8조

Typescript Interface 사용하기

```
interface ClockInterface {
    currentTime: Date;
}

class Clock implements ClockInterface {
    currentTime: Date = new Date();
    constructor(h: number, m: number) { }
}
```

Typescript Interface 사용하기

- One of TypeScript's core principles is that type checking focuses on the *shape* that values have. - http://www.typescriptlang.org/docs/handbook/interfaces.html-

Typescript Interface 사용하기

```
interface LabeledValue {
    label: string;
}

function printLabel(labeledObj: LabeledValue) {
    console.log(labeledObj.label);
}

let myObj = {size: 10, label: "Size 10 Object"};
printLabel(myObj);
```

Here, it's only the shape that matters.

Typescript Interface 사용하기

```
interface SquareConfig {
    color?: string;
    width?: number;
}

function createSquare(config: SquareConfig): {color: string; area: number} {
    let newSquare = {color: "white", area: 100};
    if (config.color) {
        newSquare.color = config.color;
    }
    if (config.width) {
        newSquare.area = config.width * config.width;
    }
    return newSquare;
}

let mySquare = createSquare({color: "black"});
```

Typescript Interface 사용하기

```
interface SquareConfig {
    color?: string;
    width?: number;
}

function createSquare(config: SquareConfig): { color: string; area: number } {
    let newSquare = {color: "white", area: 100};
    if (config.clor) {
        // Error: Property 'clor' does not exist on type 'SquareConfig'
        newSquare.color = config.clor;
    }
    if (config.width) {
        newSquare.area = config.width * config.width;
    }
    return newSquare;
}

let mySquare = createSquare({color: "black"});
```

Typescript Interface 사용하기

```
interface SquareConfig {
    color?: string;
    width?: number;
}

function createSquare(config: SquareConfig): { color: string; area: number } {
    // ...
}

let mySquare = createSquare({ colour: "red", width: 100 });
```

Typescript Interface 사용하기

```
// error: Object literal may only specify known properties, but 'colour' does not exist
in type 'SquareConfig'. Did you mean to write 'color'?
let mySquare = createSquare({ colour: "red", width: 100 });
```

Typescript Interface 사용하기

```
let mySquare = createSquare({ width: 100, opacity: 0.5 } as SquareConfig);
```

```
interface SquareConfig {
    color?: string;
    width?: number;
    [propName: string]: any;
}
```

```
let squareOptions = { colour: "red", width: 100 };
let mySquare = createSquare(squareOptions);
```

어려운 점

- 협업
  - 실력 차이
  - 남이 생각하는 방식 이해
  - 팀

싸움, 갈등?

Ground Rule

1. 버그나 고민사항 있으면 이슈에 올려 공유할 수 있도록한다.
    - 매일 아침 스크럼 시간에 확인
    - [HELP] issue template 사용하기
2. 회의시간이나 스크럼은 존댓말로 진행한다.
    - 스크럼은 개인 3분 => 총 12분 안에 끝내기, 15분 Q&A 및 작성
3. 페어 프로그래밍을 권장한다.
4. 혼자 해결하기 어려운 문제가 생겼을 때 '주저'하지말고 바로 말하기. 응답자는 질문자를 질타하지 않는다.
5. 응답자는 질문에 성실히 임할 의무가 있다. 섭섭한 일이 있으면 담아두지 않고 바로바로 말하며 감정적으로 격해지면 존댓말을 사용하여 팀을 분해시킬 수 있는 말들을 삼가할 수 있도록 한다. 비속어 사용을 지양한다.

해결 과정

- 짝 프로그래밍
  - 실력
  - 남이 생각하는 방식
- 대화
  - 회의
  - 존중
- 기술공유
  - 팀 슬랙

기여

- 고승빈 : 백엔드 구조 설계 && 팀의 중심
- 김경래 : 백엔드 구조 설계 && 팀의 재정(총무)
- 김기표 : 프론트 구조 설계 && 팀의 귀염둥이
- 이상원 : 발표 자료 준비 && 팀의 활력소