 타입스크립트 컴파일러 호출
 - ts 확장자를 가진 타입스크립트 파일을 입력으로 주면 출력으로 동일 이름의 js 확장자를 가진 자바스크립트 파일이 생성됨.
 - node로 js파일 실행

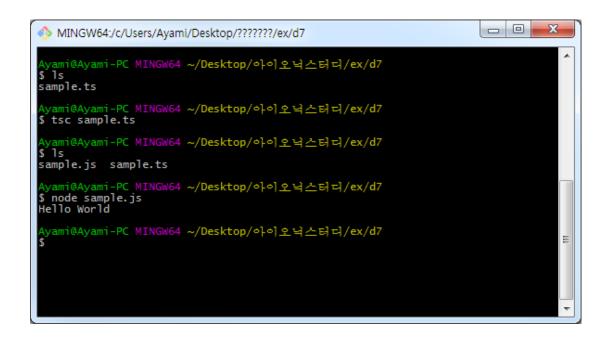
sample.ts

var message:string="Hello World";
console.log(message)

컴파일

sample.js

var message="Hello World";
console.log(message)



- 변수의 자료형
 - 타입스크립트에서는 변수에 변수의 형태(자료형)를 지정하고 동일 형태로만 사용 가능.
 - 자바스크립트는 변수의 형태를 지정할 필요가 없음 (하나의 변수를 여러 타입으로 사용 가능)

• ex1.ts

```
_ 0 X
ex1.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D) 작업(T) 도움말(H)
                                                                                               ♦ Ⅲ …
                                                TS ex1.ts
                                                           ×
         탐색기
                                                       var str='string';
       ▲ 열려 있는 편집기
                                                       console.log('str: '+ str);
          TS ex1.ts d7
       ⊿ EX
                                                       var num=1.1;
        ▶ 📹 d4
         ⊿ 📹 d6
                                                       str = num;
            JS ex1.js
 ⑻
                                                       console.log('num: '+ num);
            JS ex2.js
            JS ex3.js
 ¢
            JS ex4.js
         ⊿ 📹 d7
            TS ex1.ts
            JS sample.js
           TS sample.ts

■ d4.zip

 ⊗1 ∆ 0
                                                 줄 7, 열 27 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 🛕 TSLint
```

- 타입스크립트에서 변수의 선언
 - 변수 선언부에 사용할 형태를 알려준다
 - ex) var variable:string='string'
 - variable 변수는 string형을 가짐
 - ex) var num:number=1.1
 - num 변수는 number형을 가짐

• 타입스크립트에서 지원하는 기본 자료형

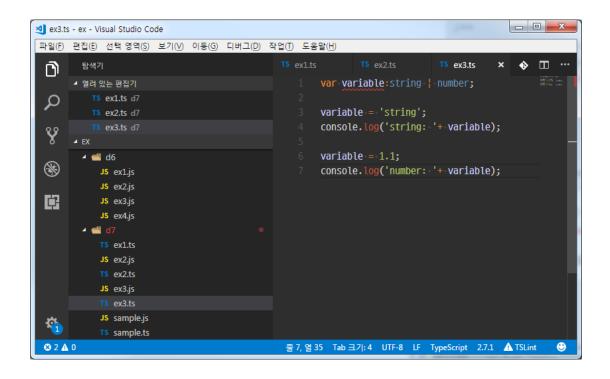
타입	설명
number	64비트의 크기를 갖는 실수(정수+소수를 포함한)형 숫자
string	유니코드 문자열
boolean	true와 false의 값을 가지는 논리형
array <type></type>	Type형태를 갖는 집합
union	하나의 변수가 여러 형태의 자료형을 가질 경우 사용
tuple	여러 형태의 array
any	정의되지 않은 임의의 타입

- 초기화되지 않은 변수는 undefined 값이 할당됨.

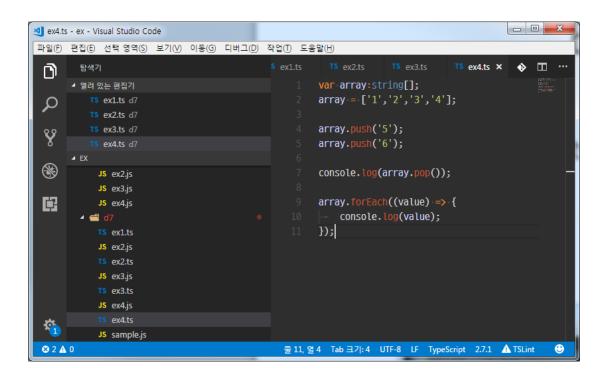
- any 자료형 (ex2.ts)
 - 여러 타입으로 변수를 사용 가능.

```
- - X
al ex2.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D) 작업(T) 도움말(H)
                                                              TS ex2.ts
                                                       var variable:any;
       ▲ 열려 있는 편집기
                                                       console.log('variable: '+ variable);
          TS ex1.ts d7
           TS ex2.ts d7
                                                       var str:string = 'string';
                                                       variable = str;
                                                       console.log('string: '+ variable);
            JS ex1.js
                                                  8 var num:number = 1.1;
            JS ex2.js
                                                       variable = num;
            JS ex3.js
            JS ex4.js
                                                 10 console.log('number: '+ variable);
         TS ex1.ts
            TS ex2.ts
            JS sample.js
           TS sample.ts
           d4.zip
 줄 10, 열 35 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 ▲ TSLint
```

- union 자료형 (ex3.ts)
 - 사용할 변수가 가지게 될 자료형을 미리 알고 있다면 any 대신 사용 가능. union은 각 형태의 or (|) 로 표기.



- array 자료형 (ex4.ts)
 - 동일한 자료형의 집합은 array를 사용
 - 자바스크립트의 array 객체와 동일



배열 리터럴을 이용한 방식 var array: string[];

제네릭을 이용한 방식 var array: Array<string>

- tuple 자료형 (ex5.ts)
 - 자료형이 다른 여러 개의 변수는 튜플 타입으로 정의.
 - array 객체의 일부 메소드를 지원, 선언한 만큼 요소를 할당해야 함

```
ex5.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D) 작업(T) 도움말(H)
                                                        var tuple:[string, number, string, string];
       ◢ 열려 있는 편집기
                                                        tuple=['one',2,'three','four'];
           TS ex1.ts d7
           TS ex2.ts d7
                                                        tuple.push(5);
           TS ex3.ts d7
                                                         tuple.push(6);
           TS ex5.ts d7
                                                        console.log(tuple.pop());
         ⊿ 📹 d7
                                                        tuple.forEach((value) => {
                                                        if (typeof value === 'string') {
            JS ex2.js
                                                                 console.log('\''+ value + '\'');
            TS ex2.ts
                                                          } else {
            JS ex3.js
                                                                 console.log(value);
            TS ex3.ts
            JS ex4.js
                                                   15 });
            TS ex4.ts
            JS ex5.js
            JS sample.js

② 2 ▲ 0

                                                            줄 15, 열 4 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 ▲ TSLint
```

- 사용자 정의 형태
 - 타입스크립트에서는 기본적으로 제공되는 자료형 외에 사용자가 원하는 형태를 클래스로 정의하여 사용하는 것이 가능.
 - 클래스는 생성자, 멤버 변수, 멤버 함수 세가지로 정의.
 - 생성자 : 클래스가 객체를 생성할 때 수행되는 함수
 - -> 객체를 초기화하기 위한 작업을 수행

• 클래스 (ex6.ts)

```
- - X
ex6.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D)
                                             작업(T) 도움말(H)
         탐색기
                                                                                              TS ex6.ts X
                                                                                                          ♦ Ⅲ …
                                                        class Animal {
       ▲ 열려 있는 편집기
                                                            public name:string;
          TS ex1.ts d7
                                                                                                   멤버 변수
                                                            private isAlive:boolean;
          TS ex2.ts d7
          TS ex3.ts d7
                                                             constructor(name:string) {
          TS ex4.ts d7
          TS ex5.ts d7
                                                                 this.name = name;
                                                                                                     생성자
 (%)
                                                                 this.isAlive = true;
          TS ex6.ts d7

■ EX
 ₽
            JS ex4.js
                                                             dead() {
         🚄 📹 d7
                                                                 this.isAlive = false;
            TS ex1.ts
            JS ex2.js
                                                                                                   멤버 함수
            TS ex2.ts
                                                            status() {
            JS ex3.js
                                                                 return this.isAlive
            TS ex3.ts
            JS ex4.js
            TS ex4.ts
            JS ex5.js
                                                        let cat = new Animal('야옹이');
            TS ex5.ts
                                                        cat.dead();
            JS ex6.js
                                                        console.log(cat.status());
            TS ex6.ts
            JS sample.js
                                                          줄 20, 열 12 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 🛕 TSLint

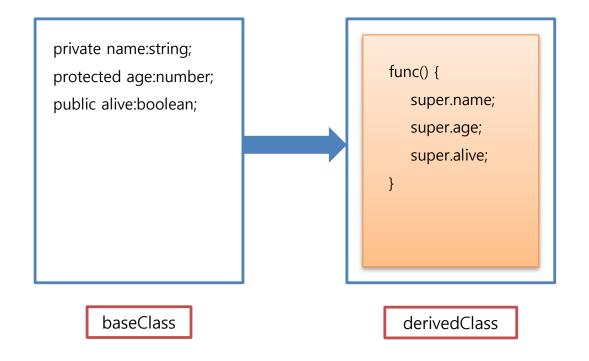
② 2 ▲ 0
```

• 클래스

- 접근 권한 : 멤버 변수 또는 멤버 함수에 접근할 수 있는 범위를 설정.
 - private : 동일한 클래스 내부에서만 접근 가능
 - protected : 동일한 클래스와 해당 클래스를 상속받은 클래스에서만 접 근 가능
 - public : 동일 클래스 및 외부에서 접근 가능
- this 키워드: 클래스 내부의 필드에 접근하거나 함수를 호출 할 때 사용.

- 클래스의 상속
 - 상속: 원래 클래스의 멤버 변수와 멤버 함수를 새로운 클래스에서도 추가 선언 없이 사용하는 방법
 - 부모 클래스, 자식 클래스
 - 부모 클래스 : 특성을 제공할 클래스 (기본 클래스)
 - 자식 클래스 : 특성을 상속받을 클래스 (파생 클래스)
 - extends 키워드를 이용하여 클래스 상속 실현
 - 자식 클래스 내부에서는 super 키워드를 이용하여 부모 클래스의 멤버 함수를 호출할 수 있음.
 - protected, public 스코프만 가능.

• 클래스



컴파일 가능 여부와 불가능일 경우 왜 그런건지?

- 클래스의 모듈화
 - 모듈화
 - 만들어진 클래스를 다른 파일에서 사용하려면?
 - 1. 그 파일에서 또 클래스를 선언한다
 - 2. 만들어진 클래스를 어떻게든 불러와 재활용한다.
 - 같은걸 계속 만들어 내는것은 생산성 저하, 유지보수 최악의 특징. (원래 내용이 바뀌면 복사한 것의 내용도 바꿔야 하니깐)
 - 재활용을 위해 존재하는 키워드가 export 키워드임.
 - 내보낼 때는 클래스 선언부 앞에 export를 붙여주면 끝.

• ex7.ts (ex6의 export 사용)

```
ex7.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D) 작업(T) 도움말(H)
 G
                                                                                TS ex6.ts ● TS ex7.ts X
        탐색기
                                                   1 export class Animal {
       ▲ 열려 있는 편집기 1이(가) 저장되지 않음
                                                   public name:string;
          TS ex1.ts d7
                                                        private isAlive:boolean;
          TS ex2.ts d7
          TS ex3.ts d7
                                                       constructor(name:string) {
          TS ex4.ts d7
                                                                this.name = name;
         TS ex5.ts d7
 ⑻
                                                                this.isAlive = true;

■ TS ex6.ts d7

          TS ex7.ts d7
 ¢

■ EX

                                                           dead() {
            JS ex2.js
                                                               this.isAlive = false;
            TS ex2.ts
            JS ex3.js
            TS ex3.ts
                                                            status() {
            JS ex4.js
                                                        return this.isAlive
            TS ex4.ts
            JS ex5.js
            TS ex5.ts
                                                           줄 17, 열 2 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 🛕 TSLint

② 2 ▲ 0
```

- export로 내보낸 모듈 불러오기
 - import 키워드를 사용하여 export로 내보낸 모듈을 불러올 수 있음.
 - import {타입명} from '파일 경로';
 - 이렇게 불러온 모듈은 사용자 정의형으로 자유롭게 사용이 가능.

• ex8.ts (바퀴벌레의 죽음)

```
- - X
ex8.ts - ex - Visual Studio Code
파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 디버그(D) 작업(T) 도움말(H)
                                                                                                          ♦ Ⅲ …
 G
                                                        TS ex5.ts
                                                                    TS ex6.ts ● TS ex7.ts
                                                                                              TS ex8.ts X
        탐색기
                                               x4.ts
                                                    import {Animal} from './ex7';
                                                                                                             ME.
       ▲ 열려 있는 편집기 1이(가) 저장되지 않음
          TS ex1.ts d7
                                                    3 var home = new Array<Animal>();
          TS ex2.ts d7
                                                    4 home.push(new Animal('Human'));
          TS ex3.ts d7
                                                    5 home.push(new-Animal('Dog'));
          TS ex4.ts d7
                                                        home.push(new Animal('Cockroach'));
          TS ex5.ts d7
 ⑻

■ TS ex6.ts d7

                                                   8 console.log(home[2].status());
          TS ex7.ts d7
 ¢
                                                   9 home[2].dead()
          TS ex8.ts d7
                                                        console.log(home[2].status());
       ■ EX
            TS ex6.ts
            JS ex7.js
            TS ex7.ts
            JS ex8.js
            TS ex8.ts
            JS sample.js
            TS sample.ts
           🚹 d4.zip
                                                           줄 11, 열 1 Tab 크기: 4 UTF-8 LF TypeScript 2.7.1 🛕 TSLint

② 2 ▲ 0
```

오늘은 여기까지~

See you next day!