2026-01-25T20:14:34Z | Joe Njenga

I Tested Oh My Claude Code The Only Agents Swarm Orchestration You Need

✦

Member-only story

Joe Njenga

Follow

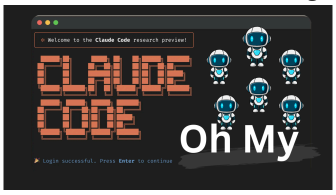
11 min read1 day ago

189

3

...

Press enter or click to view image in full size


Welcome to the Claude Code research preview!
CLAUDE CODE
Oh My
Login successful. Press Enter to continue

Oh My Claude Code makes multi-agent orchestration simple for everyone — even the least experienced Claude Code users.

> **If you've been using Claude Code for a while, you know the process. You type a prompt, get a response, maybe delegate to a subagent manually, and repeat. It works, but it's not efficient when you're building something complex.**

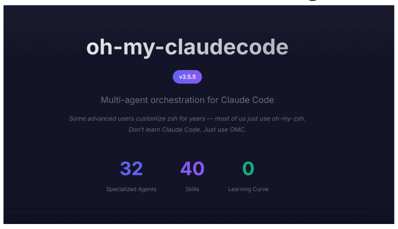
That's where Oh My Claude Code (OMC) comes in. It's like oh-my-zsh for your terminal.

> If you are not a paid Medium member, read the full article here for FREE, but consider joining Medium to help support my work — Thank you!
> **Most developers don't spend years customizing their shell — they install oh-my-zsh and get instant superpowers. OMC does the same thing for Claude Code.**

With 32 specialized agents, 40 skills, and zero learning curve.

OMC transforms Claude Code into a multi-agent orchestration system that handles complex tasks.

Press enter or click to view image in full size


oh-my-claudecode

v3.5.5

Multi-agent orchestration for Claude Code

Some advanced users customize zsh for years — most of us just use oh-my-zsh.
Don't learn Claude Code. Just use OMC.

32 Specialized Agents | 40 Skills | 0 Learning Curve

Here's a quick summary:

- ***5 Execution Modes** — Autopilot, Ultrapilot (3–5x parallel), Swarm, Pipeline, and Ecomode*
- ***32 Specialized Agents** — From architects to designers to data scientists*

- ***Smart Model Routing*** *— Haiku for quick tasks, Sonnet for standard work, Opus for complex reasoning*
- ***Natural Language Control*** *— No commands to memorize, describe what you want*
- ***8.5k+ Downloads/Month*** *— The community is finding value*

> **The tagline says it all: "Don't learn Claude Code. Just use OMC."**

In this article, I'll walk you through how OMC works, test its execution modes, and show you whether it delivers on the promise of making multi-agent orchestration accessible to everyone.

Let's get into it.

Installation & Getting Started

Getting OMC running takes about 30 seconds. There are no complex configurations or dependencies to manage.

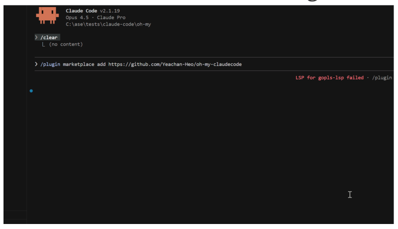
The installation is a 2-step process in Claude Code:

/plugin marketplace add https://github.com/Yeachan-Heo/oh-my-claudecode
/plugin install oh-my-claudecode

- Open your Claude Code terminal
- Add the plugin to your marketplace:

/plugin marketplace add https://github.com/Yeachan-Heo/oh-my-claudecode

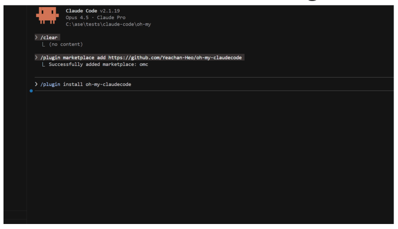
Press enter or click to view image in full size



```
Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  ⎿ (no content)

> /plugin marketplace add https://github.com/Yeachan-Heo/oh-my-claudecode

                                          LSP for gopls-lsp failed · /plugin
```

Install the plugin:

Press enter or click to view image in full size

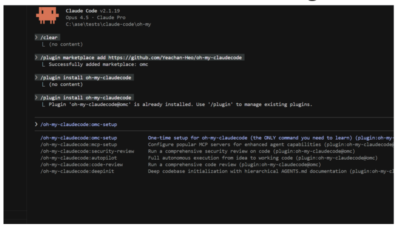


```
Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  ⎿ (no content)

> /plugin marketplace add https://github.com/Yeachan-Heo/oh-my-claudecode
  ⎿ Successfully added marketplace: omc

> /plugin install oh-my-claudecode
```

Once installed, run the setup wizard:

/oh-my-claudecode:omc-setup

Press enter or click to view image in full size



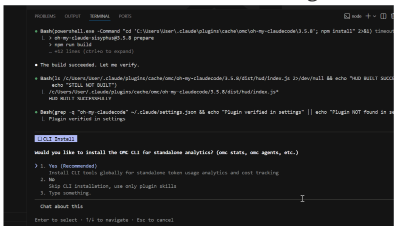
```
Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  ⎿ (no content)

> /plugin marketplace add https://github.com/Yeachan-Heo/oh-my-claudecode
  ⎿ Successfully added marketplace: omc

> /plugin install oh-my-claudecode
  ⎿ (no content)

> /plugin install oh-my-claudecode
  ⎿ Plugin 'oh-my-claudecode@omc' is already installed. Use '/plugin' to manage existing plugins.

> /oh-my-claudecode:omc-setup

  /oh-my-claudecode:omc-setup            One-time setup for oh-my-claudecode (the ONLY command you need to learn) (plugin:oh-my-
  /oh-my-claudecode:mcp-setup            Configure popular MCP servers for enhanced agent capabilities (plugin:oh-my-claudecode@
  /oh-my-claudecode:security-review      Run a comprehensive security review on code (plugin:oh-my-claudecode@omc)
  /oh-my-claudecode:autopilot            Full autonomous execution from idea to working code (plugin:oh-my-claudecode@omc)
  /oh-my-claudecode:code-review          Run a comprehensive code review (plugin:oh-my-claudecode@omc)
  /oh-my-claudecode:deepinit             Deep codebase initialization with hierarchical AGENTS.md documentation (plugin:oh-my-cl
```

When you run the setup wizard, OMC configures several things behind the scenes:

- **32 Specialized Agents** — Each with specific tools and model assignments
- **40 Skills** — From orchestration to git mastery to frontend expertise
- **Delegation Rules** — Automatic routing of tasks to the right agent
- **Model Routing** — Smart selection between Haiku, Sonnet, and Opus
- **Keyword Detection** — Natural language triggers for different modes

The setup creates a `CLAUDE.md` file in your `.claude` directory.

Press enter or click to view image in full size

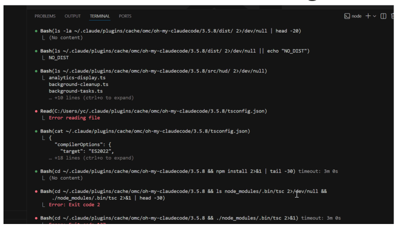


```
● Bash(powershell.exe -Command "cd 'C:\Users\User\.claude\plugins\cache\omc\oh-my-claudecode\3.5.8'; npm install" 2>&1) timeout
  ⎿ > oh-my-claude-sisyphus@3.5.8 prepare
    > npm run build
    … +12 lines (ctrl+o to expand)

● The build succeeded. Let me verify.

● Bash(ls /c/Users/User/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/dist/hud/index.js 2>/dev/null && echo "HUD BUILT SUCCE
      echo "STILL NOT BUILT")
  ⎿ /c/Users/User/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/dist/hud/index.js*
    HUD BUILT SUCCESSFULLY

● Bash(grep -q "oh-my-claudecode" ~/.claude/settings.json && echo "Plugin verified in settings" || echo "Plugin NOT found in se
  ⎿ Plugin verified in settings

☐ CLI Install

Would you like to install the OMC CLI for standalone analytics? (omc stats, omc agents, etc.)

❯ 1. Yes (Recommended)
     Install CLI tools globally for standalone token usage analytics and cost tracking
  2. No
     Skip CLI installation, use only plugin skills
  3. Type something.

  Chat about this

Enter to select · ↑/↓ to navigate · Esc to cancel
```

> **This file contains all the orchestration rules that transform Claude Code into a multi-agent system.**

During my installation, I ran into several npm build errors, but I allowed Claude Code to sort out the dependency errors one at a time.

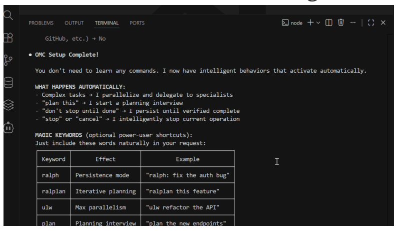
Press enter or click to view image in full size



```
● Bash(ls -la ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/dist/ 2>/dev/null | head -20)
  ⎿ (No content)

● Bash(ls ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/dist/ 2>/dev/null || echo "NO_DIST")
  ⎿ NO_DIST

● Bash(ls ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/src/hud/ 2>/dev/null)
  ⎿ analytics-display.ts
    background-cleanup.ts
    background-tasks.ts
    … +10 lines (ctrl+o to expand)

● Read(C:/Users/yc/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/tsconfig.json)
  ⎿ Error reading file

● Bash(cat ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8/tsconfig.json)
  ⎿ {
      "compilerOptions": {
        "target": "ES2022",
    … +18 lines (ctrl+o to expand)

● Bash(cd ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8 && npm install 2>&1 | tail -30) timeout: 3m 0s
  ⎿ (No content)

● Bash(cd ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8 && ls node_modules/.bin/tsc 2>/dev/null &&
      ./node_modules/.bin/tsc 2>&1 | head -30)
  ⎿ Error: Exit code 2

● Bash(cd ~/.claude/plugins/cache/omc/oh-my-claudecode/3.5.8 && ./node_modules/.bin/tsc 2>&1) timeout: 3m 0s
```

> **Finally, when the installation is complete, you should see this message that confirms successful installation.**

Press enter or click to view image in full size



```
   GitHub, etc.) → No

● OMC Setup Complete!

  You don't need to learn any commands. I now have intelligent behaviors that activate automatically.

  WHAT HAPPENS AUTOMATICALLY:
  - Complex tasks → I parallelize and delegate to specialists
  - "plan this" → I start a planning interview
  - "don't stop until done" → I persist until verified complete
  - "stop" or "cancel" → I intelligently stop current operation

  MAGIC KEYWORDS (optional power-user shortcuts):
  Just include these words naturally in your request:
  ┌─────────┬────────────────────┬──────────────────────────────┐
  │ Keyword │       Effect       │           Example            │
  ├─────────┼────────────────────┼──────────────────────────────┤
  │ ralph   │ Persistence mode   │ "ralph: fix the auth bug"    │
  ├─────────┼────────────────────┼──────────────────────────────┤
  │ ralplan │ Iterative planning │ "ralplan this feature"       │
  ├─────────┼────────────────────┼──────────────────────────────┤
  │ ulw     │ Max parallelism    │ "ulw refactor the API"       │
  ├─────────┼────────────────────┼──────────────────────────────┤
  │ plan    │ Planning interview │ "plan the new endpoints"     │
```

● OMC Setup Complete!

You don

WHAT HAPPENS AUTOMATICALLY:
- Complex tasks → I parallelize and delegate to specialists
- "plan this" → I start a planning interview
- "don't stop until done" → I persist until verified complete
- "stop" or "cancel" → I intelligently stop current operation

MAGIC KEYWORDS (optional power-user shortcuts):
Just include these words naturally in your request:

Keyword	Effect	Example
ralph	Persistence mode	"ralph: fix the auth bug"
ralplan	Iterative planning	"ralplan this feature"
ulw	Max parallelism	"ulw refactor the API"
plan	Planning interview	"plan the new endpoints"

Combine them: "ralph ulw: migrate the database"

HUD STATUSLINE:
The status bar now shows OMC state. Restart Claude Code to see it.

CLI ANALYTICS:
- omc-cli or oh-my-claudecode - View token usage and costs
- omc-analytics - Analytics commands

(You may need to restart your terminal for the commands to be in PATH)

That

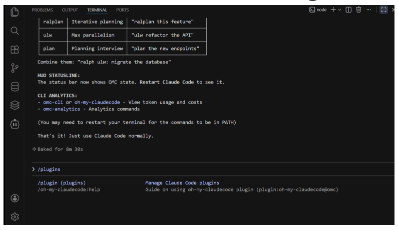
✻ Baked for 8m 30s

> **Note that setup took Claude Code 8 minutes.**

Verify the plugin is enabled:

/plugin list

Press enter or click to view image in full size



```
 | ralplan | Iterative planning | "ralplan this feature"   |
 | ulw     | Max parallelism    | "ulw refactor the API"   |
 | plan    | Planning interview | "plan the new endpoints" |

 Combine them: "ralph ulw: migrate the database"

 HUD STATUSLINE:
 The status bar now shows OMC state. Restart Claude Code to see it.

 CLI ANALYTICS:
 - omc-cli or oh-my-claudecode - View token usage and costs
 - omc-analytics - Analytics commands

 (You may need to restart your terminal for the commands to be in PATH)

 That's it! Just use Claude Code normally.

✻ Baked for 8m 30s

> /plugins

 /plugin (plugins)              Manage Claude Code plugins
 /oh-my-claudecode:help         Guide on using oh-my-claudecode plugin (plugin:oh-my-claudecode@omc)
```

Project vs Global Configuration

You have two options for configuration scope:

Project-scoped (recommended):

/oh-my-claudecode:omc-setup --local

Creates `.claude/CLAUDE.md` in your current project only.

Global:

/oh-my-claudecode:omc-setup

Creates `~/.claude/CLAUDE.md` for all Claude Code sessions.

If both exist, project config takes precedence over global.

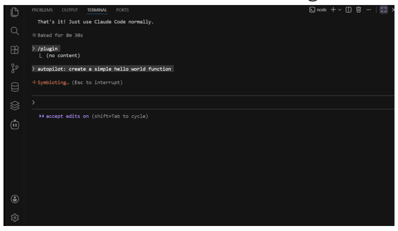
Quick Verification

After setup, test that everything works:

autopilot: create a simple hello world function

If OMC is configured correctly, you'll see Claude announce: "I'm activating autopilot for full autonomous execution."

Press enter or click to view image in full size

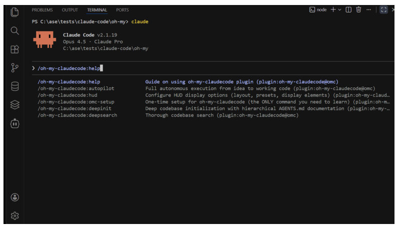


```
 That's it! Just use Claude Code normally.

✻ Baked for 8m 30s

> /plugin
  └ (no content)

> autopilot: create a simple hello world function

✢ Symbioting… (Esc to interrupt)

>

  ▸▸ accept edits on (shift+Tab to cycle)
```

Verify agents are available by running:

/oh-my-claudecode:help

Press enter or click to view image in full size



```
PS C:\ase\tests\claude-code\oh-my> claude

        Claude Code v2.1.19
        Opus 4.5 · Claude Pro
        C:\ase\tests\claude-code\oh-my

> /oh-my-claudecode:help

  /oh-my-claudecode:help           Guide on using oh-my-claudecode plugin (plugin:oh-my-claudecode@omc)
  /oh-my-claudecode:autopilot      Full autonomous execution from idea to working code (plugin:oh-my-claudecode@omc)
  /oh-my-claudecode:hud            Configure HUD display options (layout, presets, display elements) (plugin:oh-my-claud…
  /oh-my-claudecode:omc-setup      One-time setup for oh-my-claudecode (the ONLY command you need to learn) (plugin:oh-m…
  /oh-my-claudecode:deepinit       Deep codebase initialization with hierarchical AGENTS.md documentation (plugin:oh-my-…
  /oh-my-claudecode:deepsearch     Thorough codebase search (plugin:oh-my-claudecode@omc)
```

/oh-my-claudecode:help

● oh-my-claudecode Help

You don't need to learn any commands! OMC enhances Claude Code with intelligent behaviors that activate automatically.
What Happens Automatically

When You...	I Automatically...
Give me a complex task	Parallelize and delegate to specialist agents
Ask me to plan something	Start a planning interview
Need something done completely	Persist until verified complete

Work on UI/frontend	Activate design sensibility
Say "stop" or "cancel"	Intelligently stop current operation

Magic Keywords (Optional Shortcuts)

Include these words naturally in your request:

Keyword	Effect	Example
autopilot	Full autonomous execution	"autopilot: build a todo app"
ralph	Persistence mode	"ralph: fix all the bugs"
ulw	Max parallelism	"ulw refactor the API"
eco	Token-efficient mode	"eco fix all errors"
plan	Planning interview	"plan the new endpoints"
ralplan	Iterative planning	"ralplan this feature"

Combine them: "ralph ulw: migrate the database"

Stopping Things

Just say "stop", "cancel", or "abort" - I'll figure out what to stop.

First Time Setup

/oh-my-claudecode:omc-setup

This is the only command you need to know.

Available Skills

Run /oh-my-claudecode:<skill> for any of these:
- autopilot - Full autonomous execution
- ultrawork - Maximum parallel execution
- ecomode - Token-efficient parallelism
- ultrapilot - Parallel autopilot (3-5x faster)
- swarm - N coordinated agents
- pipeline - Sequential agent chaining
- plan - Strategic planning interview
- ralph - Persistence until complete
- analyze - Deep investigation
- deepsearch - Thorough codebase search
- research - Parallel scientist orchestration
- tdd - Test-driven development
- code-review - Comprehensive code review
- security-review - Security vulnerability scan
- build-fix - Fix build/type errors
- doctor - Diagnose installation issues
- hud - Configure statusline display

Version: 3.4.0 | https://github.com/Yeachan-Heo/oh-my-claudecode

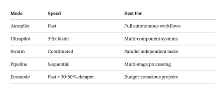
5 Execution Modes

OMC offers five distinct execution modes.

> **Each is designed for different scenarios — from fully autonomous builds to budget-conscious token optimization.**

Here's the breakdown:

Press enter or click to view image in full size



Mode	Speed	Best For
Autopilot	Fast	Full autonomous workflows
Ultrapilot	3-5x faster	Multi-component systems
Swarm	Coordinated	Parallel independent tasks
Pipeline	Sequential	Multi-stage processing
Ecomode	Fast + 30-50% cheaper	Budget-conscious projects

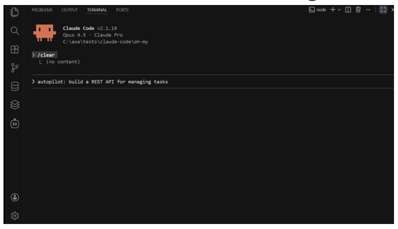
Let me walk through each one.

1. Autopilot — Full Autonomous Execution

Autopilot is the flagship mode. You describe what you want, and OMC handles everything from planning to implementation to testing.

autopilot: build a REST API for managing tasks

Press enter or click to view image in full size



```
Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  L (no content)

> autopilot: build a REST API for managing tasks
```

What happens:

- *Automatic planning and requirements gathering*
- *Parallel execution with specialized agents*
- *Continuous verification and testing*
- *Self-correction until completion*

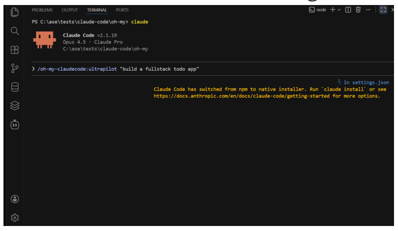
> **This mode combines the best of ralph (persistence), ultrawork (parallelism), and plan (strategic thinking) into one streamlined experience.**

2. Ultrapilot — Parallel Autopilot (3–5x Faster)

Ultrapilot takes autopilot and supercharges it with up to 5 concurrent workers.

/oh-my-claudecode:ultrapilot "build a fullstack todo app"

Press enter or click to view image in full size



```
PS C:\ase\tests\claude-code\oh-my> claude

Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /oh-my-claudecode:ultrapilot "build a fullstack todo app"

                                                          In settings.json
         Claude Code has switched from npm to native installer. Run `claude install` or see
         https://docs.anthropic.com/en/docs/claude-code/getting-started for more options.
```

Key features:

- *Automatic task decomposition into parallelizable subtasks*
- *File ownership coordination to prevent conflicts*
- *3–5x speedup on multi-component systems*

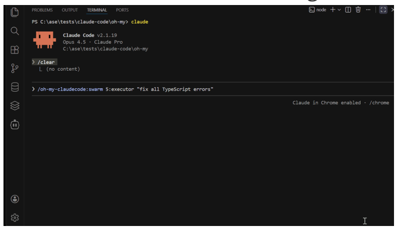
> **Best for large refactoring jobs or building apps with multiple components.**

3. Swarm — Coordinated Agent Teams

Swarm spawns N agents that work from a shared task pool. Each agent claims atomic tasks, executes them, and marks completion.

/oh-my-claudecode:swarm 5:executor "fix all TypeScript errors"

Press enter or click to view image in full size



```
PS C:\ase\tests\claude-code\oh-my> claude

Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  L (no content)

> /oh-my-claudecode:swarm 5:executor "fix all TypeScript errors"

                                             Claude in Chrome enabled · /chrome
```

Key features:

- *Atomic task claiming prevents duplicate work*
- *5-minute timeout per task with auto-release*
- *Scales from 2 to 10 workers*
- *Clean completion when all tasks are done*

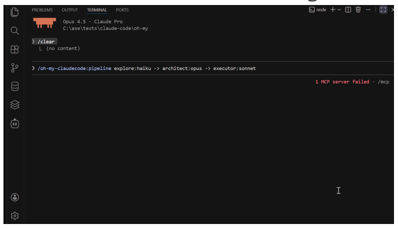
> **Think of it like a team of developers picking tasks from a sprint board.**

4. Pipeline — Sequential Agent Chaining

Pipeline chains agents in sequence, with data passing between stages. Perfect for workflows that need a specific order.

/oh-my-claudecode:pipeline explore:haiku -> architect:opus -> executor:sonnet

Press enter or click to view image in full size



```
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  L (no content)

> /oh-my-claudecode:pipeline explore:haiku -> architect:opus -> executor:sonnet
                                                   1 MCP server failed · /mcp
```

Built-in presets:

- ***review*** *— explore → architect → critic → executor*
- ***implement*** *— planner → executor → tdd-guide*
- ***debug*** *— explore → architect → build-fixer*
- ***refactor*** *— explore → architect-medium → executor-high → qa-tester*

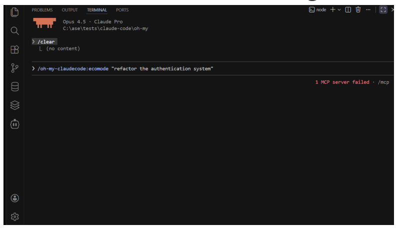
> **You can also create custom chains for your specific workflow.**

5. Ecomode — Token-Efficient Parallelism

Ecomode gives you parallel execution with 30–50% token savings. It uses smart model routing to minimize costs.

/oh-my-claudecode:ecomode "refactor the authentication system"

Press enter or click to view image in full size



```
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  L (no content)

> /oh-my-claudecode:ecomode "refactor the authentication system"
                                                   1 MCP server failed · /mcp
```

How it saves tokens:

- *Simple tasks → Haiku (ultra-cheap)*
- *Standard work → Sonnet (balanced)*
- *Complex reasoning → Opus (only when needed)*

> **Perfect for Pro plan users who want speed without burning through their quota.**

Magic Keywords & Smart Delegation

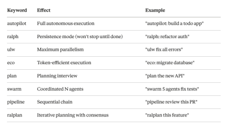
One of OMC's best features is that you don't need to memorize commands.

> **It detects intent from natural language and activates the right behaviors automatically.**

But for power users who want explicit control, magic keywords provide precision.

Magic Keywords Reference

Press enter or click to view image in full size



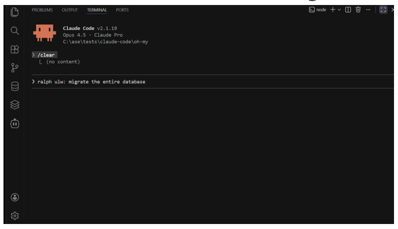
Keyword	Effect	Example
autopilot	Full autonomous execution	"autopilot: build a todo app"
ralph	Persistence mode (won't stop until done)	"ralph: refactor auth"
ulw	Maximum parallelism	"ulw fix all errors"
eco	Token-efficient execution	"eco: migrate database"
plan	Planning interview	"plan the new API"
swarm	Coordinated N agents	"swarm 5 agents fix tests"
pipeline	Sequential chain	"pipeline review this PR"
ralplan	Iterative planning with consensus	"ralplan this feature"

You can combine keywords for superpowers:

ralph ulw: migrate the entire database

This activates persistence (won't stop) + ultrawork (maximum parallelism).

Press enter or click to view image in full size

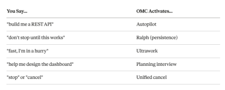


```
Claude Code v2.1.19
Opus 4.5 · Claude Pro
C:\ase\tests\claude-code\oh-my

> /clear
  L (no content)

> ralph ulw: migrate the entire database
```

Natural Language Detection

Even without keywords, OMC understands intent:

Press enter or click to view image in full size

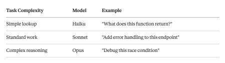


You Say...	OMC Activates...
"build me a REST API"	Autopilot
"don't stop until this works"	Ralph (persistence)
"fast, I'm in a hurry"	Ultrawork
"help me design the dashboard"	Planning interview
"stop" or "cancel"	Unified cancel

Smart Model Routing

OMC selects the right model based on task complexity:

Press enter or click to view image in full size



Task Complexity	Model	Example
Simple lookup	Haiku	"What does this function return?"
Standard work	Sonnet	"Add error handling to this endpoint"
Complex reasoning	Opus	"Debug this race condition"

> **This saves tokens while ensuring quality. Simple tasks don't waste Opus credits, and complex tasks get the reasoning power they need.**

32 Specialized Agents

OMC doesn't just route by complexity, but it routes by domain.

Here's the agent roster organized by specialty:

Analysis & Architecture

- *architect (Opus) — Complex debugging, system design*
- *architect-medium (Sonnet) — Standard analysis*
- *architect-low (Haiku) — Quick questions*

Execution

- *executor-high (Opus) — Complex refactoring*
- *executor (Sonnet) — Feature implementation*
- *executor-low (Haiku) — Simple code changes*

Search & Exploration

- *explore-high (Opus) — Architectural search*
- *explore-medium (Sonnet) — Thorough search*
- *explore (Haiku) — Quick code lookup*

Frontend & Design

- *designer-high (Opus) — Complex UI systems*
- *designer (Sonnet) — UI components*
- *designer-low (Haiku) — Simple styling*

Other Specialists

- *planner (Opus) — Strategic planning*
- *critic (Opus) — Plan review*
- *researcher (Sonnet) — Documentation lookup*
- *writer (Haiku) — Technical docs*
- *qa-tester (Sonnet) — CLI testing*
- *scientist (Sonnet) — Data analysis*
- *vision (Sonnet) — Image analysis*

> **When you say "fix the UI colors," OMC delegates to the designer agent. When you say "debug this crash," it goes to the architect.**

You don't manage this routing — it happens automatically.

Final Thoughts

After testing Oh My Claude Code across different modes and scenarios, here's my take.

- **Zero learning curve is real** — I didn't need to read documentation to get started. Install, set up, and you're orchestrating multi-agent workflows immediately.
- **The autopilot mode is impressive** — For building small apps or features, it handles the entire workflow from planning to implementation to verification.
- **Smart model routing saves tokens** — Instead of always using Opus, OMC routes simple tasks to Haiku and standard work to Sonnet. This adds up to 30–50% savings on complex projects.
- **Natural language** — Saying "don't stop until done" triggers Ralph mode. Saying "fast" triggers parallelism. You don't need to memorize keywords
- **Agent specialization makes sense** — Having dedicated agents for frontend, architecture, testing, and documentation means each task gets handled by the right specialist with the right model.

> **OMC has its own philosophy about how Claude Code should work. If you prefer manual control over everything, this might feel like too much automation.**

If you're using Claude Code for anything beyond simple one-off prompts, OMC is worth trying.

> **Have you tried Oh My Claude Code, and what was your experience? Let me know in the comments below.**

Resources:

- GitHub: https://github.com/Yeachan-Heo/oh-my-claudecode
- Website: https://yeachan-heo.github.io/oh-my-claudecode-website/
- Documentation: Check `/oh-my-claudecode:help` after installation

Claude Code Masterclass Course

Press enter or click to view image in full size

Every day, I'm working hard to build the ultimate Claude Code course, which demonstrates how to create workflows that coordinate multiple agents for complex development tasks. It's due for release soon.

It will take what you have learned from this article to the next level of complete automation.

New features are added to Claude Code daily, and keeping up is tough.

The course explores Agents, Hooks, advanced workflows, and productivity techniques that many developers may not be aware of.

Once you join, you'll receive all the updates as new features are rolled out.

This course will cover:

- *Advanced subagent patterns and workflows*
- *Production-ready hook configurations*
- *MCP server integrations for external tools*
- *Team collaboration strategies*
- *Enterprise deployment patterns*
- *Real-world case studies from my consulting work*

If you're interested in getting notified when the Claude Code course launches, **[click here to join the early access list →](#)**

(*Currently, I have 3000+ already signed-up developers)*

> ***I'll share exclusive previews, early access pricing, and bonus materials with people on the list.***

Let's Connect!

If you are new to my content, my name is Joe Njenga

> *Join thousands of other software engineers, AI engineers, and solopreneurs who read my content* ***daily on Medium and on YouTube where I review the latest AI engineering tools and trends****. If you are more curious about my projects and want to receive detailed guides and tutorials,* ***join thousands of other AI enthusiasts in my weekly AI Software engineer newsletter***

If you would like to connect directly, you can reach out here:

AI Integration Software Engineer (10+ Years Experience)

Software Engineer specializing in AI integration and automation. Expert in building AI agents, MCP servers, RAG...

njengah.com

Follow me on Medium | YouTube Channel | X | LinkedIn