

# When Errors are No Longer Scary

<https://product.kyobobook.co.kr/detail/S000213494491>

바로 해결할 수는 없더라도 힌트를 발견할 수 있다.

에러를 만날 때마다 새로운 것을 배울 수 있다.

개발자의 업무 시간 40%는 에러에 대한 조사와 수정에 소요된다.

에러의 구성요소

- 에러의 종류
- 에러 메시지
- 스택 트레이스
- : 에러 발생의 흐름까지 나타낸 것
- : 함수가 어떤 순서로 호출되는지 표시하는 이력 정보
- : 최종 지점에서 에러가 발생한 것이므로 먼저 해당 부분을 살펴보는 것이 효율적
- : 모든 행을 읽으려고 하지 말고 최종 지점의 첫 번째 행부터 읽으면 문제를 쉽게 파악 가능

에러의 유형 예시

- syntax error : 표기 방식의 문제
- reference error
- type error : 인수값 확인
- range error
- parse error
- value error

- no method error
- argument error
- runtime error
- name error
- attribute error
- import error
- index error
- key error

## 디버깅

: 에러의 실제 원인을 찾고 수정하는 작업

: 정보를 눈으로 확인하면서 확실하게 에러를 확인 할 수 있는 작업

## 이진탐색

프론트, 백엔드, 서버, 데이터 베이스

최소한의 코드로 에러를 재현

코드를 물리적으로 제거하고 의심스러운 부분을 조금씩 줄여나가는 것

한 번에 여러개의 가설을 검증하기보다 처음 세운 가설이 타당한지 검증하는 것에 집중한다

작은 의문에 귀를 기울인다

테디 베어 효과

러버덕 디버깅

## 디버거

: 프로그램 실행 중에 특정 위치에서 작업을 중단하게 하는 도구

Xdebug

debug.gem

브레이크 포인트

: 실행 중인 프로그램을 임의의 위치에서 일시 정지하는 기능 제공

: 프로그램 상태 관찰 가능

step into

step over

step out

변수 감시

에러를 프로그램에서 처리 ~ uncaught

에러 트래킹 도구

: sentry

: rollbar

로그 도구

: logflare

: papertrail

: longtail

: datadog

workaround 회피술

재할당 자제하기

스코프 최소화하기

단일 책임의 원칙 지키기

순수 함수 활용하기

타입 의식하며 코드 작성하기

테스크 코드 활용하기

동적 타입 언어

정적 타입 언어