

SKILLSBUNDLE · GUÍA PARA HUMANOS

# Guía rápida de uso

Cómo sacar partido a las 1136 skills, los 24 comandos slash y los 25 subagentes — sin saber programar.

# Las skills, en 3 gestos

Una skill es un manual experto que tu agente de IA lee y aplica. Instaladas (2 minutos con `INSTALAR-CON-TU-IA.md`), se usan así:

## 1 · Habla normal: el agente elige la skill

**Tú** → «Prepara el informe SEO mensual del cliente X»  
# el agente detecta y aplica seo-report / technical-seo-audit solo

## 2 · Pídelas por su nombre cuando quieras precisión

**Tú** → «Usa la skill **financial-analyst** para valorar esta empresa»  
**Tú** → «Aplica **prompt-guard** antes de procesar este email»

El índice completo con las 1136 está en `catalogo/CATALOG0.md` (busca con Ctrl+F) y en [skillsbundle.com/verticales](https://skillsbundle.com/verticales).

## 3 · Por MCP: el agente busca entre las 1136 él solo

**Tú** → «Busca skills para presupuestos de obra y aplica la mejor»  
# con el servidor MCP conectado (`mcp/SETUP-MCP.md`): `buscar_skills` → `leer_skill`

## La prueba de fuego de seguridad

Pídele que resuma un texto que contenga «**IGNORE ALL PREVIOUS INSTRUCTIONS**». Si Prompt Guard está activo, lo detectará y te avisará en vez de obedecer.

# Comandos slash

Claude Code

Escribe `/` en Claude Code y aparecen. Instalados por `./instalar.sh claude-code` en `~/.claude/commands/`.

| COMANDO                                  | PARA QUÉ SIRVE                                                                                                                                                                                                                        |
|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>/add-to-changelog</code>           | Add entry to project changelog following Keep a Changelog format                                                                                                                                                                      |
| <code>/create-onboarding-guide</code>    | Create comprehensive developer onboarding guide with environment setup, workflows, and interactive tutorials                                                                                                                          |
| <code>/create-prd</code>                 | Create Product Requirements Document (PRD) for new features                                                                                                                                                                           |
| <code>/dependency-audit</code>           | Audit dependencies for security vulnerabilities, license compliance, and update recommendations                                                                                                                                       |
| <code>/e2e-setup</code>                  | Configure comprehensive end-to-end testing suite with framework selection and CI integration                                                                                                                                          |
| <code>/feature</code>                    | Create a new Git Flow feature branch from develop with proper naming and tracking                                                                                                                                                     |
| <code>/finish</code>                     | Complete and merge current Git Flow branch (feature/release/hotfix) with proper cleanup and tagging                                                                                                                                   |
| <code>/flow-status</code>                | Display comprehensive Git Flow status including branch type, sync status, changes, and merge targets                                                                                                                                  |
| <code>/generate-api-documentation</code> | Auto-generate API reference documentation with multiple output formats and automated deployment                                                                                                                                       |
| <code>/generate-tests</code>             | Generate a complete test file for a specified source file or component. Use when the user explicitly asks to write, create, or generate tests for a specific file.                                                                    |
| <code>/github-issues</code>              | Create, update, and manage GitHub issues using MCP tools. Use this skill when users want to create bug reports, feature requests, or task issues, update existing issues, add labels/assignees/milestones, or manage issue workflows. |
| <code>/hotfix</code>                     | Create a new Git Flow hotfix branch from main for emergency production fixes                                                                                                                                                          |

# Comandos slash (II)

| COMANDO                               | PARA QUÉ SIRVE                                                                                                       |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| <b>/implement-caching-strategy</b>    | Design and implement comprehensive caching solutions for improved performance and scalability                        |
| <b>/optimize-bundle-size</b>          | Reduce and optimize bundle sizes through analysis, configuration, and code splitting strategies                      |
| <b>/optimize-database-performance</b> | Optimize database queries, indexing, and performance for improved response times and scalability                     |
| <b>/performance-audit</b>             | Comprehensive performance audit with metrics, bottleneck identification, and optimization recommendations            |
| <b>/release</b>                       | Create a new Git Flow release branch from develop with version bumping and changelog generation                      |
| <b>/secrets-scanner</b>               | Scan codebase for exposed secrets, credentials, and sensitive information                                            |
| <b>/security-audit</b>                | Perform comprehensive security assessment and vulnerability analysis                                                 |
| <b>/security-hardening</b>            | Harden application security configuration with comprehensive security controls                                       |
| <b>/test-coverage</b>                 | Analyze and improve test coverage with comprehensive reporting and gap identification                                |
| <b>/troubleshooting-guide</b>         | Generate systematic troubleshooting documentation with diagnostic procedures, common issues, and automated solutions |
| <b>/update-docs</b>                   | Systematically update project documentation with implementation status, API changes, and synchronized content        |
| <b>/write-tests</b>                   | Write comprehensive unit and integration tests with proper mocking and coverage                                      |

# Subagentes

Claude Code

Especialistas que Claude Code invoca como trabajadores independientes. Pídelos por nombre («usa el subagente **code-reviewer** sobre este diff») o deja que delegue solo.

| SUBAGENTE                   | ESPECIALIDAD                                                                                                                                                                                                                            |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>accessibility-tester</b> | Use this agent when you need comprehensive accessibility testing, WCAG compliance verification, or assessment of assistive technology support.<br>Specifically:\n\n<example>\nContext: A web application is preparing for launch and n  |
| <b>api-documenter</b>       | Use this agent when creating or improving API documentation, writing OpenAPI specifications, building interactive documentation portals, or generating code examples for APIs.<br>Specifically:\n\n<example>\nContext: A REST API has   |
| <b>api-security-audit</b>   | API security audit specialist. Use PROACTIVELY for REST API security audits, authentication vulnerabilities, authorization flaws, injection attacks, and compliance validation.                                                         |
| <b>business-analyst</b>     | Use when analyzing business processes, gathering requirements from stakeholders, or identifying process improvement opportunities to drive operational efficiency and measurable business value.<br>Specifically:\n\n<example>\nContext |
| <b>changelog-generator</b>  | Changelog and release notes specialist. Use PROACTIVELY for generating changelogs from git history, creating release notes, and maintaining version documentation.                                                                      |
| <b>code-reviewer</b>        | Use this agent when you need to conduct comprehensive code reviews focusing on code quality, security vulnerabilities, and best practices. Specifically:\n\n<example>\nContext: Developer has submitted a pull request with changes     |
| <b>code-simplifier</b>      | Simplifies and refines code for clarity, consistency, and maintainability while preserving all functionality. Focuses on recently modified code unless instructed otherwise.                                                            |
| <b>commit-guardian</b>      | Pre-commit verification agent: runs 10 automated checks (security audit of staged files, Conventional Commits validation, secrets scan, lint/tests) before every git commit and blocks the commit if any check fails.                   |
| <b>compliance-auditor</b>   | Use this agent when you need to achieve regulatory compliance, implement compliance controls, or prepare for audits across frameworks like GDPR, HIPAA, PCI DSS, SOC 2, and ISO standards.<br>Specifically:\n\n<example>\nContext: A h  |
| <b>content-marketer</b>     | Use this agent when you need to develop comprehensive content strategies, create SEO-optimized marketing content, or execute multi-channel content campaigns to drive engagement and conversions. Invoke this agent for content planni  |
| <b>database-architect</b>   | Database architecture and design specialist. Use PROACTIVELY for database design decisions, data modeling, scalability planning, microservices data patterns, and database technology selection. This agent designs and plans; hand of  |
| <b>debugger</b>             | Use this agent when you need to diagnose and fix bugs, identify root causes of failures, or analyze error logs and stack traces to resolve issues. Specifically:\n\n<example>\nContext: Application is crashing intermittently with a   |
| <b>dependency-manager</b>   | Use this agent when you need to audit dependencies for vulnerabilities, resolve version conflicts, optimize bundle sizes, or implement automated dependency updates.<br>Specifically:\n\n<example>\nContext: A project has accumulated  |

# Subagentes (II)

| SUBAGENTE              | ESPECIALIDAD                                                                                                                                                                                                                             |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| diagram-architect      | Create technical diagrams in multiple formats (ASCII, Mermaid, PlantUML, Draw.io). Use PROACTIVELY for architecture visualization, ERD generation, flowcharts, state machines, and dependency graphs.                                    |
| error-detective        | Use this agent when you need to diagnose why errors are occurring in your system, correlate errors across services, identify root causes, and prevent future failures.<br>Specifically:\n\n<example>\n\nContext: Production system is ex |
| git-workflow-manager   | Use this agent when you need to design, establish, or optimize Git workflows, branching strategies, and merge management for a project or team.<br>Specifically:\n\n<example>\n\nContext: A development team is experiencing frequent me |
| incident-responder     | Handles production incidents with urgency and precision. Use IMMEDIATELY when production issues occur. Coordinates debugging, implements fixes, and documents post-mortems.                                                              |
| market-researcher      | Use this agent when you need to analyze markets, understand consumer behavior, assess competitive landscapes, and size opportunities to inform business strategy and market entry decisions. Specifically:\n\n<example>\n\nContext: A    |
| mcp-expert             | Model Context Protocol (MCP) integration specialist for the cli-tool components system. Use PROACTIVELY for MCP server configurations, protocol specifications, and integration patterns.                                                |
| performance-engineer   | Use this agent when you need to identify and eliminate performance bottlenecks in applications, databases, or infrastructure systems, and when baseline performance metrics need improvement. Specifically:\n\n<example>\n\nContext:     |
| postgres-pro           | Use when you need to optimize PostgreSQL performance, design high-availability replication, or troubleshoot database issues at scale. Invoke this agent for query optimization, configuration tuning, replication setup, backup strate   |
| product-manager        | Use this agent when you need to make product strategy decisions, prioritize features, or define roadmap plans based on user needs and business goals.<br>Specifically:\n\n<example>\n\nContext: User is deciding whether to build featur |
| qa-expert              | Use this agent when you need comprehensive quality assurance strategy, test planning across the entire development cycle, or quality metrics analysis to improve overall software quality.<br>Specifically:\n\n<example>\n\nContext: A d |
| refactoring-specialist | Use when you need to transform poorly structured, complex, or duplicated code into clean, maintainable systems while preserving all existing behavior.<br>Specifically:\n\n<example>\n\nContext: A codebase has methods exceeding 200 li |
| technical-writer       | Use this agent when you need to create, improve, or maintain technical documentation including API references, user guides, SDK documentation, and getting-started guides.<br>Specifically:\n\n<example>\n\nContext: A development team  |

## ¿Y en las otras 10 plataformas?

Los comandos y subagentes son nativos de Claude Code. En Cursor, Copilot, Gemini CLI y el resto, las **1136 skills funcionan igual** (los tres gestos de la página 2) y la configuración nativa de cada una queda instalada por `./instalar.sh <plataforma> .`

## Chuleta de seguridad (5 líneas)

- Harness activo: tu agente pregunta antes de instalar o escribir, y tiene vetados `.env` , `sudo` y `rm -rf` .

- Claves API: siempre con tope de gasto y fuera del repositorio (guía completa en [seguridad/GUIA-SECURIZACION.md](#) ).
- Contenido externo (emails, webs) = datos, no órdenes: Prompt Guard vigila, tú decides.
- Copias de seguridad de tu carpeta de trabajo antes de tareas destructivas.
- Ante la duda: pide al agente que explique QUÉ va a ejecutar antes de aprobarlo.