Ben Linford

ben@sharedsapience.com

# WORKING WITH AI TO BUILD TOOLS

**http://bit.ly/4cQ8aHk**

# WHY IS THIS IMPORTANT?

**Visualizing the Multi-Layered AI Gap**

# The Multi-Layered AI Gap: From Access to Agency

LAYER 5

## Consumer vs Shaper Gap

Using defaults vs actively choosing, configuring, owning tools. Tiny minority shapes their AI stack.

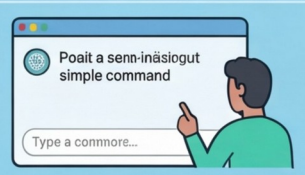

VS


0.05%

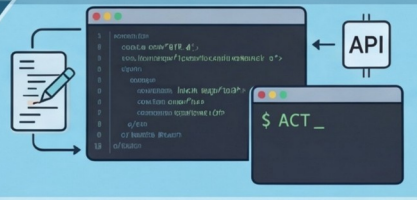
LAYER 4

## Chat vs Builder Gap

Chat window users vs workflow builders. <1% build agents, tools, or automated systms.




Coding





<1%

LAYER 3

## Casual vs Habitual Gap

Tried once vs daily use. Only - 25% use AI multiple times daily. Most stuck at "fun demo" stage.

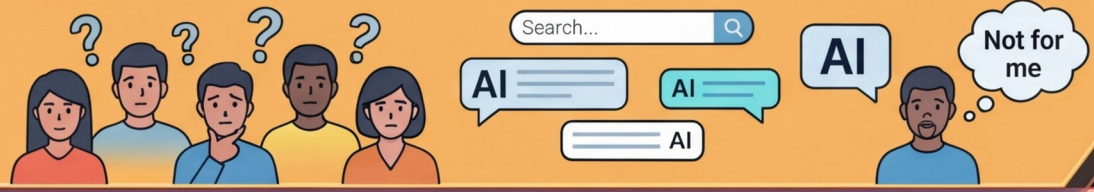

Fun Demo


25%

LAYER 2

## Awareness vs Relevance Gap

Many have heard of AI but don't see relevance to their lives. Majority of non-users say "don't see the need".

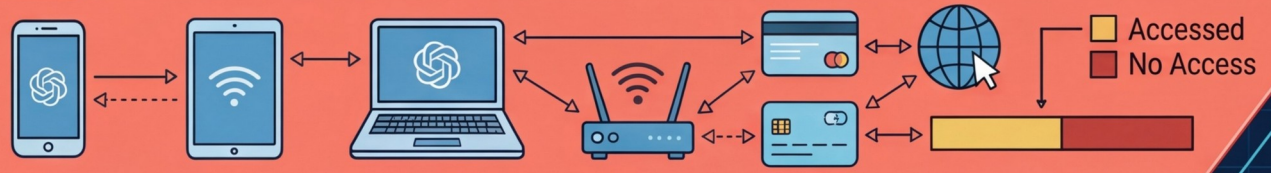



25%

LAYER 1

## Access Gap

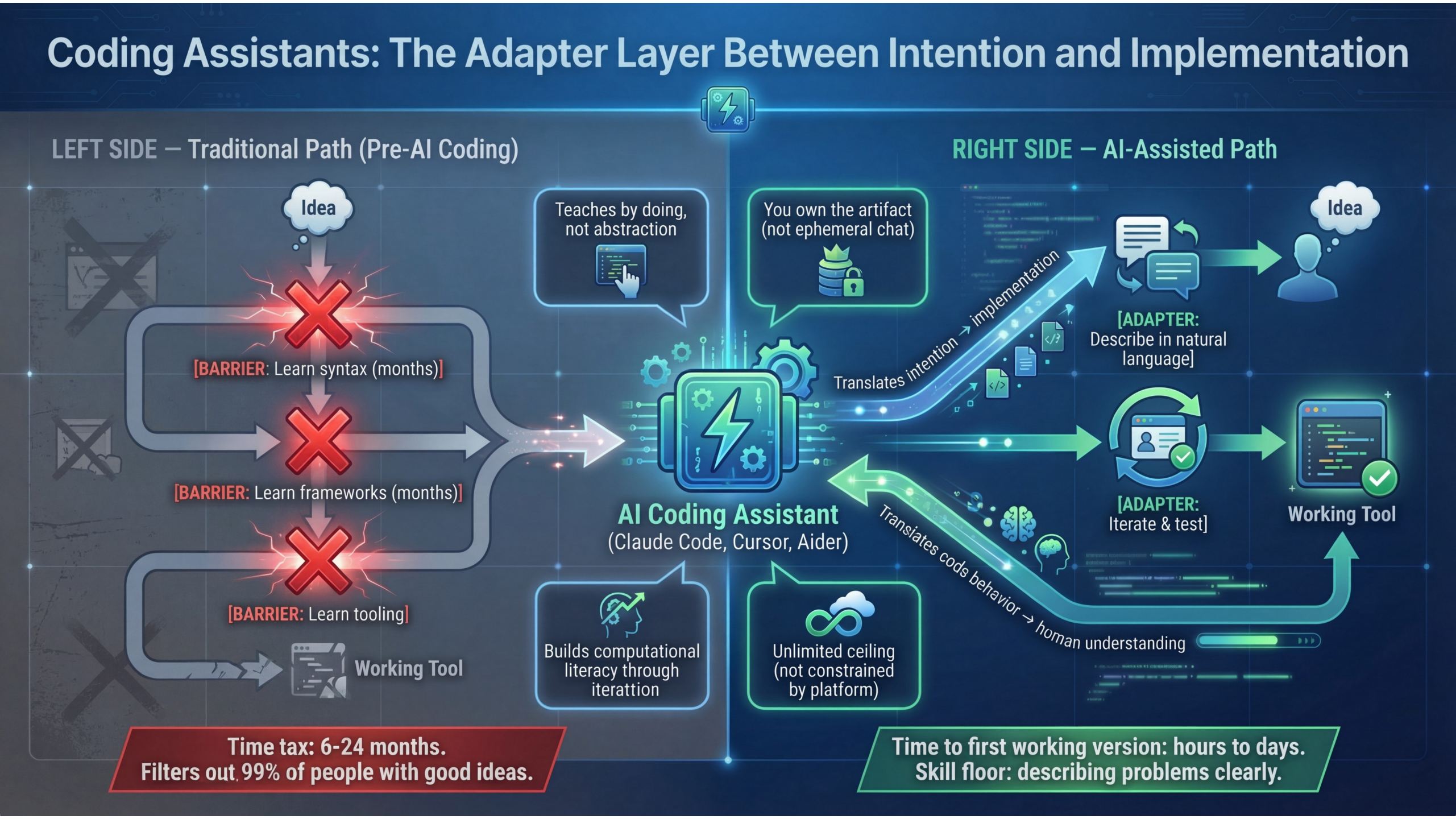
**Who can use AI at all?** ~50% of US adults have tried GenAI, but global access highly uneven.




50%

DECREASING POPULATION SIZE

# Coding Assistants: The Adapter Layer Between Intention and Implementation



# Why Terminals Work Better Than GUIs for AI Agents

Technical comparison of AI agent effectiveness in Terminal/CLI vs GUI environments.

## Terminal/CLI Environment

- ✓ **Deterministic feedback** (exit codes: 0=success, 1+=failure)
- ✓ **Text-based I/O** (AI's native language)
- ✓ **Minimal context** (only shows what's needed)
- ✓ **Composable** (pipe, chain, script commands)
- ✓ **Self-correcting** (AI can verify without human)

```
user@ai-terminal:~$ run_task --verify --deploy
Output: Task initiated. Processing data...
Verification: OK.
Deployment: SUCCESS.
EXIT CODE: 0
>
```

**Success rate on complex tasks:**

**72%**

High reliability for automated tasks.

## AI agents prefer simple, text-based interfaces

## GUI Environment


**Success rate on complex tasks:**

**12%**

Low reliability for automated tasks.

- ✗ **Ambiguous feedback** (AI must interpret visual state)
- ✗ **Pixel-based** (requires parsing visual hierarchies)
- ✗ **Context overload** (entire interface state dumped)
- ✗ **Isolated actions** (not composable)
- ✗ **Stateful and error-prone** (low verification)

**The Translation Gap:** AI builds best in terminals. Humans prefer GUIs. Coding assistants let you work where AI excels (CLI) then translate to where humans want to see results (GUI).

Source: Firecrawl 2026, Terminal Is All You Need (arXiv 2026)

# Closing the AI Gap: A Roadmap Through Coding Literacy

## CURRENT STATE

**AI Builders:**
<1% of population

The Gap

**AI Users:**
~50% of population

## FUTURE STATE

**AI Builders:**
**10-20%** of population

**AI-Literate Population:**
**80%+**

## THE BRIDGE: AI Coding Assistants

### TRACK 1: Access & Infrastructure

- Open-source coding assistants (Aider, Continue)
- Free tier models for education
- Cloud dev environments (reduce hardware barriers)

### TRACK 2: Education & Curriculum

- Teach problem decomposition first, syntax later
- AI coding as core literacy (like spreadsheets in 1980s)
- Build portfolio of working tools, not abstract exercises

### TRACK 3: Skills Development

- Iteration over perfection
- Testing & verification habits
- Computational thinking through practice

## OUTCOME

**OUTCOME:**
**Lower economic barrier to entry**

**OUTCOME:**
**Inverted pedagogy:**

**OUTCOME:**
**Inverted pedagogy: learn by building**

**OUTCOME:**
**Transfer from user to builder to shaper**

## The Compounding Effect

- Once people build one tool that solves their own problem, they:
  - → Gain confidence to build more
  - → See possibilities others miss
  - → Move up the ladder from chat user to builder to shaper
  - → Join the top percentage actively working with AI

# Your Call to Action: Close the Gap by Building

**4. The Result: Empowerment & Agency**

Close the gap in your own work.
Move from consumer to shaper.
Build your own solutions.

**Build Your Future with AI**

"It would be nice if I could make this easier by..."
"I wish this process wasn't so manual..."

**1. Start Where You Are. Identify Your Gaps.**

Pain Points & Inefficiencies

**2. The Pivotal Question**

**"Is this something I could build?"**

Shift from passive user to active problem-solver.

**YES**
**(Become a Builder)**

**No**
**(Remain a User)**

Stuck in Gaps

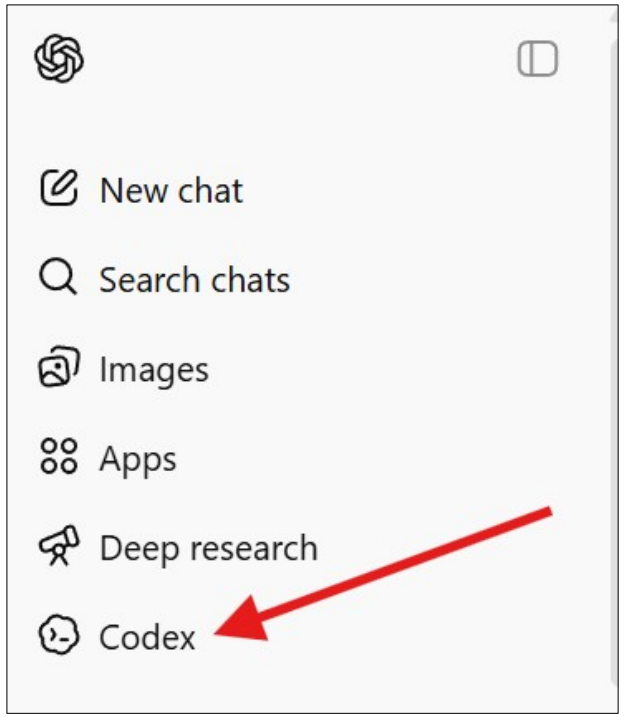
**3. The New Way to Build: Leverage AI Coding Assistants**

Use natural language to translate intention into implementation. Bridge the gap between idea and working tool.

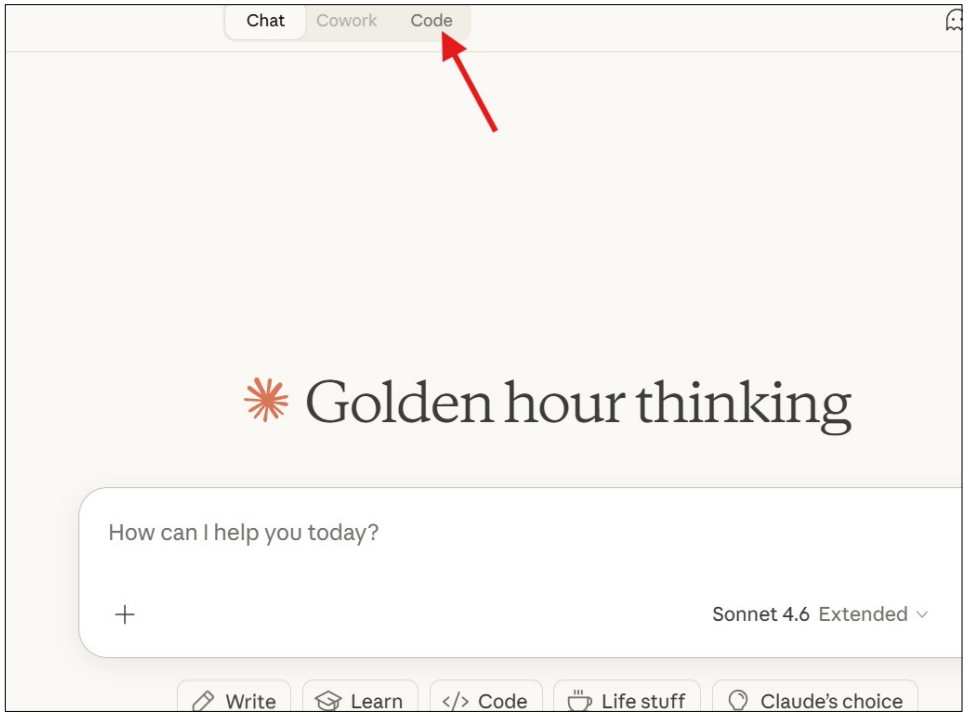

**Don't just use AI. Build with it. Shape your future.**

# TWO OF MANY OPTIONS

- ChatGPT Codex




- Claude Code



# BUILDING WITH AI: FROM IDEA TO APP

## 1. IDENTIFY A NEED


## 2. PROMPT AI SCAFFOLD


## 3. ITERATE WITH AI


## 4. IDENTIFY HOSTING

## 5. VERSION CONTROL



## 1. IDENTIFY A NEED


Simple app solutions (e.g., grading helper, chore tracker)


## 2. PROMPT AI SCAFFOLD


Planning mode, app architecture


## 3. ITERATE WITH AI


UI/navigation/ functionality loop


## 4. IDENTIFY HOSTING


Choose deployment platform


## 5. VERSION CONTROL


(e.g., Github) for code management

**EMPOWER YOURSELF: BUILD TAILORED SOLUTIONS, SAVE SUBSCRIPTIONS**

# CODING ASSISTANT PROMPT BUILDER

Use the QR code or go to:

https://tfp24601.github.io/Coding-Assistant-prompt-builder/

# DEMO

# OUR BUILD REPO

Use the QR code or go to:

https://github.com/tfp24601/emu-summit-build

# CONSULTATION & CONNECTION

- Consultation page, website

https://sharedsapience.com/consultation-personalized-help/

- Substack

https://sharedsapience.substack.com/