Beyond the Chatbot

Engineering Your Proactive Digital Twin

CKEditor

The last 12 months taught me one thing: The best approach to get into various utilization with AI is.....
JUST GET STARTED!

Ondřej Chrastina
Developer Advocate
CKEditor

ondrej.chrastina.dev

Agenda

- NanoClaw installation
- Connection to Telegram
- Living files & Building digital twin
- Connect to APIs via OneCLI
- Capabilities & Use cases ✨

https://l.ead.me/gh-digital-twin

*Everything on virtual machine

The Launchpad: Prerequisites

The Virtual Machine

HARD PREREQUISITE

OS: Linux VM (Ubuntu 22.04+ LTS)

Tools: UTM, VirtualBox, or KVM

Specs: Min 8GB RAM / 30GB Disk

Pro-Tip: Run native archs (ARM64 on Apple Silicon, x86_64 on Intel) to avoid lag.

AI Access

REQUIRED CREDENTIALS

Claude Pro
or Anthropic API Key

— OR —

ChatGPT Plus
or OpenAI API Key

The Channel

TELEGRAM INTERFACE

✔ Telegram installed on your mobile device.

✔ Chat with **@BotFather** to trigger setup.

✔ Generate a unique **Bot Token**.

Surviving Conference WiFi: The Pre-Cache

`scripts/prepare.sh`

Downloads ~2 GB of dependencies (Docker, Node, pnpm, Claude CLIs, Base Images) prior to the event.

Pins the installation strictly to v2.1.17 to guarantee zero behavior changes or surprise downloads.

NOGO: Docker-in-Docker Setup

Warning: Do not attempt to run the Docker-in-Docker setup for NanoClaw. The source explicitly warns of 9 distinct DinD-specific issues. Stick to the standard VM host installation.

The Monolith vs. The Bespoke Micro-Framework

OpenClaw

Scale
~500,000 lines of code, 70+ dependencies

Security
Application-level allowlists

Memory
Shared Node process

Customization
53 configuration files

NanoClaw

Scale
One Node host, a few source files

Security
OS-level Linux container isolation

Memory
Per-agent filesystem isolation

Customization
Zero config sprawl; modified directly via AI prompts

OpenClaw and Friends: Claw, Nano, Zero, Pico... So Many Overlapping Projects, I'm Confused

Hey everyone,

I've been exploring self-hosted AI/OpenClaw alternative projects, but it's starting to get really confusing. There are already many projects, often with overlapping or almost identical names. For example:

* [openclaw/openclaw](https://github.com/openclaw/openclaw)
* [qwibitai/nanoclaw](https://github.com/qwibitai/nanoclaw)
* [sipeed/picoclaw](https://github.com/sipeed/picoclaw)
* [openagen/zeroclaw](https://github.com/openagen/zeroclaw)
* [zeroclaw-labs/zeroclaw](https://github.com/zeroclaw-labs/zeroclaw)
* [nanobot-ai/nanobot](https://github.com/nanobot-ai/nanobot)
* [HKUDS/nanobot](https://github.com/HKUDS/nanobot)

*** EDIT (Found even more)**

* [TinyAGI/tinyclaw](https://github.com/TinyAGI/tinyclaw)
* [warengonzaga/tinyclaw](https://github.com/warengonzaga/tinyclaw)
* [nullclaw/nullclaw](https://github.com/nullclaw/nullclaw)
* [nearai/ironclaw](https://github.com/nearai/ironclaw)
* [heypinchy/pinchy](https://github.com/heypinchy/pinchy)
* [qhkm/zeptoclaw](https://github.com/qhkm/zeptoclaw)

- https://evoailabs.medium.com/openclaw-nanobot-picoclaw-ironclaw-and-zeroclaw-this-claw-craziness-is-continuing-87c72456e6dc
- https://www.reddit.com/r/SelfHosting/comments/1r7z4wf/openclaw_and_friends_claw_nano_zero_pico_so_many/

The Paradigm Shift: Verbalize, Don't Code

NanoClaw abandons configuration files entirely. Instead of tweaking settings in a bloated framework, you command the AI to modify its own underlying codebase.

Legacy Config Sprawl

~~# config.yaml~~
~~settings:~~
~~trigger_word: "bot"~~
~~history_limit: 100~~

❌ ABANDONED ENTIRELY

AI-Driven Commands

$ _

"Change my trigger word to @Bob and store conversation summaries weekly."

⚡ SELF-MODIFYING CODEBASE

The codebase is intentionally small enough that an LLM can understand, modify, and safely rewrite it without breaking the architecture.

Message to Memory: The Architectural Flow

The Database Bridge

Two SQLite files per session.

- Exactly one writer per file.
- No cross-mount contention.

Container Boundary

Agents run isolated in Docker.

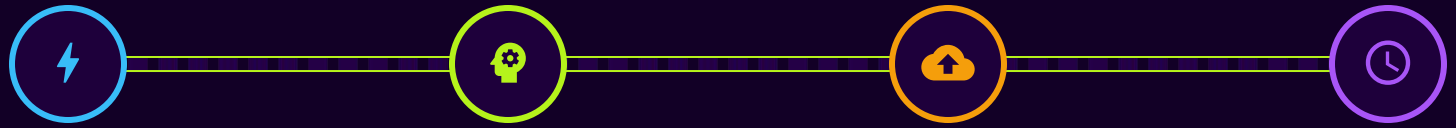
- Complete environment safety.
- Only see explicitly mounted directories.

OneCLI Agent Vault

Secure credentials vault.

- Never enters the container.
- Injected strictly at the outbound request level.

The Web Summer Camp 2026 Roadmap



PHASE 1

1. Install & Pair

Run nanoclaw.sh, add AI credentials, and pair with a Telegram bot for immediate DM access.

PHASE 2

2. Living Files

Execute living-files to teach the agent your profile, giving it persistent, personalized memory.

PHASE 3

3. Knowledge

Connect OneCLI OAuth to route approved agent outputs directly into a private GitHub repository.

PHASE 4

4. Operations

Use natural language to deploy a schedule task that wakes the agent automatically for a daily brief.

Let’s get to it!

See you soon

Engineering Your
Proactive Digital Twin

ondrej.chrastina.dev

https://l.ead.me/gh-digital-twin