Start Here

Getting Started

Goal: go from **zero** → **first working chat** (with sane defaults) as quickly as possible.

Fastest chat: open the Control UI (no channel setup needed). Run `openclaw dashboard` and chat in the browser, or open `http://127.0.0.1:18789/` on the gateway host. Docs: **Dashboard** and **Control UI**.

Recommended path: use the **CLI onboarding wizard** (`openclaw onboard`). It sets up:

- model/auth (OAuth recommended)
- gateway settings
- channels (WhatsApp/Telegram/Discord/Mattermost (plugin)/…)
- pairing defaults (secure DMs)
- workspace bootstrap + skills
- optional background service

If you want the deeper reference pages, jump to: **Wizard**, **Setup**, **Pairing**, **Security**.

Sandboxing note: `agents.defaults.sandbox.mode: "non-main"` uses `session.mainKey` (default `"main"`), so group/channel sessions are sandboxed. If you want the main agent to always run on host, set an explicit per-agent override:

```
{
  "routing": {
    "agents": {
      "main": {
        "workspace": "~/.openclaw/workspace",
        "sandbox": { "mode": "off" }
      }
    }
  }
}
```

0) Prereqs

Node >=22

pnpm (optional; recommended if you build from source)

Recommended: Brave Search API key for web search. Easiest path: `openclaw configure --section web` (stores `tools.web.search.apiKey`). See **Web tools**.

macOS: if you plan to build the apps, install Xcode / CLT. For the CLI + gateway only, Node is enough. Windows: use **WSL2** (Ubuntu recommended). WSL2 is strongly recommended; native Windows is untested, more problematic, and has poorer tool compatibility. Install WSL2 first, then run the Linux steps inside WSL. See **Windows (WSL2)**.

1) Install the CLI (recommended)

```
curl -fsSL https://openclaw.ai/install.sh | bash
```

Installer options (install method, non-interactive, from GitHub): **Install**.

Windows (PowerShell):

```
iwr -useb https://openclaw.ai/install.ps1 | iex
```

Alternative (global install):

```
npm install -g openclaw@latest
```

```
pnpm add -g openclaw@latest
```

2) Run the onboarding wizard (and install the service)

```
openclaw onboard --install-daemon
```

What you'll choose:

- **Local vs Remote** gateway
- **Auth**: OpenAI Code (Codex) subscription (OAuth) or API keys. For Anthropic we recommend an API key; `claude setup-token` is also supported.
- **Providers**: WhatsApp QR login, Telegram/Discord bot tokens, Mattermost plugin tokens, etc.
- **Daemon**: background install (launchd/systemd; WSL2 uses systemd)
  - **Runtime**: Node (recommended; required for WhatsApp/Telegram). Bun is **not recommended**.
- **Gateway token**: the wizard generates one by default (even on loopback) and stores it in `gateway.auth.token`.

Wizard doc: **Wizard**

Auth: where it lives (important)

- **Recommended Anthropic path:** set an API key (wizard can store it for service use). `claude setup-token` is also supported if you want to reuse Claude Code credentials.
- OAuth credentials (legacy import): `~/.openclaw/credentials/oauth.json`

Auth profiles (OAuth + API keys): `~/.openclaw/agents/<agentId>/agent/auth-profiles.json`

Headless/server tip: do OAuth on a normal machine first, then copy `oauth.json` to the gateway host.

3) Start the Gateway

If you installed the service during onboarding, the Gateway should already be running:

```
openclaw gateway status
```

Manual run (foreground):

```
openclaw gateway --port 18789 --verbose
```

Dashboard (local loopback): `http://127.0.0.1:18789/` If a token is configured, paste it into the Control UI settings (stored as `connect.params.auth.token`).

⚠️ **Bun warning (WhatsApp + Telegram):** Bun has known issues with these channels. If you use WhatsApp or Telegram, run the Gateway with **Node**.

3.5) Quick verify (2 min)

```
openclaw status
openclaw health
openclaw security audit --deep
```

4) Pair + connect your first chat surface

WhatsApp (QR login)

```
openclaw channels login
```

Scan via WhatsApp → Settings → Linked Devices.

WhatsApp doc: **WhatsApp**

Telegram / Discord / others

The wizard can write tokens/config for you. If you prefer manual config, start with:

- Telegram: **Telegram**
- Discord: **Discord**
- Mattermost (plugin): **Mattermost**

Telegram DM tip: your first DM returns a pairing code. Approve it (see next step) or the bot won't respond.

5) DM safety (pairing approvals)

Default posture: unknown DMs get a short code and messages are not processed until approved. If your first DM gets no reply, approve the pairing:

```
openclaw pairing list whatsapp
openclaw pairing approve whatsapp <code>
```

Pairing doc: **Pairing**

From source (development)

If you're hacking on OpenClaw itself, run from source:

```
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. `pnpm build` also bundles A2UI assets; if you need to run just that step, use `pnpm canvas:a2ui:bundle`.

Gateway (from this repo):

```
node openclaw.mjs gateway --port 18789 --verbose
```

7) Verify end-to-end

In a new terminal, send a test message:

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

If `openclaw health` shows "no auth configured", go back to the wizard and set OAuth/key auth — the agent won't be able to respond without it.

Tip: `openclaw status --all` is the best pasteable, read-only debug report. Health probes: `openclaw health` (or `openclaw status --deep`) asks the running gateway for a health snapshot.

Next steps (optional, but great)

- macOS menu bar app + voice wake: **macOS app**
- iOS/Android nodes (Canvas/camera/voice): **Nodes**
- Remote access (SSH tunnel / Tailscale Serve): **Remote access** and **Tailscale**
- Always-on / VPN setups: **Remote access**, **exe.dev**, **Hetzner**, **macOS remote**

‹ OpenClaw Onboarding Wizard ›