Practical AI Tooling Tips For The Too Busy Developer

Slides at scottsauber.com

@scottsauber

Audience

- Looking for ideas how to best leverage AI
- Been too busy shipping value to pay attention
- This’ll be a 100-200 level talk
- More than just “AI in your IDE”
- This an all day workshop, but we have 60 mins

Poll

- Who’s using AI tools? Every day?
- Copilot?
- Claude Code?
- Cursor?
- Windsurf?
- Something else?

Agenda

- Am I still going to have a job?
- Define all the AI Buzzwords like Context, Instructions, Skills, and more
- Talk about tools like Copilot and Claude Code
- Show some demos
- Practical Tips about where AI fits and where it doesn’t

Goals

- Give you ideas you can start using today
- Where AI tooling is going in the future

Who am I?

- Director of Engineering at Lean TECHniques
- Microsoft MVP
- Dometrain Author
- Redgate Community Ambassador
- Co-organizer of Iowa .NET User Group

Am I going to be out of a job?

History Lesson (2000s – Present)

- What set teams apart was their commitment to modern engineering practices and modern work practices – proven by DORA
- Fast feedback
- Working in small batches
- Learning Culture
- Automated Tests
- CI/CD
- Cloud
- Database Automation
- Monitoring and Observability
- And more…

Which ones are still relevant in an AI-dominated world?

History Lesson (2000s – Present)

- What set teams apart was their commitment to modern engineering practices and modern work practices
- Fast feedback ✅
- Working in small batches ✅
- Learning Culture ✅
- Automated Tests ✅
- CI/CD ✅
- Cloud ✅
- Database Automation ✅
- Monitoring and Observability ✅
- And more…

The Fundamentals Remain The Same

Jevon's Paradox

- Economic Principle of increasing the the efficiency of a resource actually leads to more use, not less
- More fuel efficient engines, means cheaper travel, which encourages people to drive, which means more fuel consumption
- Faster Internet speeds has led to more internet usage not less (ie streaming)
- If history repeats itself - becoming more efficient with creating software will actually result in MORE software, not less
- Building faster changes the Build vs Buy equation
- We all still have jobs!

“SaaS is Dead.”

- Satya Nadella, Microsoft CEO

These tools are the worst they will ever be today.

Alright let’s talk AI Tools

AI is an Amplifier

- Good practices – move faster at high quality
- Bad practices – move faster at lower quality
- How do you verify your app works? Compile, Tests, Lint, Format. Are you doing those things already?
- Do you have judgement to know when AI went off the rails?
- Your PR output might 2-10x when using AI correctly
- Do your PRs sit for days without reviewing already?
- Do you rubberstamp PRs?
- You still own the code – not AI

Large Language Models

- Engine trained on massive amounts of text to learn patterns of meaning
- Literally just predicts one token at a time based on probabilities
- Often people just say Models
- Best Models is a misnomer – it’s tradeoffs
- Accuracy vs Cost vs Speed
- Claude Opus 4.5 is the “best” model for one-shotting code in my experience, but it’s the most expensive and often slow
- Haiku responds very quickly, but isn’t as deep

Context

- The information an LLM is allowed to consider for a request
- LLMs have a limited Context Window
- What makes up Context:
- Conversation History
- Code in your repo
- User Prompt
- Instructions
- Clearing context is key – when one task is done, start a new chat
- Compacting can also lead to hallucinations

AI Dev Tools

- Unifies LLMs with codebases to provide the most context to get the best results
- GitHub Copilot (most ubiquitous)
- Claude Code (best AI dev tool)
  - Preview Feb ‘25, GA May ‘25
- Cursor (best autocomplete)
- Lots of others
- Best == my opinions and experiences

Modes

- Ask
  - Answers a question. ChatGPT
- Plan
  - Formulates your Prompt to feed to Agent/Edit mode
  - Asks you questions to clarify intent
  - You should be using Plan mode for any large changes
- Agent
  - “Implement this feature…” “Fix the failing tests”
- Edit
  - “Just change this”
  - Limited Context, but you know exactly what needs to change

Instructions

- Set of Markdown files to define your team's/organization's rules
- "Don't use Arrange, Act, Assert comments in tests"
- "Create me an Instructions file" - your AI tool can create this based on the patterns in your codebase
- Tip: anything the agent does wrong? Add it to your Instructions
  - This is key

Instructions

- From

Formatting

- Apply code-formatting style defined in `.editorconfig`.
- Prefer file-scoped namespace declarations and single-line using directives.
- Insert a newline before the opening curly brace of any code block (e.g., after `if`, `for`, `while`, `foreach`, `using`, `try`, etc.).
- Ensure that the final return statement of a method is on its own line.
- Use pattern matching and switch expressions wherever possible.
- Use `nameof` instead of string literals when referring to member names.
- Place private class declarations at the bottom of the file.

Nullable Reference Types

- Declare variables non-nullable, and check for `null` at entry points.
- Always use `is null` or `is not null` instead of `== null` or `!= null`.
- Trust the C# null annotations and don't add null checks when the type system says a value cannot be null.

Building

Always run restore first to set up the local SDK. Run `./restore.sh` (Linux/macOS) or `./restore.cmd` (Windows) first to install the local SDK. After restore, you can use standard `dotnet` commands, which will automatically use the local SDK when available due to the paths configuration in global.json.

Prerequisites

1. **Restore First**: Always run `./restore.sh` (Linux/macOS) or `./restore.cmd` (Windows) to set up the local .NET SDK (~30 seconds)

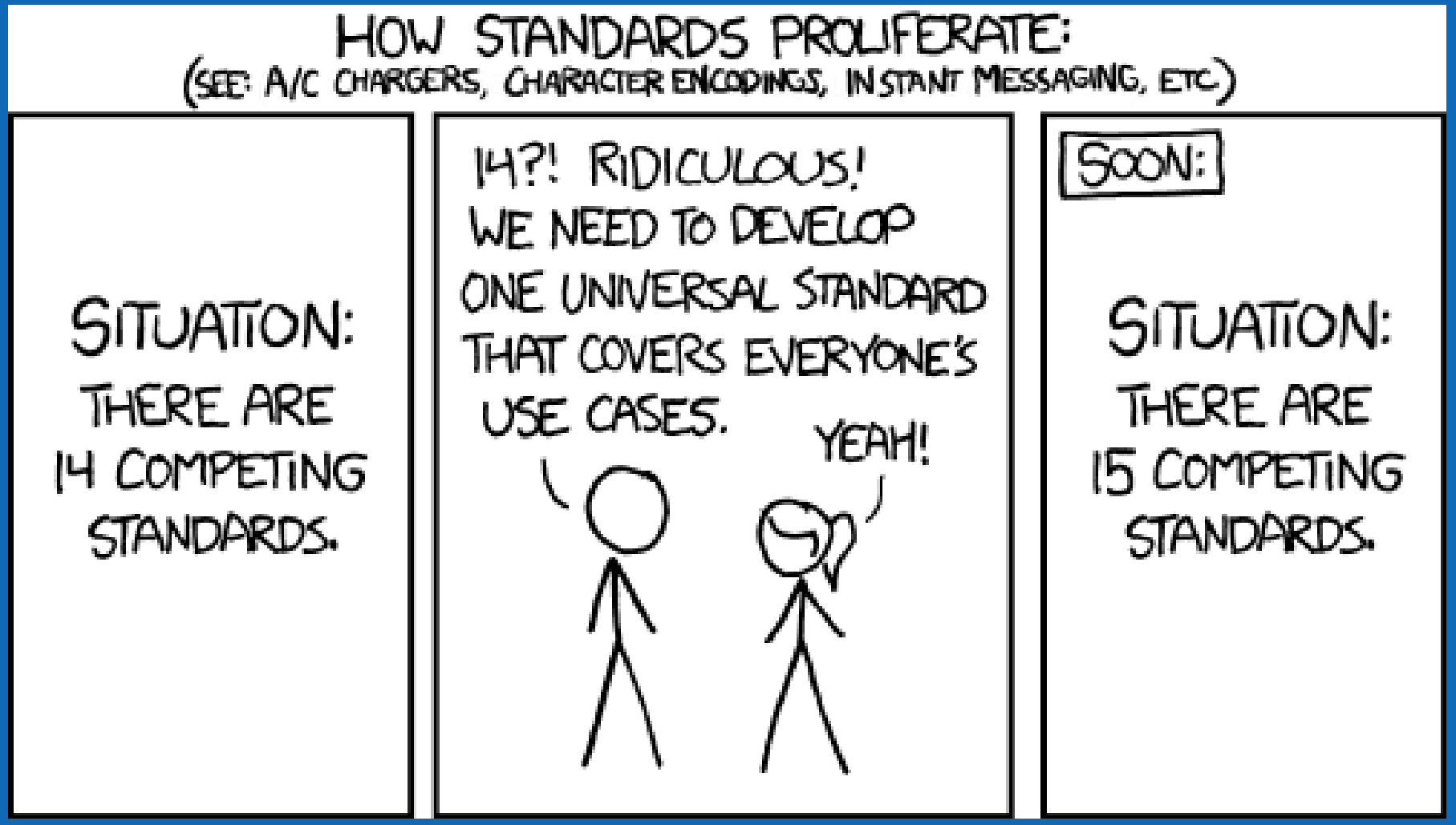
Build Commands

- **Full Build**: `./build.sh` (Linux/macOS) or `./build.cmd` (Windows) - defaults to restore + build (~3-5 minutes)
- **Build Only**: `./build.sh --build` (assumes restore already done)


INSTRUCTIONS
INSTRUCTIONS EVERYWHERE
imgflip.com

Instructions Everywhere

- Every tool now respects AGENTS.md in the root
- .cursorrules, .cursor/rules
- CLAUDE.md
- GEMINI.md
- .github/instructions/copilot-instructions.md


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

Skills

- Rather than loading all those Instructions into context every time, what if we just did it selectively as needed?
- Saves context, saves tokens, saves money
- .github/skills/<skill name>/SKILL.md
- .claude/skills/<skill name>/SKILL.md
  - Recognized by Copilot too
- Just became a thing in October
- https://github.com/anthropics/skills

Skills

skills > frontend-design > SKILL.md

```
---
name: frontend-design
description: Create distinctive, production-grade frontend interfaces with high design quality. Use this skill when the user asks to build web components, pages, artifacts, posters, or applications (examples include websites, landing pages, dashboards, React components, HTML/CSS layouts, or when styling/beautifying any web UI). Generates creative, polished code and UI design that avoids generic AI aesthetics.
license: Complete terms in LICENSE.txt
---

This skill guides creation of distinctive, production-grade frontend interfaces that avoid generic "AI slop" aesthetics. Implement real working code with exceptional attention to aesthetic details and creative choices.

The user provides frontend requirements: a component, page, application, or interface to build. They may include context about the purpose, audience, or technical constraints.

## Design Thinking

Before coding, understand the context and commit to a BOLD aesthetic direction:
- **Purpose**: What problem does this interface solve? Who uses it?
```

In theory Skills should be just as good…

- …but i
- Puttin
  to Ver
- Good
- Maybe

s [according](#)

Configuration	Build	Lint	Test
Baseline	84%	95%	63%
Skill (default behavior)	84%	89%	58%
Skill with explicit instructions	95%	100%	84%
`AGENTS.md`	100%	100%	100%

Copilot Coding Agent (and friends)

- Agents can run in the browser, not just on your machine
- Assign it a task or issue, watch progress (if you want), review the PR
- Create specialized agents for certain tasks – just more Instructions files
- Practical tip: Feature Toggle Remover agent
- This isn’t coming to Azure DevOps

Demo

Code Reviews

- Copilot Code Reviews – you should be doing this
- ~70% of the comments are valid in my experience
- This isn’t coming to Azure DevOps either

Demo

Workflow

- Plan extensively so you can “1 shot”
- Most Plans only take a few minutes to create
- Planning will ask you clarifying questions
- Tip: When it creates “tasks” – ask it to add a validation line on how it validates it’s done with the task

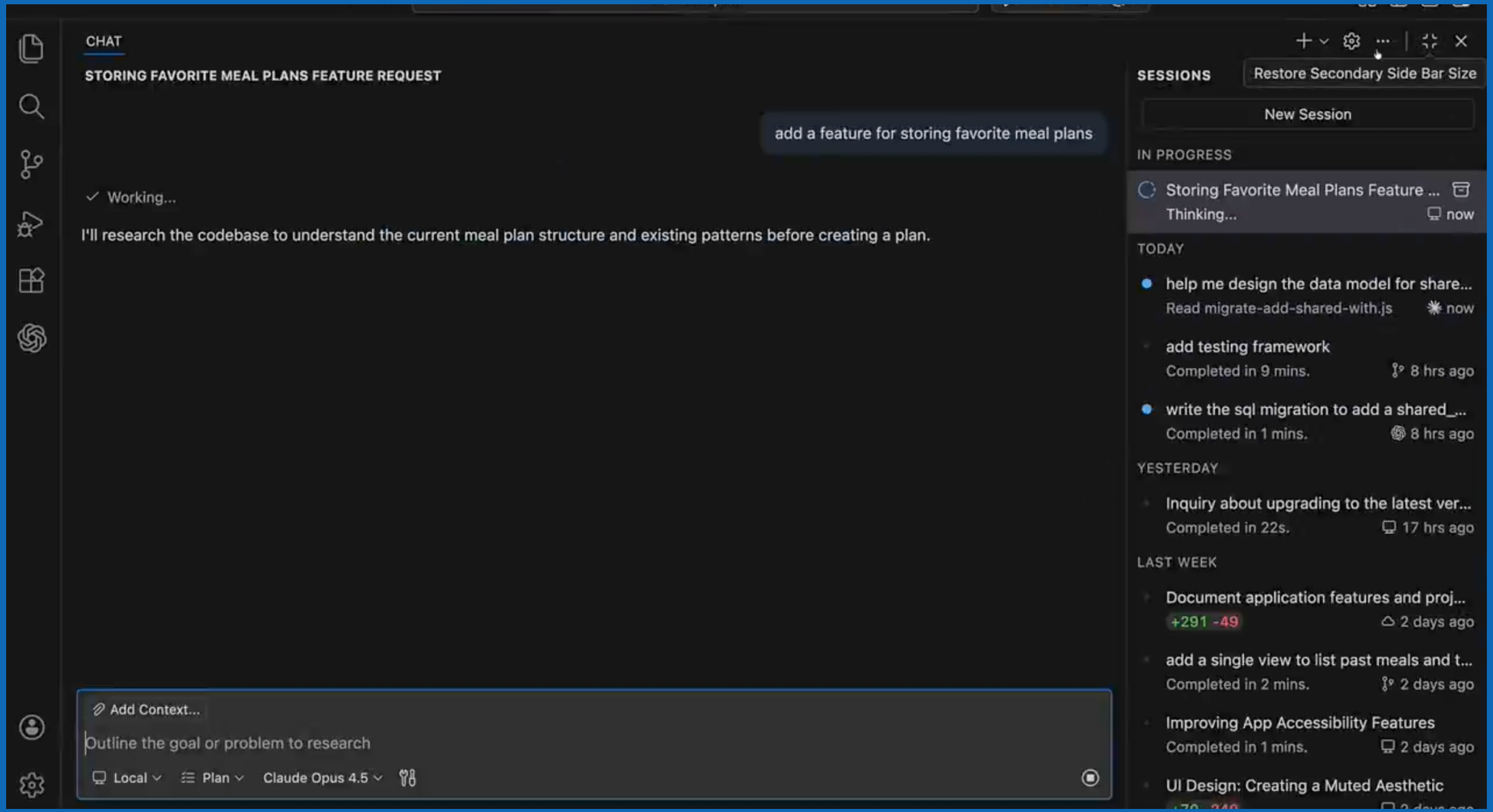
Demo

Autonomous Modes

- Put your agent in a while loop, because agents can be lazy
- Ralph Wiggum
- Copilot Autopilot for CLI – just released last week
- Tight Feedback loops are extremely important – how does AI know it’s done?
- Validation lines I mentioned during Plan Mode
- Compiling, Tests, Linting, Formatting
- These are important without AI!

Where’s it going

- Multi-agent workflows, UI orchestrating CLIs, with Editor built-in
- OpenAI Codex just dropped on Monday that is exactly this
- VS Code did yesterday
- The bottleneck is moving from writing code to reviewing code
- Voice, 120WPM speaking vs 40WPM typing
- The best AI-infused devs I know today spend 80-90% of their day reviewing code, not writing it
- Context window expanding


CHAT
STORING FAVORITE MEAL PLANS FEATURE REQUEST
add a feature for storing favorite meal plans
Working...
I'll research the codebase to understand the current meal plan structure and existing patterns before creating a plan.
Add Context...
Outline the goal or problem to research
Local
Plan
Claude Opus 4.5
SESSIONS
Restore Secondary Side Bar Size
New Session
IN PROGRESS
Storing Favorite Meal Plans Feature ...
Thinking... now
TODAY
help me design the data model for share...
Read migrate-add-shared-with.js now
add testing framework
Completed in 9 mins. 8 hrs ago
write the sql migration to add a shared_...
Completed in 1 mins. 8 hrs ago
YESTERDAY
Inquiry about upgrading to the latest ver...
Completed in 22s. 17 hrs ago
LAST WEEK
Document application features and proj...
+291 -49 2 days ago
add a single view to list past meals and t...
Completed in 2 mins. 2 days ago
Improving App Accessibility Features
Completed in 1 mins. 2 days ago
UI Design: Creating a Muted Aesthetic

Quick Hitters To Use Today

- Start using Instructions
- Any time it does something you don’t want – add it to instructions
- Automatically add Copilot Code Reviews (if you’re on GitHub)
- Start with Plan mode – add validation for when it’s complete
- Use your tools to do more than generate code:
  - Estimation
  - Create stories
  - Understand code
- If unsure what model to use – pick a Claude one, prefer Opus 4.5 (for now) if you don’t mind the premium requests

Things are changing ALL THE TIME

- Just this week
- Monday - OpenAI released Codex for managing multiple agents
- Wednesday - VS Code released support for managing multiple agents
- Anthropic was supposed to release Sonnet 5, but delayed… maybe it released during this talk

How do I stay up to date?

- Follow people on Twitter / LinkedIn
- @bcherny – Boris Cherny who created Claude Code
- @leeerob – Lee Robinson, Cursor
- @code – VS Code
- @claudeai – Claude
- @housecor – Cory House, consultant

Takeaways

- Ideas on where AI is and where it’s going
- Ideas you can start using literally today
- Now’s the time to jump in, because the productivity is real
- “Change has never been this fast but will never be this slow again.”

Resources

- This slide deck at https://scottsauber.com

Questions?

ssauber@leantechniques.com

Add me on LinkedIn:

@scottsauber

@scottsauber.com

Slides at scottsauber.com

Thanks!

Add me on LinkedIn:

Slides at scottsauber.com