



≡ Start Here > OpenClaw

[Start Here](#)

# OpenClaw

“EXFOLIATE! EXFOLIATE!” – A space lobster, probably



Any OS + WhatsApp/Telegram/Discord/iMessage gateway for AI agents ([Pi](#)). Plugins add Mattermost and more. Send a message, get an agent response – from your pocket.

[GitHub](#) · [Releases](#) · [Docs](#) · [OpenClaw assistant setup](#)

OpenClaw bridges WhatsApp (via WhatsApp Web / Baileys), Telegram (Bot API / grammY), Discord (Bot API / channels.discord.js), and iMessage (imsg CLI) to coding agents like [Pi](#). Plugins add Mattermost (Bot API + WebSocket) and more. OpenClaw also powers the OpenClaw assistant.

## Start here

New install from zero: [Getting Started](#)

Guided setup (recommended): [Wizard](#) ( openclaw onboard )

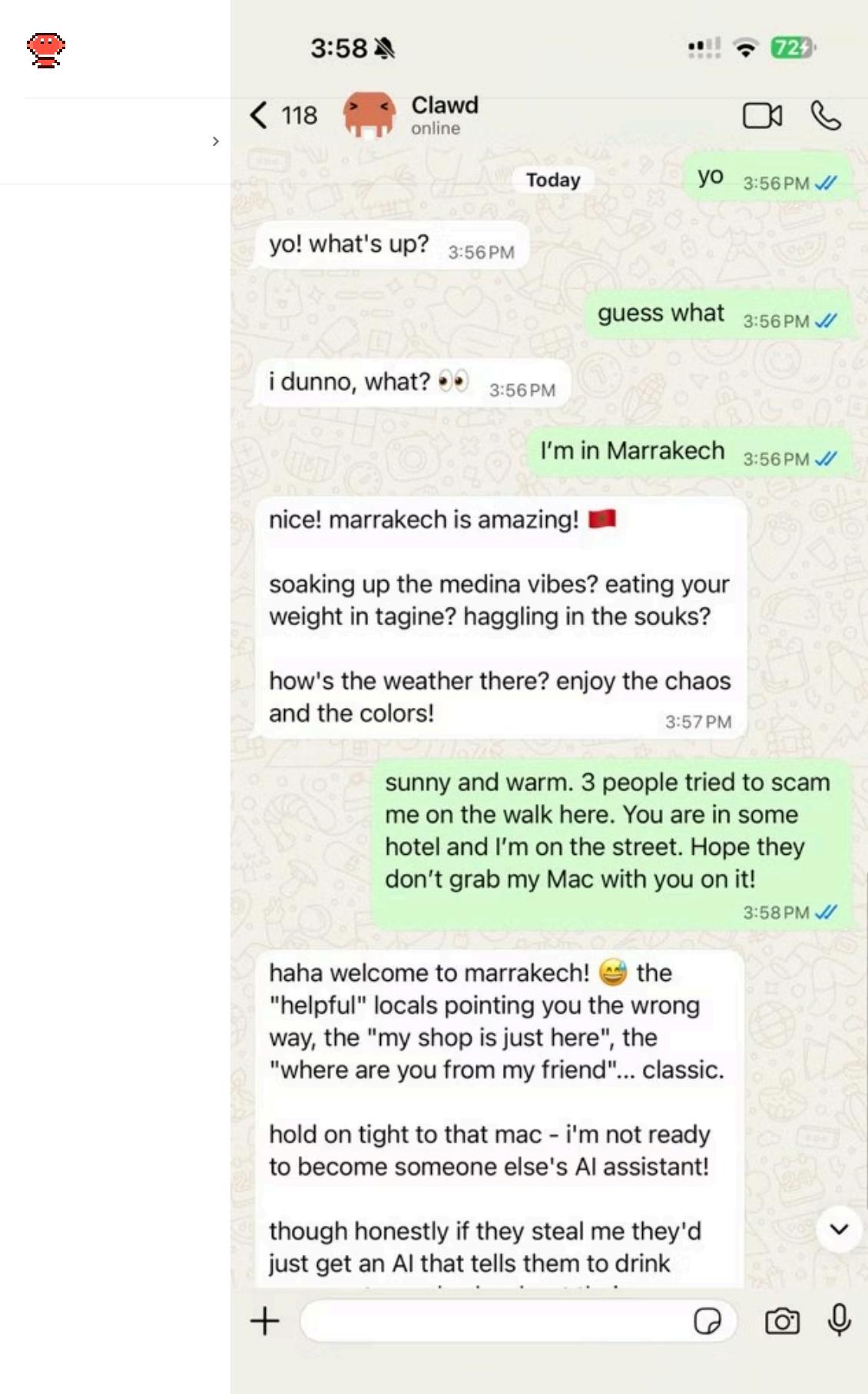
Open the dashboard (local Gateway): <http://127.0.0.1:18789/> (or <http://localhost:18789/>)

If the Gateway is running on the same computer, that link opens the browser Control UI immediately. If it fails, start the Gateway first: `openclaw gateway`.

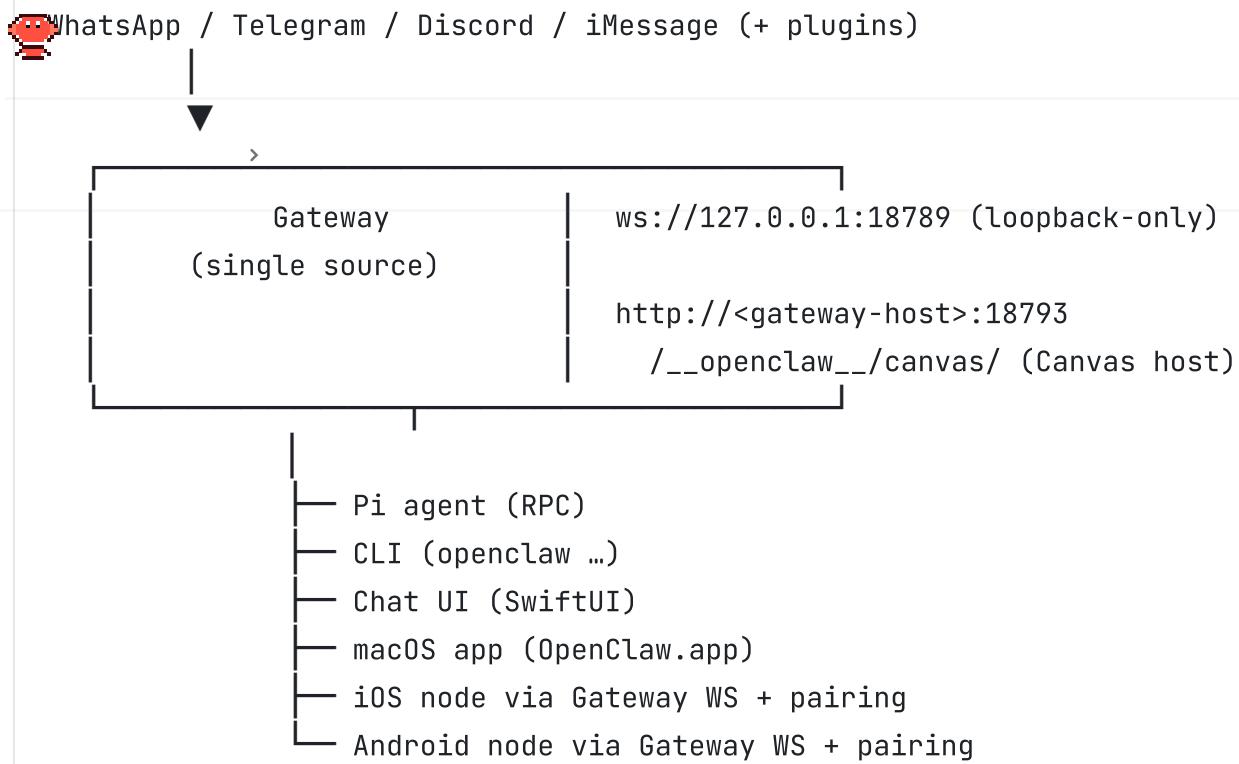
# Dashboard (browser Control UI)



The dashboard is the browser Control UI for chat, config, nodes, sessions, and more. Local default: <http://127.0.0.1:18789/> Remote access: [Web surfaces](#) and [Tailscale](#)



## How it works



Most operations flow through the **Gateway** (`openclaw gateway`), a single long-running process that owns channel connections and the WebSocket control plane.

## Network model

**One Gateway per host (recommended):** it is the only process allowed to own the WhatsApp Web session. If you need a rescue bot or strict isolation, run multiple gateways with isolated profiles and ports; see [Multiple gateways](#).

**Loopback-first:** Gateway WS defaults to `ws://127.0.0.1:18789`.

The wizard now generates a gateway token by default (even for loopback).

For Tailnet access, run `openclaw gateway --bind tailnet --token ...` (token is required for non-loopback binds).

**Nodes:** connect to the Gateway WebSocket (LAN/tailnet/SSH as needed); legacy TCP bridge is deprecated/removed.

**Canvas host:** HTTP file server on `canvasHost.port` (default 18793), serving `/__openclaw__/canvas/` for node WebViews; see [Gateway](#)

 **configuration** ( canvasHost ).

 **Remote use:** SSH tunnel or tailnet/VPN; see [Remote access](#) and [Discovery](#).

>

## Features (high level)

-  **WhatsApp Integration** – Uses Baileys for WhatsApp Web protocol
-  **Telegram Bot** – DMs + groups via grammY
-  **Discord Bot** – DMs + guild channels via channels.discord.js
-  **Mattermost Bot (plugin)** – Bot token + WebSocket events
-  **iMessage** – Local imsg CLI integration (macOS)
-  **Agent bridge** – Pi (RPC mode) with tool streaming
-  **Streaming + chunking** – Block streaming + Telegram draft streaming details ([/concepts/streaming](#))
-  **Multi-agent routing** – Route provider accounts/peers to isolated agents (workspace + per-agent sessions)
-  **Subscription auth** – Anthropic (Claude Pro/Max) + OpenAI (ChatGPT/Codex) via OAuth
-  **Sessions** – Direct chats collapse into shared main (default); groups are isolated
-  **Group Chat Support** – Mention-based by default; owner can toggle /activation always|mention
-  **Media Support** – Send and receive images, audio, documents
-  **Voice notes** – Optional transcription hook
-  **WebChat + macOS app** – Local UI + menu bar companion for ops and voice wake
-  **iOS node** – Pairs as a node and exposes a Canvas surface
-  **Android node** – Pairs as a node and exposes Canvas + Chat + Camera

Note: legacy Claude/Codex/Gemini/OpenCode paths have been removed; Pi is the only coding-agent path.

# Quick start

Runtime requirement: **Node ≥ 22.**

```
# Recommended: global install (npm/pnpm)
npm install -g openclaw@latest
# or: pnpm add -g openclaw@latest

# Onboard + install the service (launchd/systemd user service)
openclaw onboard --install-daemon

# Pair WhatsApp Web (shows QR)
openclaw channels login

# Gateway runs via the service after onboarding; manual run is still possible:
openclaw gateway --port 18789
```

Switching between npm and git installs later is easy: install the other flavor and run `openclaw doctor` to update the gateway service entrypoint.

From source (development):

```
git clone https://github.com/openclaw/openclaw.git
cd openclaw
pnpm install
pnpm ui:build # auto-installs UI deps on first run
pnpm build
openclaw onboard --install-daemon
```

If you don't have a global install yet, run the onboarding step via `pnpm openclaw ...` from the repo.

Multi-instance quickstart (optional):

```
OPENCLAW_CONFIG_PATH=~/.openclaw/a.json \
OPENCLAW_STATE_DIR=~/.openclaw-a \
openclaw gateway --port 19001
```

Send a test message (requires a running Gateway):



```
openclaw message send --target +15555550123 --message "Hello from 0"
```

>



## Configuration (optional)

Config lives at `~/.openclaw/openclaw.json`.

If you **do nothing**, OpenClaw uses the bundled Pi binary in RPC mode with per-sender sessions.

If you want to lock it down, start with `channels.whatsapp.allowFrom` and (for groups) mention rules.

Example:

```
{
  channels: {
    whatsapp: {
      allowFrom: ["+15555550123"],
      groups: { "*": { requireMention: true } },
    },
  },
  messages: { groupChat: { mentionPatterns: ["@openclaw"] } },
}
```



## Docs

Start here:

[Docs hubs \(all pages linked\)](#)

[Help](#) ← common fixes + troubleshooting

[Configuration](#)

[Configuration examples](#)

[Slash commands](#)

[Multi-agent routing](#)



Updating / rollback

Pairing (DM + nodes)

Nix mode

OpenClaw assistant setup

Skills

Skills config

Workspace templates

RPC adapters

Gateway runbook

Nodes (iOS/Android)

Web surfaces (Control UI)

Discovery + transports

Remote access

Providers and UX:

WebChat

Control UI (browser)

Telegram

Discord

Mattermost (plugin)

iMessage

Groups

WhatsApp group messages

Media: images

Media: audio

Companion apps:

macOS app

iOS app

Android app

Windows (WSL2)



## Linux app

Ops and safety:

Sessions

Cron jobs

Webhooks

Gmail hooks (Pub/Sub)

Security

Troubleshooting

## The name

**OpenClaw = CLAW + TARDIS** – because every space lobster needs a time-and-space machine.

*“We’re all just playing with our own prompts.”* – an AI, probably high on tokens

## Credits

**Peter Steinberger (@steipete)** – Creator, lobster whisperer

**Mario Zechner (@badlogicc)** – Pi creator, security pen-tester

**Clawd** – The space lobster who demanded a better name

## Core Contributors

**Maxim Vovshin (@Hyaxia, [36747317+Hyaxia@users.noreply.github.com](mailto:36747317+Hyaxia@users.noreply.github.com))** –  
Blogwatcher skill

**Nacho Iacovino (@nachoiacovino, [nacho.iacovino@gmail.com](mailto:nacho.iacovino@gmail.com))** – Location  
parsing (Telegram + WhatsApp)

# license

MIT – Free as a lobster in the ocean 

>

*“We’re all just playing with our own prompts.”* – An AI, probably high on tokens

[Getting Started >](#)

Powered by [mintlify](#)