

## ITEM 33.

**string** 타입보다  
더 구체적인 타입 사용하기

'23. 06. 16. (FRI) 신현호 (@SWARUY)

눈빛 애교 어피치

잘생기면

23:26

잘생기면

23:26

눈빛 애교 어피치

좋은점이야 많지

23:26

구체적으로?

23:27

눈빛 애교 어피치

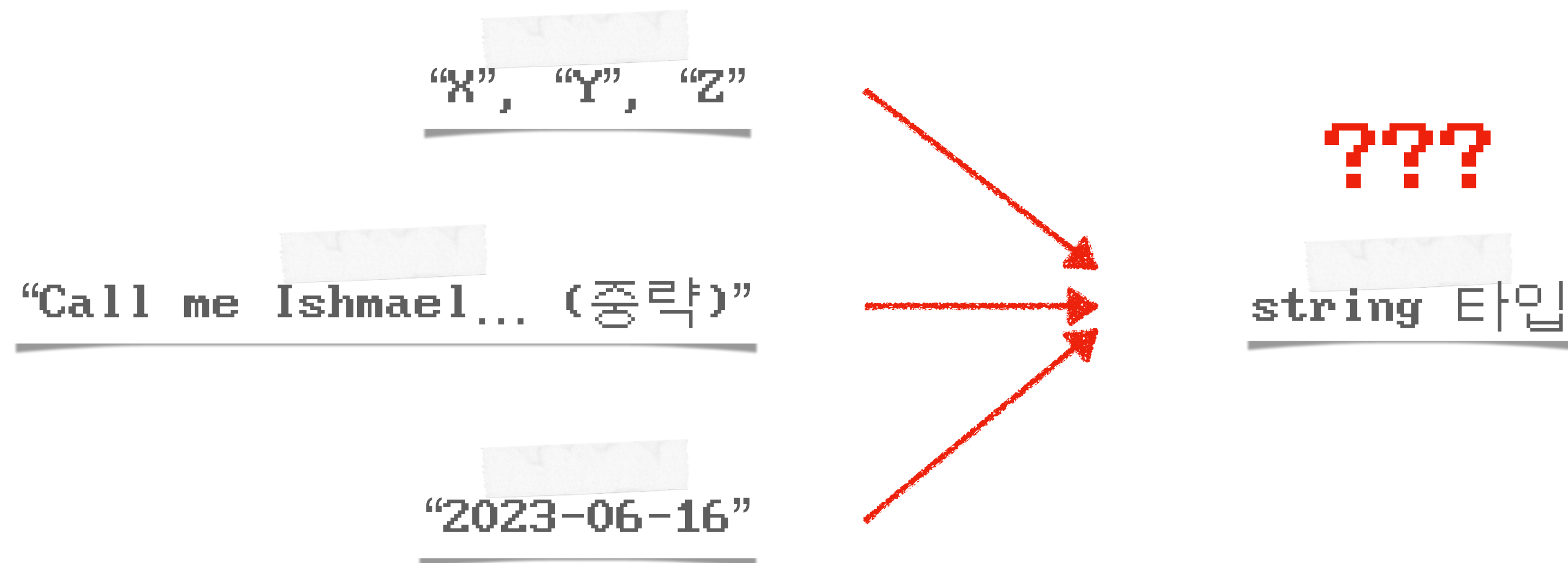
못생긴 놈은 몰라도 돼

23:27

좀유도 문등 톨타곤 레

23:51

string 타입은요...




`string` 타입의 범위는 매우 넓습니다, 셋 모두 `string` 타입을 가집니다.  
`string`으로 표현되는 범위가 너무 넓어서 `string` 타입으로 변수를 선언하려한다면  
혹시 그보다 더 좁은 타입이 적절하지 않을까를 생각해봐야합니다

```
interface Album {  
    artist: string;  
    title: string;  
    releaseDate: string;    // YYYY-MM-DD  
    recordingType: string;  // 예를 들어, "live" 또는 "studio"  
}
```

---

**string** 타입이 남발된 모습입니다. 게다가 주석에 타입 정보까지 있는걸 보아하니 현재 인터페이스조차 잘못되어있네요.



```
const kindOfBlue: Album = {  
  artist: 'Miles Davis',  
  title: 'Kind of Blue',  
  releaseDate: 'August 17th, 1959', // 날짜 형식이 다릅니다.  
  recordingType: 'Studio',           // 오타(대문자 S)  
};                                   // 정상
```

---

이런식으로 잘못된 값을 넣어도 정상동작하겠네요

```
interface Album {  
  artist: string;  
  title: string;  
  releaseDate: string; // YYYY-MM-DD  
  recordingType: string; // 예를 들어, "live" 또는 "studio"  
}
```

```
const kindOfBlue: Album = {  
  artist: 'Miles Davis',  
  title: 'Kind of Blue',  
  releaseDate: 'August 17th, 1959', // 날짜 형식이 다릅니다.  
  recordingType: 'Studio',           // 오타(대문자 S)  
};                                    // 정상
```

**releaseDate** 의 날짜형식도 다르고, **recordingType** 또한 잘못된 값으로 들어가있습니다  
그러나 타입 자체는 문제가 없기때문에 해당 객체는 **Album**에 할당 가능하며 타입체커를 통과합니다

```
function recordRelease(title: string, date: string) { /* ... */ }  
recordRelease(kindOfBlue.releaseDate, kindOfBlue.title); // 오류여야 하지만 정상
```

---

타입이 모두 `string`으로 동일하기때문에, 매개변수의 순서가 바뀌었지만  
타입체커가 오류를 잡아내지 못하고있습니다.

```
type RecordingType = 'studio' | 'live';
```

```
interface Album {  
  artist: string;  
  title: string;  
  releaseDate: Date;  
  recordingType: RecordingType;  
}
```



```
const kindOfBlue: Album = {  
  artist: 'Miles Davis',  
  title: 'Kind of Blue',  
  releaseDate: new Date('1959-08-17'),  
  recordingType: 'Studio'  
  // ~~~~~ "Studio" 형식은 'RecordingType' 형식에 할당할 수 없습니다.  
};
```

하지만 이런식으로 타입을 좁히게 된다면,  
타입스크립트는 오류를 더 세밀하게 체크할 수 있습니다

## 타입을 좁히면...

1.

타입을 명시적으로 정의함으로써  
다른 곳으로 타입이 전달되어도 타입 정보가 유지됩니다.

2.

타입을 명시적으로 정의하고  
해당 타입의 의미를 설명하는 주석을 붙여 넣을 수 있습니다.

3.

`keyof` 연산자로 더욱 세밀하게 객체의 속성 체크가 가능해집니다.



`string`도 사실...

`string`은 `any`와 비슷한 문제를 가지고 있습니다.

따라서 잘못 사용하게 되면 무효한 값을 허용하게 되고,  
타입간의 관계도 감추어버릴 수 있습니다.

타입스크립트에서 `string`의 부분 집합을 정의할 수 있는 기능은  
자바스크립트 코드에 타입 안전성을 크게 높입니다.