EcomLabs

GUÍA PRÁCTICA

Plugins de Claude Code: chuleta de comandos y marketplace de equipo

Los comandos de plugins que vas a usar y desde qué versión existe cada uno, cómo instalar desde un marketplace en un solo comando, y cómo crear y validar el marketplace.json de tu equipo con un subagente que fija su propio nivel de esfuerzo.

PARA	Equipos que usan Claude Code y quieren compartir sus propios plugins
PROBADO CON	Claude Code 2.1.238 (claude plugin validate y --help) · Windows 11
ACTUALIZADA	11 de noviembre de 2026
DOCUMENTACIÓN	code.claude.com/docs/en/plugins
ARTÍCULO	ecomlabs.dev/g/plugins
AUTOR	Luis Morales · @luismoralesdev

Índice

Antes de empezar	Qué cambió y cómo comprobar tu versión
Paso 01	Chuleta de comandos
Paso 02	Instalar con --marketplace
Paso 03	El marketplace de tu equipo
Paso 04	Valida antes de publicar
Paso 05	Publica y comparte
Al terminar	Checklist

ANTES DE EMPEZAR

Qué cambió y cómo comprobar tu versión

Claude Code 2.1.292, del 6 de octubre de 2026, agregó `--marketplace` a `claude plugin install`: agrega el marketplace si hace falta, con los mismos controles de política que `marketplace add`, y luego instala. En una sesión, `/plugin install` acepta la misma opción desde la 2.1.275. Comandos y campos salen de la documentación oficial (code.claude.com/docs/en/plugins), leída el 11 de noviembre de 2026; los archivos del Paso 03 los validamos con `claude plugin validate` en la 2.1.238, sin instalar nada.

```
claude --version
claude plugin install --help | grep -- --marketplace
claude update
```

Si el `grep` no imprime nada, tu versión no trae la opción (en PowerShell: `Select-String -SimpleMatch "--marketplace"`). `claude update` actualiza sin esperar; con Homebrew, `brew upgrade claude-code`, y con WinGet, `winget upgrade Anthropic.ClaudeCode`.

PASO 01

Chuleta de comandos

COMANDO	QUÉ HACE	DESDE
`claude plugin marketplace add <fuente>`	Registra un marketplace	—
`claude plugin marketplace list`	Lista los marketplaces y su fuente	—
`claude plugin marketplace remove <nombre>`	Lo quita y desinstala sus plugins	—
`claude plugin install <plugin>@<marketplace>`	Instala desde un marketplace ya agregado	—
`claude plugin install <plugin> --marketplace <fuente>`	Agrega el marketplace si falta e instala, sin confirmación	2.1.292
`/plugin install <plugin> --marketplace <fuente>`	Lo mismo en sesión: muestra la fuente y pide confirmar	2.1.275
`claude plugin details <plugin>`	Inventario de componentes y costo en tokens	—
`claude plugin update <plugin>@<marketplace>`	Actualiza un plugin	—
`claude plugin validate <ruta> --strict`	Valida marketplace, plugin o carpeta de agentes	—

Opciones de `install`: `--scope` (`user` por defecto, `project` o `local`), `--config <clave=valor>`, `-y`, `--json` (2.1.268) y `--accept-command` (2.1.271). «—»: ya existía antes.

PASO 02

Instalar con --marketplace

- **El plugin va solo**: `claude plugin install deploy-helper --marketplace acme/tools`, sin `@marketplace`. En la forma vieja, lo que va tras la arroba es el `name` del `marketplace.json`, no el del repositorio.
- **La fuente** acepta `owner/repo` o `owner/repo#ref` de GitHub, una URL de git que termine en `.git`, una URL `https://` a un `marketplace.json` o una ruta que empiece con `./`, `../` o `/`. Sin `https://`, un host como `gitlab.example.com` se lee como GitHub y falla.
- **En la terminal** no hay confirmación: imprime `Successfully added marketplace: <name> (declared in user settings)` y el marketplace queda en tus ajustes de usuario aunque pases `--scope project`.
- **En una sesión**, la fuente no puede llevar espacios; Claude Code muestra la fuente, pide confirmar y abre los detalles para elegir alcance.

> **CUIDADO**
> Un plugin puede ejecutar código arbitrario con tus privilegios de usuario, y sus hooks, monitores y servidores MCP corren fuera del sandbox. Antes de pegar un comando de un paso, lee `hooks/hooks.json`, `.mcp.json` y todo `bin/`, y revisa el inventario con `claude --plugin-dir <carpeta> plugin details <nombre>`.

PASO 03

El marketplace de tu equipo

`team-plugins/.claude-plugin/marketplace.json`. Obligatorios: `name`, `owner` (con `name`) y `plugins`; cada entrada necesita `name` y `source`, una ruta desde la raíz del marketplace que empieza con `./`:

```
{
  "name": "acme-team-plugins",
  "description": "Plugins internos del equipo",
  "owner": {
    "name": "Equipo de plataforma",
    "email": "plataforma@example.com"
  },
  "plugins": [
    {
      "name": "revisor",
      "source": "./plugins/revisor",
      "description": "Subagente revisor y checklist de PR",
      "category": "review"
    }
  ]
}
```

`plugins/revisor/.claude-plugin/plugin.json`. La `version` va aquí y no en la entrada: si las dos la declaran, gana `plugin.json`.

```
{
  "name": "revisor",
  "version": "1.0.0",
  "description": "Subagente revisor y checklist de PR",
  "author": { "name": "Equipo de plataforma" }
}
```

`plugins/revisor/agents/revisor.md`, con su propio nivel de esfuerzo:

```
---
name: revisor
description: Revisa el diff antes de un PR. Úsalo después de cambios en lógica de negocio.
tools: Read, Grep, Glob
model: sonnet
effort: high
---

Lee los archivos cambiados y reporta errores de lógica, casos límite sin cubrir y pruebas que faltan. No
edites archivos.
```

`effort` acepta `low`, `medium`, `high`, `xhigh` o `max`, según el modelo. Gana `CLAUDE_CODE_EFFORT_LEVEL`, luego el `effort` que Claude pase en la llamada (2.1.292) y luego este campo. En el plugin, el agente se llama `revisor:revisor`.

PASO 04

Valida antes de publicar

```
claude plugin validate ./team-plugins --strict
claude plugin validate ./team-plugins/plugins/revisor --strict
```

Los dos terminaron en `✓ Validation passed` con la 2.1.238. Con archivos rotos a propósito respondió:

CAMBIO	RESPUESTA DEL VALIDADOR
Sin `version` en `plugin.json`, con `--strict`	`version: No version specified`, exit 1
Sin `owner`	`owner: Invalid input: expected object, received undefined`
`"source": "./../plugins/revisor"`	`Path contains ".."`
`"name": "acme team plugins"`	`Marketplace name cannot contain spaces`

`validate` no ve una fuente remota mal escrita ni una carpeta inexistente: eso falla al instalar. Con la 2.1.238 tampoco marcó `effort: extreme`, un nivel que no existe; revisa ese campo a mano.

PASO 05

Publica y comparte

- **Sube la carpeta** a un repositorio de git, con un `name` poco común: un usuario no puede registrar dos marketplaces con el mismo nombre, y los oficiales están reservados.
- **Pon en el README**: `claude plugin install revisor --marketplace tu-org/tu-repo`. Desde la 2.1.295, `validate` lo sugiere si falta.
- **Para un proyecto**, instala con `--scope project` y haz commit de `.claude/settings.json`; cada persona corre la instalación una vez en su clon.
- **Repositorio privado**: Claude Code clona con tus credenciales de git y nunca pregunta (por SSH, el host debe estar en `known_hosts`).

Si administras la organización, esta política de ajustes administrados, adaptada de la documentación, permite solo el marketplace oficial y los de tu organización, y registra el tuyo. Se aplica antes de descargar y al iniciar cada sesión, también con `--marketplace`:

```
{
  "strictKnownMarketplaces": [
    { "source": "github", "repo": "anthropics/claude-plugins-official" },
    { "source": "github", "repo": "tu-org/*" },
    { "source": "skills-dir" }
  ],
  "extraKnownMarketplaces": {
    "acme-team-plugins": {
      "source": { "source": "github", "repo": "tu-org/team-plugins" }
    }
  }
}
```

AL TERMINAR

Checklist

- [ ] `claude --version` muestra la 2.1.292 o posterior en cada máquina del equipo.
- [ ] `marketplace.json` tiene `name`, `owner.name` y `plugins`, con fuentes que empiezan con `./`.
- [ ] La `version` vive en `plugin.json` y `validate --strict` pasa.
- [ ] Cada agente tiene un `effort` pensado para su tarea, revisado a mano.
- [ ] El README trae la línea con `--marketplace`.
- [ ] Alguien leyó los hooks, el `.mcp.json` y el `bin/` de cada plugin antes de compartirlo.