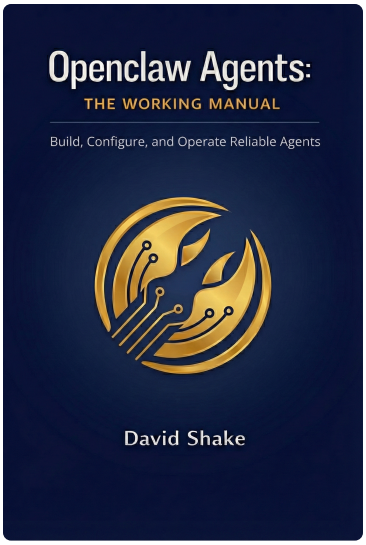
A FREE GUIDE FROM



Build, Configure, and Operate Reliable Agents

This guide is a free excerpt from the full manual, which covers everything you need to run OpenClaw agents reliably — architecture, channels, security, models, memory, automation, and six complete setup guides.

GET THE FULL BOOK

davidshake.com/openclaw-manual

Guide 4: Personalizing Soul & Persistent Memory

Out of the box, your OpenClaw agent works. It answers questions, follows instructions, and generally does what you ask. But it also sounds like every other AI assistant on the planet – polite, eager, and completely generic. Like a new hire who smiles a lot but doesn't know your name yet.

This guide fixes that. We're going to turn a default agent into *your* agent – one that talks the way you want, remembers what matters to you, and handles recurring situations without being told twice. The tools are simple: a handful of plain text files and a memory system built on Markdown. The difference they make is enormous.

> **Warning: A Moving Target**
>
> The workspace file format and memory system in OpenClaw are actively evolving. The general principles in this guide – be specific, be concise, curate regularly – will remain sound regardless of implementation changes. But specific file paths, configuration keys, or memory behaviors may shift between versions. If something doesn't match what you see, check davidshake.com/openclaw-manual/updates for the latest.

The Problem: Why Default Agents Feel Robotic

Every AI model ships with a built-in persona – some variation of "I'm a helpful, harmless assistant." This is fine for one-off questions, but it's terrible for a personal assistant you interact with daily. The result is an agent that uses the same tone for everything, adds unnecessary disclaimers to every answer, doesn't know your preferences, and responds with the enthusiasm of a customer service bot reading from a script.

The fix isn't complicated, but it requires you to be *specific*. The single biggest mistake people make is being too vague. Writing "be helpful and friendly" in your SOUL.md is like telling a new employee to "do a good job" – technically correct, completely useless. The AI already defaults to helpful and friendly. You haven't told it anything new.

What actually changes behavior is concrete instruction. And that's what this guide teaches you to write.

Understanding the Workspace Files

Your agent's personality, instructions, and knowledge about you live in the **workspace** – a folder at `~/.openclaw/workspace` containing plain text Markdown files. If you haven't already, read **Chapter 9: The Workspace** for the full overview. Here's the quick version of the files we'll focus on:

File	Purpose	Analogy
SOUL.md	Personality, tone, boundaries	A character sheet in a role-playing game
AGENTS.md	Workflows, triggers, standing orders	An employee training manual
USER.md	Your profile – who the agent is talking to	A contact card the agent keeps on its desk
IDENTITY.md	The agent's name and emoji	A nameplate
MEMORY.md	Durable long-term facts	A reference card
memory/ folder	Daily conversation logs	A daily diary

Every one of these files is plain Markdown. Open them in any text editor, Obsidian, VS Code – whatever you like. Edit them directly. Changes take effect on the next conversation.

> **Info: Where Are These Files?**
>
> All workspace files live at `~/.openclaw/workspace/`. The `~` is shorthand for your home folder. On macOS, that's `/Users/yourname/`. On Linux, it's `/home/yourname/`. You can navigate there in Terminal with `cd ~/.openclaw/workspace` or open the folder in your file manager.

Writing an Effective SOUL.md

SOUL.md is the single most impactful file in the workspace. It defines *how* your agent communicates – its personality, its style, its limits. Think of it as the difference between talking to a generic customer service chatbot and talking to a colleague who knows your communication style.

Rule 1: Be Specific, Not Generic

The most common mistake is writing instructions that sound meaningful but say nothing the model doesn't already do. Compare:

```
<!-- Bad: vague, tells the model what it already does -->
## Communication Style
- Be helpful and friendly
- Provide clear answers
- Be respectful
```

```
<!-- Good: specific, changes actual behavior -->
## Communication Style
- Keep responses under 3 paragraphs unless I explicitly ask for detail
- Use dry humor sparingly -- think deadpan, not dad jokes
- When I ask a technical question, lead with the answer, then explain if needed
- Never use phrases like "Great question!" or "I'd be happy to help!"
- If you're uncertain, say "I'm not sure" plainly -- don't hedge with five qualifiers
```

The bad version is a set of platitudes. The good version is a set of *constraints* that actually shape output. Notice how much of the good version is about what *not* to do – that's not an accident.

Rule 2: Define What the Agent Should NOT Do

Negative instructions are often more powerful than positive ones. AI models have strong default behaviors – excessive politeness, over-qualification, lengthy preambles. You can't override these with vague positive instructions like "be concise." You override them by naming the specific behavior you don't want.

```
## Don'ts
- Don't start responses with "Sure!" or "Absolutely!" or "Of course!"
- Don't add safety disclaimers unless I'm asking about something genuinely dangerous
- Don't repeat my question back to me before answering
- Don't offer to "help with anything else" at the end of every response
- Don't use emoji unless I use them first
- Don't apologize unless you've actually made a mistake
```

This might feel blunt, but the agent doesn't have feelings to hurt. What it does have is a strong tendency toward certain behaviors, and the clearest way to redirect those tendencies is to name them explicitly.

Rule 3: Set Communication Defaults

Think about the small decisions that come up in every response: how long should it be? How formal? Should it use bullet points or prose? What should it do when it doesn't know something? Setting defaults for these saves you from constantly correcting the agent.

```
## Defaults
- Default response length: short (2-4 sentences). I'll ask for more if I want it.
- Default format: plain text, not Markdown, unless the content genuinely benefits from formatting
- When I ask "what do you think?" I want your actual assessment, not a list of pros and cons
- When multiple approaches exist, recommend one and explain why -- don't give me a menu
- Timezone: US Eastern. When referencing times, use ET unless I specify otherwise.
```

Rule 4: Give It a Voice

This is where SOUL.md gets fun. You're not just configuring a tool – you're shaping a personality. The agent can be warm, terse, academic, irreverent, or anything in between. Give it texture.

```
## Voice
- You're calm and direct. Think of a senior engineer explaining something at a whiteboard,
  not a sales rep pitching a product.
- You can reference movies, books, and history when an analogy fits naturally.
  Don't force it.
- Swearing is fine in casual conversation. Match my register.
- When I'm clearly frustrated, acknowledge it briefly and move to the solution.
  Don't therapize.
```

> **Tip: Start Small, Iterate Often**

> Don't try to write the perfect SOUL.md on your first attempt. Start with 10-15 lines covering the things that bother you most about default AI behavior. Use the agent for a few days, notice what annoys you, and add a line addressing it. The best SOUL.md files are grown, not written in one sitting.

Writing AGENTS.md for Workflow Automation

If SOUL.md is *who* the agent is, AGENTS.md is *how it works*. This is where you define standing instructions, trigger phrases, and recurring workflows. Think of it as training a new assistant: "When this happens, do that."

Trigger Phrases

The simplest pattern is a keyword that activates a specific workflow:

```
## Trigger Phrases

### "standup"
When I say "standup" or "morning report":
1. Check my calendar for today's events
2. Summarize any unread messages from overnight
3. List my open tasks, sorted by priority
4. Ask what I'm planning to focus on today

### "end of day"
When I say "end of day" or "eod":
1. Summarize what we accomplished today
2. List anything unresolved or carried over
3. Write a brief daily log entry to memory

### "draft"
When I say "draft [topic]":
1. Write a first draft -- rough is fine, don't over-polish
2. Keep it to one page unless I specify length
3. Use my usual writing voice (reference SOUL.md)
4. Present it in a code block so I can copy it easily
```

Standing Instructions and Tool Preferences

Beyond triggers, you can set instructions for categories of interaction, and specify tool defaults:

```
## Standing Instructions

### Code reviews
When I share code for review:
- Focus on logic errors and maintainability, not style nitpicks
- If the code is fine, just say so. Don't invent suggestions.

### Research
When I ask you to look into something:
- Start with a one-paragraph summary, then bullet points for key findings
- Don't pad with filler if there isn't much to say

## Tool Preferences
- For web searches, prefer concise queries
- For file operations, confirm before deleting anything
- Default to ripgrep over grep when searching files
```

Filling Out USER.md So the Agent Remembers You

USER.md is the agent's cheat sheet about you. The better this file is, the less you'll repeat yourself. The bootstrapping process (the initial setup conversation) creates a starting version, but you should expand it over time.

What to Include

Think about what a good human assistant would naturally learn in the first few weeks of working with you:

```
# User Profile

## Basics
- Name: Jordan Chen
- Location: Portland, OR (US Pacific timezone)
- Job: Senior product manager at a fintech startup

## Work Context
- Our product is a B2B payments platform
- Main tools: Linear for tasks, Notion for docs, Slack for comms
- Sprint cycles are two weeks, starting Mondays
- I report to the VP of Product (Dana)

## Preferences
- I prefer bullet points over long paragraphs
- I think in frameworks -- help me structure ideas, don't just brainstorm
- I hate meetings. If something can be an async message, suggest that.
- Morning person. I'm sharpest before noon.

## Family & Personal
- Partner: Alex (they/them), works in education
- Dog: Biscuit (golden retriever, 4 years old)
- Hobby: Woodworking on weekends

## Communication Style
- Direct. I'd rather hear "that idea won't work because X" than a diplomatic runaround.
- I use "lmk" for "let me know" and "ty" for "thank you" -- don't expand these
```

What NOT to Include

USER.md is a profile, not a password manager. Never put these in your workspace files:

- **Passwords or API keys** – Use environment variables or a secrets manager
- **Social Security numbers, bank account numbers, or financial credentials**
- **Authentication tokens or session cookies**
- **Anything you'd be uncomfortable seeing in a log file**

> **Warning: Security First**
>
> Workspace files are plain text on your filesystem. Anyone with access to your machine can read them. If you're syncing your workspace to a cloud service or version-controlling it with Git, be especially careful about what goes into USER.md. Treat it like a document that might be read by a stranger, and keep sensitive data elsewhere. See **Chapter 10: Security** for a thorough treatment.

How Persistent Memory Works

Understanding the memory system is essential to personalization. If you don't know how your agent remembers (and forgets), you'll be confused when it seems to lose context. The full story is in **Memory: How the AI Actually Remembers**, but here's the practical overview.

The Two Layers

Layer 1: Daily Logs. Every day, your agent keeps a running diary at `~/.openclaw/workspace/memory/YYYY-MM-DD.md`. As you talk, it jots down facts, decisions, and context. When a new session starts, OpenClaw loads today's log and yesterday's log into the AI's context. Anything older drops out of immediate view.

Layer 2: MEMORY.md. This is the permanent reference card at `~/.openclaw/workspace/MEMORY.md`. It holds durable facts – your name, your preferences, important decisions. MEMORY.md is loaded into *every* session, no matter how old it is.

Think of it this way: daily logs are your assistant's scratch pad, and MEMORY.md is a sticky note pinned to the monitor.

How Compaction Preserves Context

When a conversation approaches the context window limit, OpenClaw runs **compaction** – summarizing older parts of the conversation to free up space. Before compaction, the agent is prompted to flush important details to memory files first. But compaction isn't perfect. If you mentioned something briefly and the agent didn't flag it as important, it might not make it into MEMORY.md. This is why manual curation matters.

When and Why the Agent "Forgets"

Your agent will seem to forget things in predictable situations:

1. **New session, information was only in the conversation.** If something was discussed but never written to a memory file, it disappears when the session ends. Fix: say "remember this" during the conversation, or add it to MEMORY.md yourself.
2. **Information is in an old daily log.** Only today's and yesterday's logs are loaded automatically. Older logs exist on disk but won't surface unless the agent performs a memory search. Fix: move important facts to MEMORY.md so they're always loaded.
3. **MEMORY.md was compacted or grew too large.** If MEMORY.md becomes very long, less important entries might get pruned during maintenance. Fix: keep MEMORY.md focused on durable facts and review it periodically.
4. **Context window filled with other content.** Workspace files, memory files, and conversation history all compete for context window space. If the conversation is long and memory files are large, something gets squeezed out. Fix: keep workspace files concise.

> **Info: The Golden Rule of AI Memory**
>
> If it's not written to a file, it doesn't exist after the session ends. Your agent's memory is exactly as good as the files on disk. No more, no less. If something matters, make sure it's in MEMORY.md.

Using External Memory Tools

OpenClaw's built-in memory system works well for most use cases, but some users want cross-session context that's more robust than file-based memory – especially when switching between devices or running multiple agents. Tools like Supermemory.ai can serve as an external memory layer: a cloud-based memory service that stores, indexes, and retrieves context across sessions and devices.

> **Tip: When to Consider External Memory**
>
> External memory tools are most valuable when you (a) use OpenClaw across multiple devices, (b) run multiple agents that need shared context, or (c) find yourself frequently re-explaining things the agent should already know. If none of these apply, the built-in memory system is likely sufficient.

Setting up an external memory tool typically involves adding it as a tool in your OpenClaw configuration. Check the specific tool's documentation for integration instructions, and see **Chapter 7: Tools & Skills** for how OpenClaw manages tool access.

Curating Memory: Editing, Deleting, and Correcting

This is the part most people skip, and it's the part that matters most. Your agent's memory is not a black box – it's a folder of text files. You should be reading and editing them regularly, the same way you'd correct a colleague who has the wrong information.

Reading and Correcting What the Agent Remembers

Start by opening `~/.openclaw/workspace/MEMORY.md` and browsing the files in `~/.openclaw/workspace/memory/`. You'll likely find a mix of accurate facts, outdated information, and things that are slightly wrong. This is normal.

See something wrong? Edit it directly. The agent doesn't know the difference between what it wrote and what you edited. Next session, it reads the files as they are and treats everything as truth.

Deleting What Shouldn't Be There

Some things end up in memory that you'd rather the agent forget – a vent session about a coworker, outdated project details from a job you've left, experimental instructions you were testing. Delete those lines, or delete entire daily log files if they're old and irrelevant. There is no "undo" on the agent's side – once you remove a line, the agent has no way to recover it.

Adding What's Missing

Sometimes the best way to improve your agent's memory is to write in it yourself. Open MEMORY.md and add facts the agent should always know:

```
## Work
- Currently leading the Q2 platform migration project
- Key stakeholders: Dana (VP Product), Raj (Engineering Lead), Sam (Design)
- Deadline: March 15

## Preferences (added manually)
- I prefer dark mode in all applications
- When recommending restaurants, I'm vegetarian
- My preferred code editor is VS Code with the Catppuccin theme
```

The agent will read these entries in every future session and treat them as established facts.

The Bootstrapping Ritual

The bootstrap process – the initial conversation where the agent asks your name, learns your preferences, and sets up the workspace files – creates a starting point. But it's not a one-time event. You can re-run it to refresh the agent's understanding of you.

```
openclaw bootstrap
```

Re-running bootstrap is useful when you've made significant life or work changes, your workspace files have drifted, or you feel like the agent's responses have become stale or misaligned.

> **Warning: What Re-Bootstrapping Does**
>
> Re-running bootstrap triggers a new orientation conversation. The agent will re-read your workspace files and may update them based on the conversation. It does *not* delete your memory files or conversation history – those are preserved. But it may overwrite parts of SOUL.md, USER.md, and IDENTITY.md if the conversation leads in that direction. If you've carefully hand-tuned those files, back them up first.

Think of re-bootstrapping like a quarterly check-in with your assistant. "Here's what's changed, here's what's working, here's what isn't."

Common Pitfalls

These are the mistakes that show up again and again.

Overly Long SOUL.md Files

Every workspace file competes for space in the context window. A 3,000-word SOUL.md sounds thorough, but it means less room for conversation history, memory, and tool results. Worse, if the file is long enough, the model may start ignoring instructions buried in the middle – attention isn't uniform across long documents.

Target length: 200-400 words for SOUL.md. If you can't fit it in that range, you're probably including things that belong in AGENTS.md or USER.md instead.

Contradictory Instructions

"Be concise" and "always explain your reasoning in detail" can't both be true. "Never use emoji" alongside "use emoji to keep things casual" will confuse the model. Read your workspace files as a whole and look for conflicts. A useful test: read your SOUL.md out loud. If it sounds like it was written by a committee, simplify it.

Instructions That Fight the Model's Nature

Every AI model has inherent tendencies baked in by its training. You can *redirect* these tendencies, but you can't fully override them. You can make the model less verbose, but probably not monosyllabic. You can reduce hedging, but it may still qualify truly uncertain answers. Work *with* the model's grain, not against it. If you're writing increasingly aggressive instructions to force a behavior and it's still not working, the model's safety training may be preventing it. That's usually a feature, not a bug.

Forgetting to Curate Memory

Memory files grow automatically. Without periodic review, they accumulate outdated facts and minor errors. Every few weeks, open MEMORY.md and recent daily logs. Five minutes of maintenance prevents weeks of "why does it keep getting this wrong?"

Worked Example: Before and After

Let's see the full transformation from a default agent to a personalized one.

Before: The Default Agent

With an empty workspace, a conversation looks like this:

```
You: What should I have for dinner?

Agent: There are many great dinner options! Here are some ideas:
1. Grilled chicken with roasted vegetables
2. Pasta with marinara sauce
3. A fresh salad with your favorite toppings
4. Stir-fry with rice
5. Tacos with various fillings

Would you like me to suggest a specific recipe for any of these? I'd be happy to help
you plan a delicious meal! 😊
```

Generic. Over-eager. Useless.

After: The Personalized Agent

With a tuned SOUL.md, a populated USER.md, and curated MEMORY.md:

```
You: What should I have for dinner?

Agent: You mentioned you're trying to cook more this week. Given that it's Wednesday
and you've got that early meeting tomorrow, something quick makes sense.

That black bean soup recipe you bookmarked last weekend would work -- you already have
most of the ingredients from the grocery run on Monday. About 30 minutes, mostly
hands-off. Want the recipe pulled up?
```

Same model, completely different experience. It knows you're vegetarian (USER.md), remembers your grocery run (daily log), knows about tomorrow's meeting (calendar tool + AGENTS.md), and speaks without filler (SOUL.md). Here are the workspace files that produced this:

```
<!-- SOUL.md (excerpt) -->
## Style
- Short, direct answers. No preamble.
- Recommend one option with reasoning, not a list of five.
- Reference things you know about my schedule and preferences.
- No emoji. No exclamation marks. No "I'd be happy to help."
```

```
<!-- USER.md (excerpt) -->
## Diet
- Vegetarian. No exceptions, no "you could substitute chicken" suggestions.

## Cooking
- Trying to cook more at home in 2026
- Prefers recipes under 45 minutes on weeknights
```

```
<!-- MEMORY.md (excerpt) -->
- Grocery run on Monday 2/8: black beans, rice, peppers, onions, cilantro, limes
- Bookmarked a black bean soup recipe from NYT Cooking on Sunday
- Early meeting Thursday 2/12 at 8am with Dana -- prep needed
```

Troubleshooting

The agent ignores my SOUL.md instructions. Check the file path – it must be at `~/.openclaw/workspace/SOUL.md` exactly. Open it and verify the content is what you expect. If the file is very long (over 500 words), try trimming it. Some instructions may be getting lost in a large context. Also check for contradictions between SOUL.md and AGENTS.md.

The agent keeps forgetting things I've told it. Open MEMORY.md and the recent daily logs. If the information isn't there, it was never written to disk. During future conversations, say "remember this" explicitly when you mention something important. For truly durable facts, add them to MEMORY.md yourself.

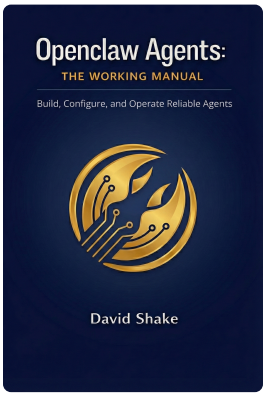
My AGENTS.md trigger phrases don't work. Make sure the trigger phrase is distinct and not something that appears in normal conversation. "When I say 'standup'" is clear. "When I ask about my day" is ambiguous. Also verify the file is saved and the agent has started a new session since your edit – workspace files are loaded at session start.

The agent's personality drifts over long conversations. This is a known limitation of context windows. As the conversation grows and older content gets compacted, the influence of SOUL.md can weaken relative to the volume of conversation. Starting a fresh session (or just waiting for the session timeout) reloads the workspace files at full strength.

Memory files are getting very large. MEMORY.md should stay under 200 lines. If it's growing past that, prune aggressively. Remove anything outdated, consolidate related entries, and move transient information to daily logs. Old daily log files (more than a few weeks) can usually be deleted.

Re-bootstrapping overwrote my custom SOUL.md. Before re-running bootstrap, back up your workspace: `cp -r ~/.openclaw/workspace ~/.openclaw/workspace-backup`. After bootstrap, compare the files and merge any changes you want to keep.

The agent responds differently on different channels. Check if you have channel-specific session scoping enabled. With `per-channel-peer` scoping, each channel gets its own session and loads memory independently. If you want consistent behavior, use the default `main` scoping mode. See **Sessions** for details.



Enjoyed this guide?

The full book covers architecture, channels, security, models, memory, automation, and six complete setup guides — everything you need to run OpenClaw agents that actually work.

GET THE BOOK

Kindle Edition • **Apple Books** • **Direct (EPUB)**

BUY ME A COFFEE

If this free guide saved you time or money, you can say thanks with a one-time contribution. No account needed.

$5 • **$10** • **$25**