

# Module 11

## Maintenance and Troubleshooting 🦀

Keeping Your Agent Running Smoothly

# Navigation Chart

---

By the end of this module, you will be able to:

1. **Follow** a daily and weekly maintenance routine
2. **Monitor and optimize** your token spending
3. **Update** 🦀 OpenClaw safely
4. **Back up and restore** your workspace
5. **Diagnose and fix** the most common errors
6. **Use** the command cheat sheet for quick reference
7. **Know** where to get help from the community

## Ship's Logbook

| Term               | Definition   |
|--------------------|--|
| Token burn         | The rate at which your AI conversations consume tokens (and money)                                 |
| Context compaction | Using <code>/compact</code> to summarize conversation history and reduce token usage               |
| Workspace backup   | A Git commit of your <code>~/ .openclaw/</code> directory that captures your agent's current state |
| Rolling update     | Updating 🦸 OpenClaw while the gateway is running, with minimal downtime                            |

# Realistic Timelines

---

## Level 1 -- Works Immediately

Basic conversation, file management, simple research, document summarization

## Level 2 -- Requires Hours to Days of Setup

Email automation, trading bots, social media management, code projects, multi-step workflows

## The Reality


- YouTube demos show polished setups that took **days or weeks** to build
- Budget **2-4 hours/week** for setup and maintenance during your first month
- The investment pays off -- but not on day one

## ROI

5 hours/week saved = **200+ hours/year**. At \$50/hour, that's \$10,000/year in value. But expect setup time first

# Daily Maintenance Checklist

This takes about **2 minutes**. Do it every day during your first month.



-  Gateway is running: `openclaw status`
- No errors in logs: `openclaw service logs`
- Token usage reasonable: check your provider dashboard
- Morning brief was sent: check Telegram
- No pending pairing requests: check dashboard for unknown senders

*If everything checks out, you are good. If anything is off, see the troubleshooting slides later in this deck.*

# Weekly Maintenance Checklist

---

This takes about **10 minutes**. Do it every weekend.

- Security audit: `openclaw security audit --deep`
- Health check: `openclaw doctor`
- Fix any issues: `--fix` flags on both commands
- Review API spending: visit your provider dashboard
-  Workspace backup: `git add -A && git commit -m "Weekly backup"`
- Review core files: are `USER.md` and `MEMORY.md` still accurate?
- Disable unused  skills: `openclaw skills list`
- Check disk space: `df -h` (inside WSL2)

# Cost Reduction Strategies

| Strategy                            | Savings                           | Effort |
|-------------------------------------|-----------------------------------|--------|
| Enable prompt caching               | ~90% off system prompt costs      | Low    |
| Use Haiku for heartbeats            | \$50-190/month                    | Low    |
| Set heartbeat to 55 min             | Stays within cache window         | Low    |
| Install ClawRouter                  | Up to 90% via intelligent routing | Medium |
| Use <code>/compact</code> regularly | ~67% per conversation             | Low    |

## Cost Reduction Strategies (continued)

| Strategy   | Savings                              | Effort |
|--|--------------------------------------|--------|
| <b>Start new sessions</b> for new topics           | Prevents runaway token counts        | Low    |
| <b>Disable unused</b> 🐟 <b>skills</b>              | Reduces context (3K-14K tokens/call) | Low    |
| <b>Use Sonnet for routine tasks</b>                | 50-75% cheaper than Opus             | Low    |
| <b>Use budget models</b> (Kimi K2.5, MiniMax M2.5) | Up to 95% cheaper than Opus          | Low    |
| <b>Set API spending limits</b>                     | Prevents surprise bills              | Low    |



# Setting API Spending Limits

---

## Anthropic:

1. Go to [console.anthropic.com](https://console.anthropic.com) -- Settings -- Billing
2. Set monthly usage limit (e.g., \$50)
3. Set alert threshold (e.g., alert at \$40)

## OpenAI:

1. Go to [platform.openai.com](https://platform.openai.com) -- Settings -- Billing
2. Set monthly budget cap

**Recommendation:** Start with a \$50/month limit. Increase only when you understand your usage patterns.

# Updating OpenClaw

**Release channels: Use **stable** (not beta or dev)**

| Channel       | Use For                    |
|---------------|----------------------------|
| <b>stable</b> | Everyone in this course    |
| <b>beta</b>   | Early access (expect bugs) |
| <b>dev</b>    | Contributors only          |

**Always back up first:**

```
cd ~/.openclaw && git add -A && git commit -m "Pre-update backup"  
openclaw status --json > ~/pre-update-status.json
```

**Update:**

```
openclaw update
```

# Backup and Restore

## Quick backup (Git):

```
cd ~/.openclaw && git add -A && git commit -m "Backup"
```

## Full backup (archive):

```
tar -czf ~/openclaw-backup-$(date +%Y%m%d).tar.gz ~/.openclaw/
```

## What to back up:

| Directory                     | Priority  | Contains                                       |
|-------------------------------|-----------|--|
| ~/.openclaw/workspace/        | Critical  | Identity, personality, rules, 🦄 memory         |
| ~/.openclaw/config/           | Critical  | 🚢 Gateway config, 🔑 API keys, channel settings |
| ~/.openclaw/workspace/skills/ | Important | Custom 🐟 skills you built                      |

# Debugging Mindset

Before diving into specific errors, adopt this approach:

## **Rule 1: Run `openclaw doctor --fix` First**

Most "my agent is being stupid" issues are actually **configuration problems**, not AI problems. The doctor command catches and fixes the majority of them automatically.

## **Rule 2: One Workflow at a Time**

Get one workflow running **end-to-end** before adding the next. Every new integration is a new failure mode.

- Set up Telegram? Test it thoroughly before adding Discord.
- Adding a 🐟 skill? Verify it works before installing another.
- Building a cron job? Confirm it triggers correctly before scheduling more.

# ► Damage Control: Gateway Errors

## "Gateway refuses to start" (config schema rejection)

- **Cause:** One invalid key in `openclaw.json` blocks boot entirely
- **Fix:** Run `openclaw doctor` -- it identifies the bad key. Fix it, restart.
- **Prevention:** Use `openclaw config` commands instead of editing JSON directly

## "Gateway not running"


- Check: `openclaw status`
- Fix: `openclaw service start`
- Review: `openclaw service logs`

## "Port already in use"

- Find the process: `lsof -i :18789`
- Kill it or change port: `openclaw config gateway port 18790`

# ▶ Damage Control: API and Channels


## "API key invalid"

- Check  key at your provider's console
- Regenerate if needed (use the Notepad trick when pasting)
- Update: `openclaw config provider key [NEW_KEY]`

## "Rate limit exceeded"

- Wait 1-5 minutes, then retry
- If persistent: reduce heartbeat frequency

## "Telegram bot not responding"

- Check  gateway: `openclaw status`
- Check pairing: look for pending requests
- Verify token in BotFather

# ▶ Damage Control: WSL2

---

## "WSL2 not running after reboot"

- Open PowerShell: `wsl --list --running`
- If empty: open Ubuntu from Start menu

## "WSL2 very slow"

- Exclude WSL2 directory from Windows Defender
- Edit `%USERPROFILE%\.wslconfig` with memory limits
- Run: `wsl --shutdown`, then reopen Ubuntu

## "systemd not running"

- Edit `/etc/wsl.conf`:
  - Add `[boot]` on one line
  - Add `systemd=true` on next line

# 🚩 Damage Control: Performance

---

## "AI responses very slow (30+ seconds)"

- Test internet connectivity
- Check provider status page
- Use `/compact` to reduce context size
- Switch to a faster model temporarily

## "Token counter climbing very fast"

- Use `/compact` to compress conversation
- Use `/new` to start a fresh session
- Disable unused 🐟 skills
- Check if large files are being loaded unnecessarily



# Command Cheat Sheet: Gateway and Status

```
# Start / Stop
openclaw service start      # Start the daemon
openclaw service stop      # Stop the daemon
openclaw service restart   # Restart the daemon
openclaw tui                # Open chat interface

# Check Status
openclaw status             # Is the gateway running?
openclaw doctor             # Health check
openclaw doctor fix         # Auto-fix health issues
openclaw service logs       # View logs
```

# Command Cheat Sheet: Security

```
# Security Audit
openclaw security audit --deep          # Full audit
openclaw security audit --deep --fix    # Audit and auto-fix

# Authentication
openclaw config gateway auth           # Check auth settings
openclaw config gateway token rotate   # Rotate token
openclaw config gateway bind           # Check bind address

# Sandboxing
openclaw config sandbox mode [mode]    # off / non-main / all
openclaw config sandbox scope [scope]  # session / agent / shared

# Tool Policies
openclaw config tools deny [tool]      # Deny a tool
openclaw config tools elevated off     # Disable elevated mode
```

# Command Cheat Sheet: Configuration and Channels

```
# Configuration
openclaw config                # Open configuration
openclaw config gateway port   # Check/set port
openclaw config provider key    # Update 🔑 API key
openclaw config heartbeat interval # Set heartbeat frequency
openclaw config heartbeat model # Set heartbeat model

# Channels
openclaw channels add [channel] # Add messaging channel
openclaw channels login         # Re-authenticate
openclaw config channels [ch] dm-mode # Check/set DM mode

# Cron Jobs
openclaw cron list              # List scheduled jobs
openclaw cron add [name] --schedule [cron] --task [desc]
openclaw cron remove [name]     # Remove a job
openclaw cron disable [name]    # Disable a job
```

# Command Cheat Sheet: Skills and TUI

```
# Skills
openclaw skills browse          # Browse ClawHub
openclaw skills search [q]      # Search skills
openclaw skills install [n]     # Install a skill
openclaw skills list            # List installed
openclaw skills enable [n]      # Enable a skill
openclaw skills disable [n]    # Disable a skill
```

## Inside the Chat (TUI):




| Command                    | Action                               |
|----------------------------|--------------------------------------|
| <code>/help</code>         | List all slash commands              |
| <code>/compact</code>      | Compress conversation to save tokens |
| <code>/new</code>          | Start a fresh session                |
| <code>/model [name]</code> | Switch AI model                      |

## ► Rough Waters: Emergency Quick Reference


| Step           | Action                   | Command  |
|----------------|--------------------------|--|
| 1. STOP        | Kill the agent           | <code>openclaw service stop</code>                 |
| 2. CLOSE       | Lock down access         | <code>openclaw config gateway bind loopback</code> |
| 3. FREEZE      | Rotate all tokens/keys   | <code>openclaw config gateway token rotate</code>  |
| 4. INVESTIGATE | Review logs and sessions | <code>openclaw service logs</code>                 |
| 5. RESTORE     | Fix, audit, restart      | <code>openclaw security audit --deep --fix</code>  |

*Print this table and keep it next to your computer.*

# Community Resources


| Resource  | Best For  |
|---|---|
| <b>Official Docs</b> (docs.openclaw.ai)   | Configuration, features, API reference  |
| <b>GitHub</b> (github.com/openclaw/openclaw)  | Bug reports, feature requests, source code  |
|  <b>OpenClaw Discord</b> | Real-time help, community discussion  |
|  <b>OpenClaw Reddit</b>  | Longer-form discussions, tutorials  |
| <b>ClawHub</b>  |  Skills marketplace, community contributions |

## When asking for help, include:


1. What you were trying to do
2. The exact error message
3. Your  OpenClaw version ( `openclaw --version` )
4. Your OS (Windows 10 WSL2)

# Skills vs. Agents: Know the Difference

## Use a Skill when:

- Task is triggered by a specific command
- Doesn't need its own persistent  memory
- Doesn't run independently
- It's a capability, not an identity

## Use a Separate Agent when:

- Needs persistent  memory separate from main agent
- Needs its own identity (e.g., trading bot with different risk parameters)
- Runs independently and proactively
- Communicates through different channels

**Default to skills** Every agent has its own context window and token costs. Multiple agents

# Course Completion Checklist

---

## You should now have:

- **Your agent** -- a fully configured AI assistant running 24/7
- **Security** -- sandboxing, tool policies, authentication, incident response plan
- **Communication** -- Telegram (and optionally Discord/WhatsApp)
- **Automation** -- morning briefs, cron jobs, proactive heartbeats
- 🐟 **Skills** -- from ClawHub and custom-built
- **Knowledge** -- how it all works, how to maintain it, how to troubleshoot it

## What to do next:

- Use it daily -- 🌿 memory builds over time
- Update core files monthly -- your goals change
- Try reverse prompting weekly



# Treasure Chest

---

1. **Maintenance is not optional** -- daily checks (2 min) and weekly audits (10 min) prevent problems
2. **Monitor your spending** -- set limits and check your provider dashboard weekly
3. **Back up religiously** -- Git commit your workspace at least weekly
4. **Update carefully** -- back up before updating, test after updating
5. **Keep the cheat sheet accessible** -- print it, bookmark it, save it to your phone
6. **The community is your resource** -- Discord, GitHub, and Reddit when you are stuck
7. **Security is ongoing** -- run audits weekly, rotate secrets periodically

 **Bon Voyage!**

## Congratulations -- You Have Completed the Course

*You're not just running a chatbot. You're running infrastructure.  
Infrastructure for thought. Infrastructure for action. Infrastructure for a  
new kind of relationship between humans and AI.*

**Welcome to the future. You built it safely. Now go use it.**