

타입스크립트 도입 전에 @ts-check 와 JSDoc으로 시험해 보기

이펙티브 타입스크립트

@ts-check

@ts-check 지시자를 사용하면
의미 있는 오류들을 찾아 낼 수 있습니다.

선언되지 않은 전역 변수

어딘가에 ‘숨어 있는’ 변수라면, 변수를 제대로 인식할 수 있게 별도의 타입 선언 파일을 만들어야 합니다.

```
// @ts-check
console.log(user.firstName);
// ~~~~ 'user' 이름을 찾을 수 없습니다.
```

```
// @ts-check
/// <reference path="./types.d.ts" />
console.log(user.firstName); // 정상
```

```
interface UserData {
  firstName: string;
  lastName: string;
}
declare let user: UserData;
```

알 수 없는 라이브러리

서드파티 라이브러리를 사용하는 경우,
서드파티 라이브러리의 타입 정보를 알아야 합니다.

```
// @ts-check
$('#graph').style({'width': '100px', 'height': '100px'});
// ~ '$' 이름을 찾을 수 없습니다.
```

```
// @ts-check
$('#graph').style({'width': '100px', 'height': '100px'});
// ~~~~~ 'jQuery<HTMLElement>' 형식에 'style' 속성이 없습니다.
```

DOM 문제

DOM 엘리먼트와 관련된 부분에 많은 오류가 표시된다.

```
// @ts-check
const ageEl = document.getElementById('age');
```

```
ageEl.value = '12';
// ~~~~ 'HTMLElement' 유형에 'value' 속성이 없습니다.
```

```
// @ts-check
const ageEl = /** @type {HTMLInputElement} */(document.
getElementById('age'));
ageEl.value = '12'; // 정상
```

@ts-check

@ts-check 지시자를 활성화하면 이미 존재하던
JSDoc에서 부작용이 발생하기도 합니다

부정확한 JSDoc

아래의 예제에서는 두 가지의 오류가 발생합니다.

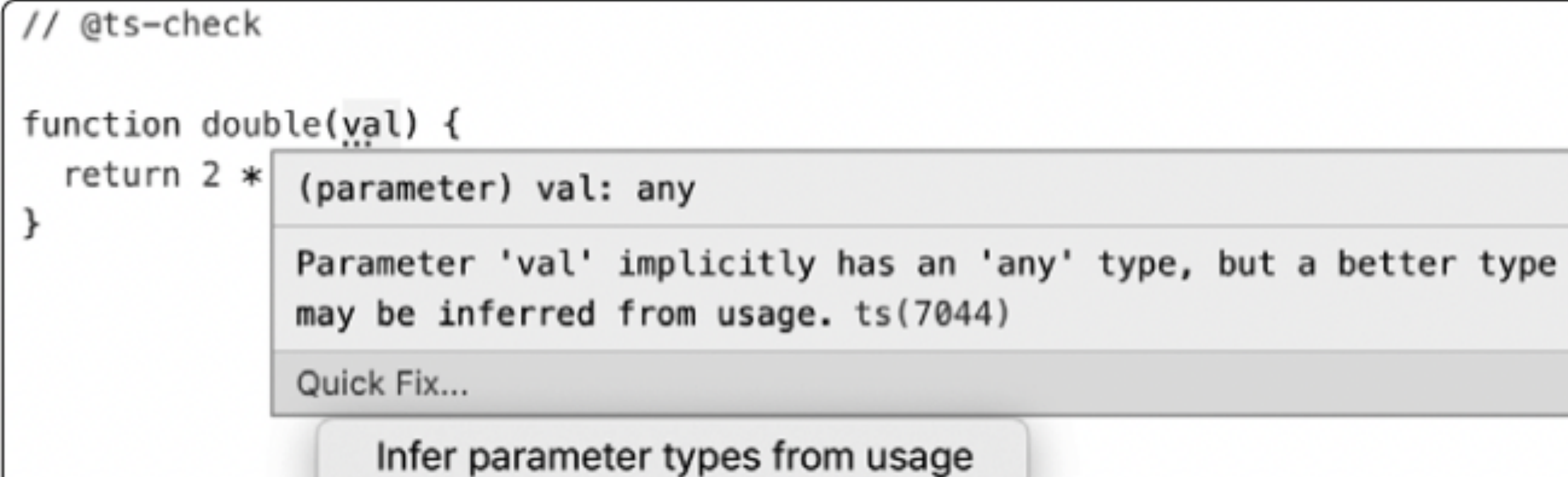
- DOM의 타입 불일치
- @return 타입에 명시된 타입과 실제 반환 타입의 차이

```
// @ts-check
/**
 * 엘리먼트의 크기(픽셀 단위)를 가져 옵니다.
 * @param {Node} el 해당 엘리먼트

 * @return {{w: number, h: number}} 크기
 */
function getSize(el) {
  const bounds = el.getBoundingClientRect();
  // ~~~~~ 'Node' 형식에
  // ~~~~~ 'getBoundingClientRect' 속성이
  // ~~~~~ 없습니다.
  return {width: bounds.width, height: bounds.height};
  // ~~~~~ '{ width: any; height: any; }' 형식은
  // ~~~~~ '{ w: number; h: number; }'에 할당할 수 없습니다.
}
```

부정확한 JSDoc

타입스크립트 언어 서비스는 타입을 추론해서 JSDoc을 자동으로 생성해줍니다.



```
// @ts-check

function double(val) {
  return 2 *
}
```

(parameter) val: any

Parameter 'val' implicitly has an 'any' type, but a better type may be inferred from usage. ts(7044)

Quick Fix...

Infer parameter types from usage

그림 8-2 타입스크립트 언어 서비스는 타입을 추론하여 타입 정보를 자동으로 생성해 줍니다.

결론

- 자바스크립트 환경에서도 @ts-check 지시자와 JSDoc 주석을 사용하면 타입스크립트와 비슷한 경험으로 작업이 가능하다.
- 하지만, @ts-check 지시자와 JSDoc 주석을 장기간 사용하는 것은 좋지 않다.
=> 주석이 코드 분량을 늘려서 로직 해석에 방해가 될 수 있기 때문