

Software Engineering

Hands-On Lecture – 01

Software Engineering in the era of AI

Traditional Software Engineering

- 전통적 소프트웨어 엔지니어링은 **결정적인 환경**에서 설계·구현됨
 - 요구사항 개발 : What + How -> Expected Result -> API + Document
- 기존의 코드 패러다임은 **구문 숙달**(Mastery of Syntax)과 **저-수준 연산**(Low-level Operations)의 체계적인 결합임
 - 엔지니어는 안정적인 구현과 유지보수, 개선을 위해 전문적인 프로그래밍 실력을 가져야 함
 - 이 과정에서 단순한 코딩 작업이 반복됨
- 소프트웨어 공학은 사실상 문서와의 씨름에 가까움
 - Architect는 요구사항을 식별하여 필요한 프로그램의 명세서를 작성함
 - PM은 요구사항을 프로젝트 단위로 분해하여 모듈식 개발을 수행 (물론 이상적인 환경에서)
 - 개발자는 문서를 읽고 코딩을 하고 문서를 다시 쓰고 ...

End of Traditional Software Engineering

- 이러한 전통적 소프트웨어 개발 구조에 끝이 찾아옴



프로그래밍의 초기 단계:

물리적 회로를 연결하여 프로그래밍 / 천공 카드 (어셈블리 코드)

컴파일 언어의 출현:

높은 추상화 수준(인터프리터 언어)에서 한번 Wrapping 하여 프로그래밍

프로그래밍의 새 시대:

자연어 기반 프로그래밍

End of Traditional Software Engineering

- 바보 같았던 AI의 똑똑한 진화
- 2022 : Chat GPT -> 코드 스니펫 제안
- 2023 : Claude / Copilot / GPT 4 / 리플릿 -> 훌륭한 조연자
- 2024 : 멀티 모달 -> 코딩 파트너
- **2025 : 바이브 코딩 (지금)**
- 2026 : 코딩 에이전트

End of Traditional Software Engineering

- 프로그래밍 관점에서 AI의 우수한 능력
 - AI는 수천 줄의 코드를 생성하는 데 매우 능숙
 - 프로그래밍 언어 간 번역, 사용자 인터페이스 생성, 버그 수정, 반복적인 작업과 패턴 인식에 탁월
- 그럼에도 불구하고 ...
 - AI는 우리가 요청하는 모든 작업 뒤에 있는 이유를 이해하지 못함
 - 실제 상황과 시나리오에 대한 인간의 입력이 필요
 - 신뢰할 수 없음 -> 작성해주는 코드의 70% 이상은 허무맹랑함 (환각)

Vibe Coding

• 바이브 코딩이란 ?

- 인간 프로그래머가 코드의 직접적인 구현자라기보다는, 반복적인 프롬프팅과 전략적 지시를 통해 LLM과 협력하는 상위 수준의 조정자로서 기능하는 새롭고 창의적인 소프트웨어 개발 방식을 의미

• 기존 코딩 방식

- 인간 프로그래머가 프로그래밍 구문 규칙에 따라 논리를 명시

지루하고
현학적임

가끔 즐기는 분들도 계심

• 바이브 코딩

- 자연어 지시
- 개념적인 개요를 전달
- 점진적 개선을 요청
- 분위기(vibe)를 전달

간단하고
뇌를 덜 쏘
스트레스 ▼
우울감 ▼

뇌를 아예 비우는 분들도 계심

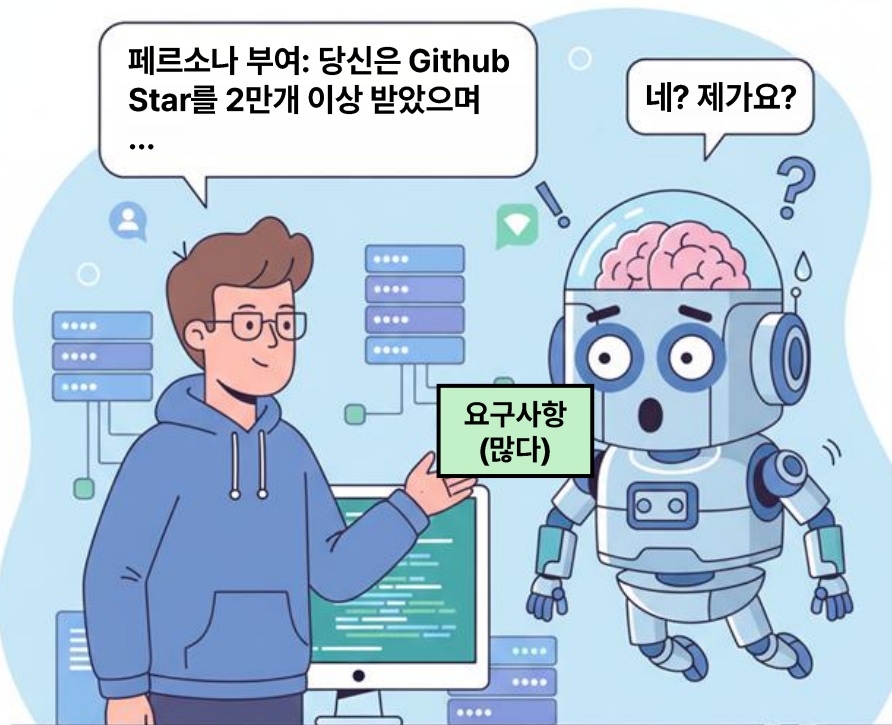
안드레이 카파시
(Andrej Karpathy)



Vibe Coding

• 바이브 코딩의 핵심

- 바이브 코딩의 핵심은 프롬프트 엔지니어링, 애자일 디자인, 인간-AI 공동 창작의 원칙을 통합하면서 언어적 부담의 상당 부분을 LLM에 추상화하는 것

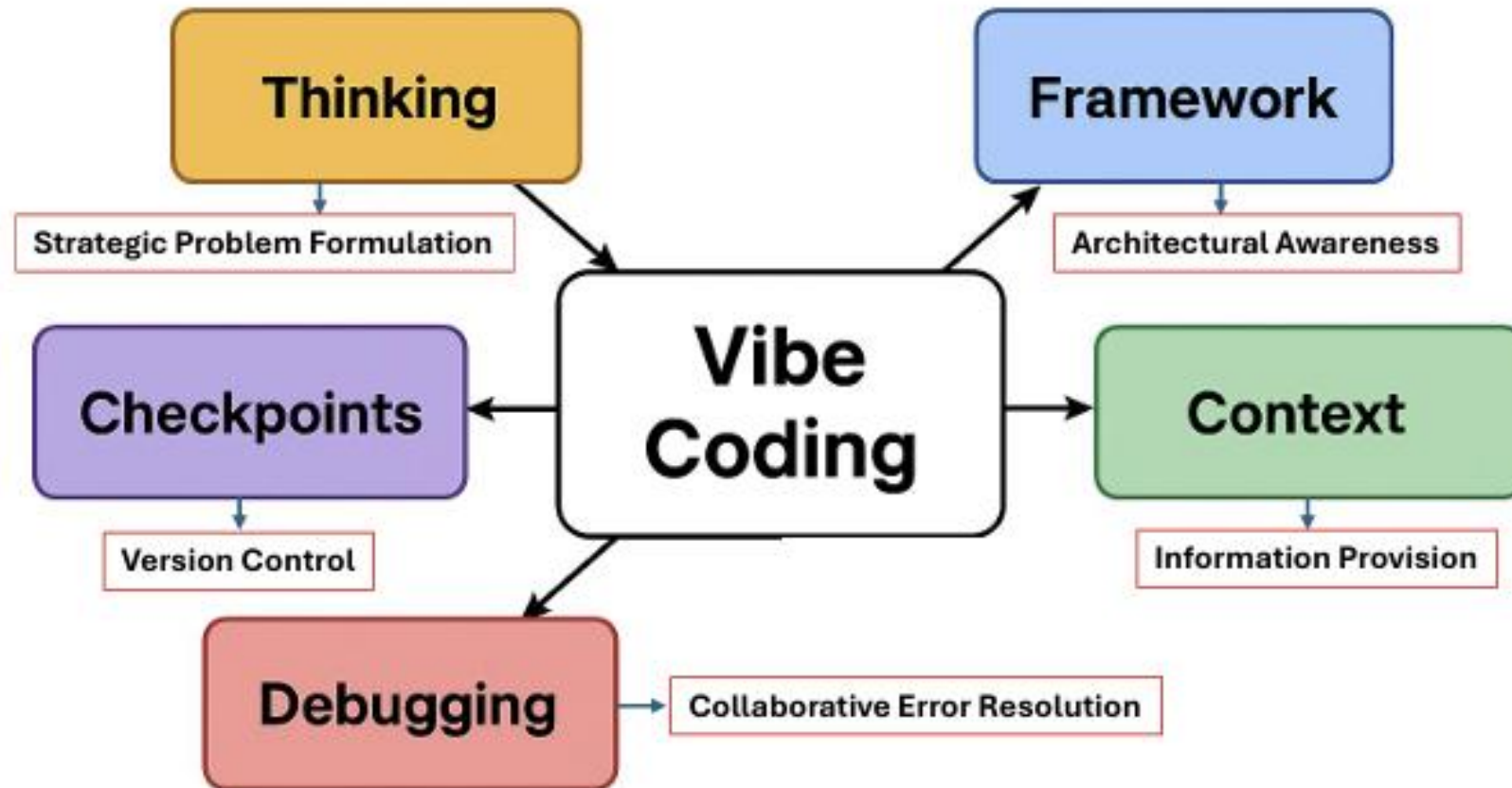


Vibe Coding

- **바이브 코딩의 인지적 전환**
 - 바이브 코딩은 개발자의 인지 능력을 증강시킴 (Ideal)
 - 저수준 구현 세부 사항의 부담을 덜어줌
 - 창의적인 문제 해결, 사용자 경험 디자인, 시스템 아키텍처에 더 집중
 - 효과적인 바이브 코딩은 개발자 기술의 전환을 필요
 - 구문의 암기보다는 개념적 표현과 전략적 상호작용을 강조

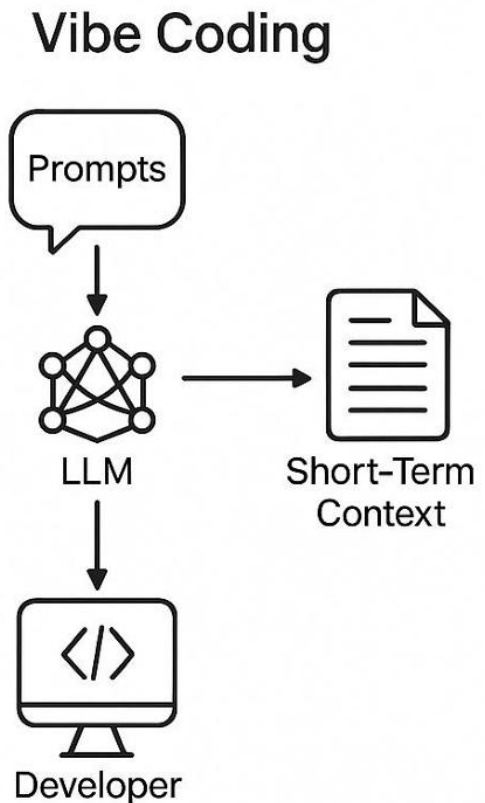
Vibe Coding

- 바이브 코딩의 필요 기술들



Vibe Coding 인터페이스

- CLI (Command-Line Interface)



```
* Subscribers View

• List(Modules/Tests)
  | Listed 57 paths (ctrl+r to expand)

• Great! I can see there's already a StringRankedSearchTests.swift file in the
  WordPressSharedTests directory. Let me examine the existing tests to understand what's
  already covered.

• Read(Modules/Tests/WordPressSharedTests/StringRankedSearchTests.swift)
  | Read 85 lines (ctrl+r to expand)

• I can see there are already some tests, but they could be expanded to cover more
  functionality. Let me enhance the test file with more comprehensive unit tests covering all
  the public methods and edge cases.

• Update(Modules/Tests/WordPressSharedTests/StringRankedSearchTests.swift)
  | Updated Modules/Tests/WordPressSharedTests/StringRankedSearchTests.swift with 1 addition
  and 1 removal
  1   import XCTest
  2   - import WordPressShared
  2   + @testable import WordPressShared
  3
  4   final class StringRankedSearchTests: XCTestCase {
  5       func testScoreInRange() {

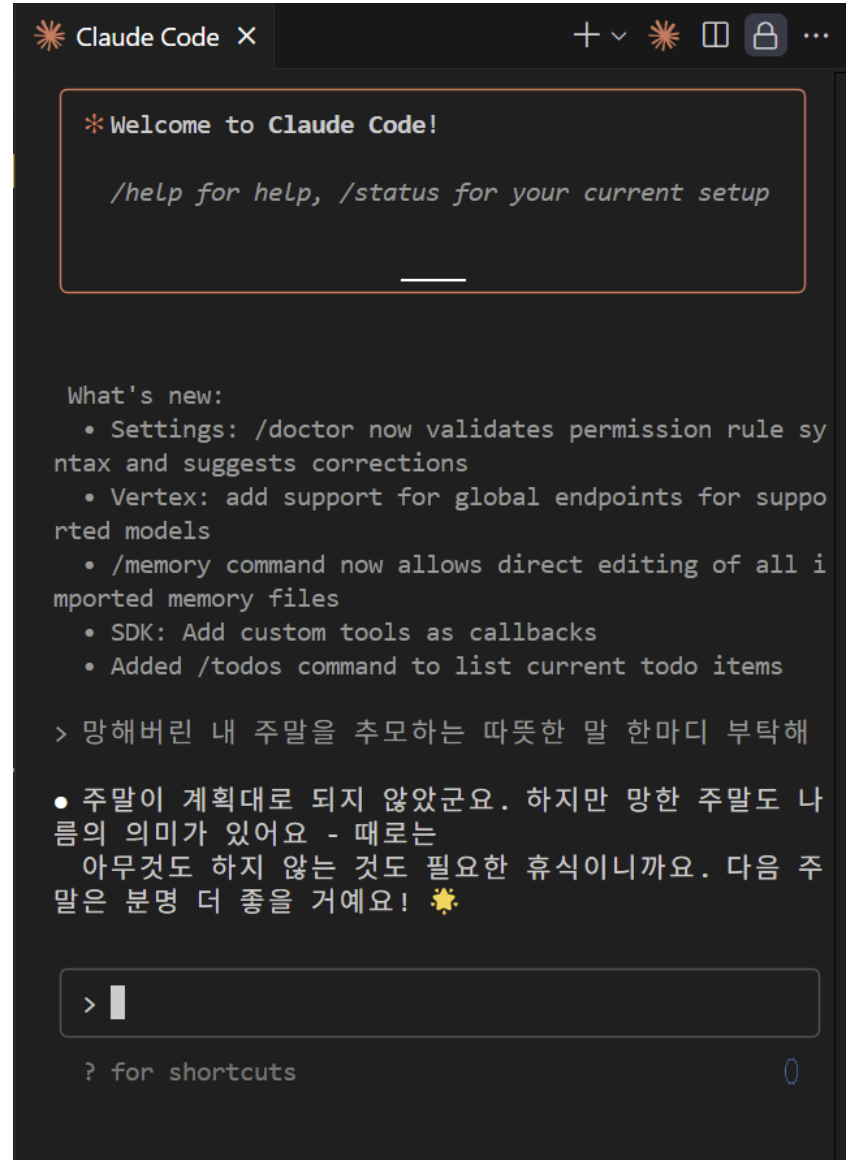
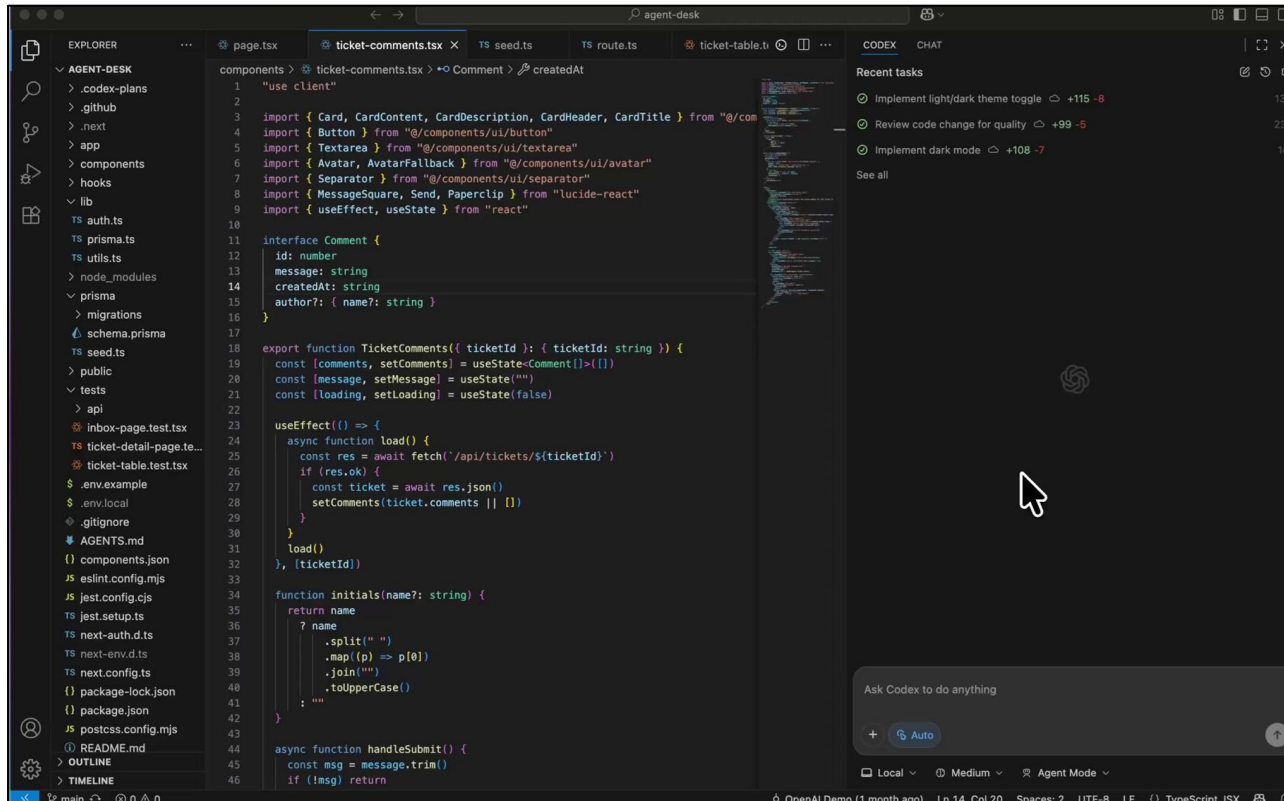
* Searching... (27s · × 425 tokens · esc to interrupt)

> |

▶▶ auto-accept edits on (shift+tab to cycle)
```

Vibe Coding 인터페이스

• IDE Extension



Vibe Coding 시대의 문제점 (논의)

Homework :: Gemini 1년무료 계정 만들기

Google Gemini의 Pro 버전, 대학생에게만 1년간 무료!

2.5 Pro 모델, Deep Research, AI 오디오 오버뷰에 확장된 액세스 권한과 더불어 채팅, 이미지 업로드, 퀴즈 생성 기능을 무제한으로 이용하고 2TB의 스토리지를 활용하세요. 1년간 무료로 사용할 수 있습니다. 대학생 전용 혜택이며, 무료 이용 혜택 등록기간은 2025년 10월 6일까지입니다.

[혜택 받기](#)[약관 확인](#)

[Gemini for Students — 1년간 무료로 Google AI Pro를 이용하세요](#)