

# OpenClaw para Audiencia Técnica

- Arquitectura, operación y casos reales
- Enfoque para ingeniería y automatización
- Guía práctica para construir asistentes útiles

Referencias: <https://docs.openclaw.ai/> |  
<https://github.com/openclaw/openclaw>

# Qué problema resuelve OpenClaw

- Unifica canales de mensajería en un solo gateway
- Permite operar asistentes IA desde plataformas reales
- Mantiene control self-hosted de datos y configuración

Referencias: <https://docs.openclaw.ai/> |  
<https://github.com/openclaw/openclaw>

# Arquitectura de alto nivel

- Canales y plugins -> Gateway -> agentes y herramientas
- Control UI, CLI y APIs sobre el mismo plano de control
- Sesiones y routing centralizados

Referencias: <https://docs.openclaw.ai/gateway/index> |  
<https://docs.openclaw.ai/gateway/protocol>

# Componentes clave

- Gateway (runtime central)
- Agentes (sesiones y ejecución)
- Canales/plugins (Telegram, WhatsApp, Discord, etc.)

Referencias: <https://docs.openclaw.ai/> | <https://docs.openclaw.ai/channels>

# Flujo de arranque recomendado

- `openclaw onboard --install-daemon`
- `openclaw gateway status`
- `openclaw dashboard`

Referencias: <https://docs.openclaw.ai/start/getting-started>

# Configuración: archivo y principios

- Config principal en `~/.openclaw/openclaw.json`
- Validación estricta por schema
- Hot reload con modos `hybrid`, `hot`, `restart`

Referencias: <https://docs.openclaw.ai/gateway/configuration>

# Seguridad y acceso

- `dmPolicy` para controlar DMs (pairing/allowlist/open)
- Recomendado loopback + autenticación gateway
- Uso de `openclaw doctor` y troubleshooting oficial

Referencias: <https://docs.openclaw.ai/gateway/security> |  
<https://docs.openclaw.ai/gateway/doctor> |  
<https://docs.openclaw.ai/gateway/troubleshooting>

# Sesiones, contexto y continuidad

- Scopes de sesión por peer/canal
- Resets configurables
- Memoria y continuidad por flujo de conversación

Referencias: <https://docs.openclaw.ai/concepts/session> |  
<https://docs.openclaw.ai/gateway/configuration>

# Agentes y routing multi-agente

- Varios agentes con workspaces separados
- Binding por canal/cuenta
- Aislamiento para contextos personal/profesional

Referencias: <https://docs.openclaw.ai/concepts/multi-agent> |  
<https://docs.openclaw.ai/gateway/configuration-reference>

# Skills y extensibilidad

- Skills para tareas específicas reutilizables
- Marketplace/community vía ClawdHub
- Plugins para canales y toolchains externos

Referencias: <https://docs.openclaw.ai/plugins/community> |  
<https://clawhub.com>

# Herramientas y automatización

- Tool calling para web, shell, archivos, mensajería
- Cron jobs y hooks para trabajo en background
- Integración con workflows de QA/DevOps

Referencias: <https://docs.openclaw.ai/tools> |  
<https://docs.openclaw.ai/automation/cron-jobs> |  
<https://docs.openclaw.ai/gateway/configuration>

# APIs HTTP y operación remota

- OpenAI-compatible/Responses/tools endpoints
- Túneles SSH o Tailscale para acceso remoto seguro
- Diagnóstico con status, health y logs

Referencias: <https://docs.openclaw.ai/gateway/openai-http-api> |  
<https://docs.openclaw.ai/gateway/openresponses-http-api> |  
<https://docs.openclaw.ai/gateway/remote>

# Estructura de archivos en proyectos con OpenClaw

- `AGENTS.md` : reglas operativas del asistente
- `SOUL.md` , `USER.md` , `MEMORY.md` : identidad, contexto y continuidad
- `memory/YYYY-MM-DD.md` : bitácora diaria

Referencias: <https://github.com/openclaw/openclaw> |  
<https://docs.openclaw.ai/>

# Por qué estos archivos importan

- Separan comportamiento, contexto y memoria
- Reducen ambigüedad entre sesiones
- Mejoran seguridad al segmentar información

Referencias: <https://docs.openclaw.ai/gateway/security> |  
<https://docs.openclaw.ai/concepts>

# Casos de uso técnicos reales

- QA automation assistant (Playwright + reportes)
- DevOps assistant (health checks + runbooks)
- Support assistant multi-canal con políticas

Referencias: <https://docs.openclaw.ai/start/showcase> |  
<https://docs.openclaw.ai/channels>

# Buenas prácticas para equipos

- Empezar con alcance pequeño y medible
- Definir políticas de acceso antes de escalar
- Versionar configuración y revisar cambios

Referencias: <https://docs.openclaw.ai/gateway/configuration-examples> | <https://docs.openclaw.ai/gateway/security>

# Qué construir después de esta charla

- Un bot personal en 1 día
- Un bot de equipo en 1 semana
- Un catálogo de skills/productos en 30 días

Referencias: <https://docs.openclaw.ai/start/getting-started> |  
<https://clawhub.com>

# Cierre

- OpenClaw combina flexibilidad técnica y operación práctica
- La clave es arquitectura + seguridad + disciplina de ejecución
- Con buen diseño, puedes convertirlo en ventaja real de producto

Referencias: <https://docs.openclaw.ai/> |  
<https://github.com/openclaw/openclaw>