CHAPTER 7

신뢰할 수 없는 코드를 쓰면서 불변성 지키기

@mugglim

안전지대(Safe zone)

불변성이 지켜지는 코드 영역

코드를 수정할 수 있는 영역

신뢰할 수 있는 영역

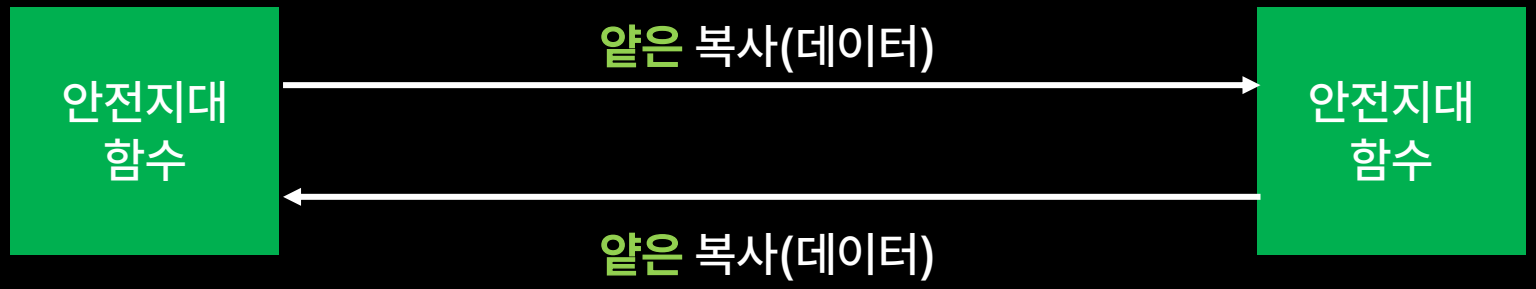
비 안전지대(No safe zone)

불변성이 지켜지지 않는 코드 영역

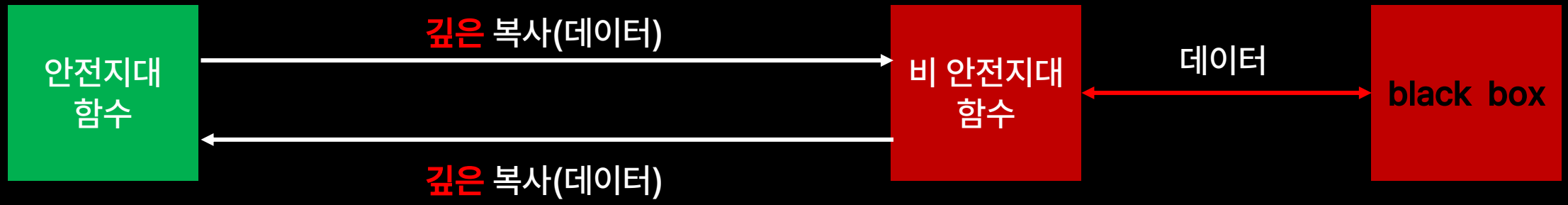
코드를 수정할 수 없는 영역
(ex) 라이브러리, 오래된 레거시 코드)

신뢰할 수 없는 영역

안전지대에서 함수 간 데이터 통신



안전지대 함수와 비 안전지대 함수 간 데이터 통신



깊은 복사(Deep clone) = 방어적 복사

```
function deepClone(target) {
    if (Array.isArray(target)) {
        return target.reduce((copy, value) => {
            copy.push(deepClone(value));
            return copy;
        }, []);
    } else if (target !== null && typeof target === 'object') {
        return Array.from(Object.keys(target)).reduce((copy, key) => {
            copy[key] = deepClone(target[key]);
            return copy;
        }, {});
    }

    return target;
}
```

왜 깊은복사를 해야 하나?

불변성을 지키기 위해!!

1. 안전지대에서 비 안전지대로 보내진 데이터가
변경 될 수 있음

2. 비 안전지대에 생성된 데이터는
이후에 변경 될 수 있음

생각 정리

1. 데이터 복사 과정에서의 성능과 불변성은 트레이드오프 관계

2. 결국은 데이터를 복사하는 과정에서의 성능을 최적화하기 위해 **깊은 복사를 지양**하고, **얕은 복사를 지향**!

3. 개발 단계가 아닌 설계 단계에서 어떻게 하면 깊은 복사를 최소화 할 수 있을지 고민이 필요

끄읏! Q&A