GitHub Actions CI 실패 진단 키트

재현부터 최소 패치와 유지관리자 보고까지

빨간 체크를 코드·인프라·계정·외부 상태로 분류하는 증거 중심 워크플로

v1.0 · 2026

1. CI 실패는 곧 코드 결함이 아니다

GitHub Actions가 빨간색이라는 사실은 결과이지 원인이 아니다. 실패는 현재 변경이 만든 회귀일 수도 있고, 기본 브랜치에도 존재하는 인프라 결함일 수도 있다. 필요한 권한이나 법적 동의가 빠진 상태일 수도 있으며, 취소된 이전 실행이 화면에 남아 있을 수도 있다.

이 키트의 목적은 실패를 성급히 수정하는 것이 아니라 누가, 어디에서, 무엇을 고쳐야 하는지 증거로 분류하는 것이다.

> 첫 질문은 “어떻게 초록색으로 만들까?”가 아니라 “이 실패를 현재 변경이 만들었다는 증거가 있는가?”다.

네 가지 결과

- 코드로 수정할 회귀
- 워크플로·러너·외부 서비스 문제
- 계정 소유자 또는 관리자 작업
- 외부 상태를 기다려야 하는 정상 대기

2. 사용할 수 있는 범위와 한계

이 가이드는 GitHub Actions 실패의 원인 분류, 재현, 최소 수정, 검증 증거, 유지관리자용 보고 형식을 제공한다.

포함하는 것

- 실패 Run과 Job을 좁히는 순서
- 기본 브랜치와 비교하는 방법
- 코드·인프라·권한·법적 게이트 분류
- 재현 명령과 환경 기록
- 최소 패치와 회귀 테스트 기준
- 유지관리자에게 전달할 증거 템플릿

포함하지 않는 것

- 모든 저장소에 통하는 자동 수정
- 비공개 저장소 접근권한 우회
- CLA, DCO, 약관의 대리 동의
- 클라우드 비용 또는 보안 정책 승인
- 테스트 통과만으로 제품 동작이 맞다는 보증

> CI를 초록색으로 만드는 것과 변경이 옳다는 것은 서로 다른 목표다.

3. 5분 초기 분류

실패 화면을 본 즉시 다음 다섯 항목을 기록한다.

1. 저장소와 PR 또는 커밋 URL 2. 실패한 Workflow, Job, Step의 정확한 이름 3. 최초 실패 시각과 마지막 성공 시각 4. 기본 브랜치의 같은 Workflow 상태 5. 첫 번째 의미 있는 오류 줄

분류 질문

- 컴파일이나 테스트가 시작되기 전에 실패했는가?
- `permission`, `secret`, `token`, `CLA`가 보이는가?
- 같은 오류가 기본 브랜치에서도 재현되는가?
- 운영체제 또는 도구 버전 한 곳에서만 실패하는가?
- 유지관리자가 변경을 요청했는가?

로그의 마지막 줄만 복사하지 않는다. 실패를 만든 명령, 그 직전의 환경 정보, 첫 오류를 함께 보관한다.

4. 증거 수집 시트

아래 블록을 작업 노트에 복사한다.

```
Repository:
Pull request / commit:
Workflow:
Job:
Step:
Runner:
Toolchain:
First failing command:
First meaningful error:
Base branch comparison:
Local reproduction:
Owner of next action:
```

좋은 증거의 조건

- 다른 사람이 같은 명령을 실행할 수 있다.
- 성공과 실패의 차이가 한 가지 이상 명시된다.
- 로그 전체 대신 관련 구간과 원본 URL을 함께 제공한다.
- 비밀값, 토큰, 고객 데이터가 제거돼 있다.

5. 실패 소유권 결정표

코드 소유

현재 변경을 제거하면 사라지고, 로컬 또는 같은 환경에서 재현되며, 실패 명령이 수정 파일을 직접 사용한다.

워크플로 소유

소스 컴파일 전에 러너 도구가 없거나, 캐시·아티팩트·서비스 컨테이너·권한 설정 때문에 실패한다.

계정 소유자

CLA, DCO 확인, 조직 승인, 비밀키, 결제, 배포 권한처럼 코드로 해결할 수 없는 단계다.

외부 소유

GitHub 장애, 패키지 레지스트리 장애, 외부 API 제한처럼 현재 저장소 밖에서 복구돼야 한다.

> 소유권이 정해지지 않은 실패에 패치를 추가하면 두 번째 장애를 만든다.