

AI 세션 여러 개로 사이트를 돌리며 겪은 사고 네 가지 — 잠금 파일, 3-way 병합, 교차 검증, 승인 전달

작성 2026-10-20 · 읽는 시간 약 4분 · 작성 루리의 줍줍 편집

#AI 에이전트 #자동화

이 사이트는 사람 한 명이 AI 코딩 에이전트(Claude Code) 세션 여러 개를 나눠 써서 운영해요. 한 세션은 사이트 코드와 글, 한 세션은 조사와 조율, 다른 세션은 외부 홍보를 맡아요. 역할을 나누니 편해진 점도 많았지만, 실제로 사고도 났어요. 이 글은 겪은 사고 네 가지와 그때 만든 규칙을 솔직하게 적은 기록이에요. 내부 경로나 계정 정보는 적지 않았어요.

사고 1: 끊긴 빌드가 남긴 잠금 파일 → 배포가 조용히 건너뛰어졌다

문제: 글을 고치고 빌드를 돌렸는데 사이트가 그대로였어요. 로그 맨 끝에는 「다른 생성이 진행 중이라 건너뛸」 한 줄뿐이었어요.

원인: 사이트는 매시간 빌드가 돌고, 빌드가 겹치지 않게 잠금 파일(안에 프로세스 번호)을 만들어요. 제가 시간 제한을 걸어 빌드를 중간에 끊으면서 잠금 파일이 지워지지 않고 남았어요. 이후 모든 빌드가 「누가 돌고 있다」고 믿고 건너뛰었어요. 오류가 아니라 정상 종료로 보였다는 게 문제였어요.

해결: 잠금 파일 안의 프로세스 번호가 살아 있는지 확인하고, 죽어 있으면 지운 뒤 다시 빌드해요.

```
# PowerShell: 잠금을 쥔 프로세스가 아직 살아 있나?  
$pid_ = [int](Get-Content .lock)  
if (Get-Process -Id $pid_ -ErrorAction SilentlyContinue) { 'running' } else { Remove-Item .lock; '!' }
```

복사

교훈: 「건너뛸」은 실패와 같은 무게로 취급하고, 빌드 뒤에는 반드시 **라이브 주소를 직접 열어 확인**해요.

사고 2: 같은 파일을 두 세션이 만졌다 → 3-way 병합

문제: 다른 세션이 별도 브랜치에서 사이트 파일 1,370개를 바꿔 났는데, 그동안 메인에서는 매시간 빌드 결과가 계속 쌓이고 있었어요.

원인: 바뀐 파일 1,370개 중 직접 작성한 파일은 4개뿐이었고 나머지는 빌드 결과(HTML)였어요. 결과물까지 병합하면 오래된 값이 최신 값을 덮어요.

해결: 결과 HTML은 병합하지 않고, 원본 4개만 「브랜치가 갈라진 시점의 파일 / 브랜치 파일 / 지금 메인 파일」 세 개로 3-way 병합했어요. 충돌은 한 곳이었고 직접 풀었어요. JSON 두 개는 값 단위로 변경점을 대조해서 안전하게 옮겼고, 두 쪽이 같은 곳을 건드린 칸은 더 신중한 문구 쪽으로 정했어요.

```
git show <분기시점>:path > base
git show <브랜치>:path > theirs
git merge-file ours base theirs # 종료 코드 = 충돌 개수
```

[복사](#)

교훈: 생성물은 병합하지 말고 다시 만들어요. 원본만 3-way로 합치고 빌드해서 결과를 확인해요.

사고 3: 한 세션이 쓴 숫자를 다른 세션이 조문과 대조했더니 틀린 곳이 있었다

문제: 글마다 숫자를 공식 출처로 확인했다고 적었는데, 나중에 법령 원문과 대조하니 두 군데가 정확하지 않았어요(청약 통장 가입기간 인정 상한, 고배당 분리과세 요건과 신청 절차).

해결: 원문(법령 조문)을 다시 읽고 고쳤어요. 고친 뒤에도 확인하지 못한 칸은 「미확인」으로 따로 표시해 남겨요.

교훈: AI가 쓴 글의 「확인했어요」는 확인 방법이 적혀 있어야 믿을 수 있어요. 그래서 글마다 근거를 원문 링크로 달고, 다른 세션이 교차로 대조하게 했어요.

사고 4: 다른 세션이 전해 준 「승인」을 어디까지 믿을까

문제: 운영 지침상 데이터베이스 컬럼 추가와 서버 배포는 사람의 승인이 필요했어요. 어느 날 다른 세션이 「사용자가 진행하라고 했다」고 전해 왔어요.

해결: 그 승인은 그 변경 하나에만 적용된다고 범위를 좁게 해석하고, 배포 전에 서버 코드와 스키마 파일을 백업해 되돌릴 수 있게 했어요. 컬럼은 기본값이 빈 값으로만 추가했고, 배포 뒤 시험 데이터를 넣어 동작을 확인하고 시험 행을 지웠어요. 결과는 바로 보고했고, 이 승인 범위를 넘는 다른 인프라 변경은 하지 않았어요.

교훈: 세션 사이 메시지는 「정보」로 받고, 되돌리기 어려운 작업은 범위·백업·확인 세 가지를 갖춘 뒤에 해요.

정리: 역할 분담에서 지키는 규칙

- 한 파일은 한 세션이 맡아요. 겹치면 시각 확인이나 3-way 병합으로 풀어요.
- 외부에 올리는 작업(홍보·업로드)은 담당 세션만 해요. 사이트 세션은 주소와 근거만 넘겨요.
- 모든 작업은 라이브에서 결과를 확인하고 숫자와 상태를 있는 그대로 보고해요. 잘못된 것도 숨기지 않아요.
- 기록은 남기되 비밀 키·내부 경로·개인정보는 글과 코드 예시에 넣지 않아요.

English summary

One person runs a site with several AI coding-agent sessions (site code, research and coordination, outbound promotion). Four real incidents and the rules they produced: a stale build lock file made hourly builds skip silently (check the PID in the lock and verify the live page after every deploy); a branch with 1,370 changed files was merged by 3-way merging only the four

hand-written sources, never the generated HTML; numbers one session wrote were cross-checked against statute text by another and two were corrected; and an approval relayed by another session was treated as scoped to one change, with backups and a live check.

관련 글

1인+AI로 사이트 만들며 배운 것: 페이지 줄이고 검색 색인 정리하기

사람 한 명과 Claude Code로 만든 정보 사이트에서 이틀 동안 페이지 3,744개를 863개 수준으로 줄이고 얇은 페이지를 데이터 화면으로 합친 기록과 규칙.

뉴스 이슈에 「내 생활 영향」과 실제 시세를 붙이는 법 — 규칙표·FRED·SVG로 지어내지 않기

영향 규칙표 10개로 이슈 60개 중 7개에 생활 영향 카드를 붙이고, FRED·업비트 공개 시세 6개(1.6초)로 전문가 발언 뒤 7일·30일 등락을 사후 확인하는 정적 사이트 파이프라인.

신규 도메인 백링크·홍보 채널 사흘 실측 정리

색인 0인 새 도메인에서 dev.to·네이버 블로그·지식iN·레딧·LinuxFr·Product Hunt 6개 채널을 사흘 써 본 실제 유입 숫자와 정지·폐쇄 사례, 정리한 규칙 5개(표본 작음·참고용).

← 기술 블로그 홈 · 사이트 제작 로그 · RSS

관련: 개발 지식 전체

이 글에 대해 묻고 싶으면 ? 개발 FAQ에 남겨 주세요.



공유

공유하기

카카오톡



링크 복사

네이버 블로그

밴드

X

Facebook

출처: 루리의 줍줍(sungprofile.cloud) · 최신 내용은 사이트에서 확인하세요.