// TERMINAL.LOG

COMANDOS SECRETOS

12

de Claude Code

Los 12 comandos que casi nadie conoce y que van a cambiar cómo programas con IA.

DESLIZA →

$ _

// CONTEXTO

NO USAS CLAUDE CODE A TOPE.

La mayoría de devs usan 3 o 4 comandos y se quedan ahí.

Estos 12 son los que marcan la diferencia entre 'usar IA' y tener un sistema real de desarrollo con IA.

Integrados + Bundled Skills

v2.1.63+

// CMD_01 [INTEGRADO]

01

/insights

Tu reporte HTML sobre cómo usas Claude

PROBLEMA

No sabes si estás usando Claude Code bien o repitiendo patrones que te frenan.

SOLUCION

Analiza tus últimas 30 sesiones y te da quick wins para mejorar ya.

- `$ /insights`
  Genera reporte HTML con tus patrones de uso

→ **TIP**

Ejecútalo una vez al mes. Es como ir al médico: siempre sale algún consejo útil.

// CMD_02 [SKILL]

02

/batch

Cambios masivos en paralelo con agentes aislados

PROBLEMA

Migrar 40 ficheros a mano o en una sola sesión es lento y frágil.

SOLUCION

Descompone el trabajo en 5-30 unidades y lanza un agente por cada una.

- `$ /batch`
  `migra src/ de Solid a React`

→ **TIP**

Sé concreto. 'Reemplaza moment.js por dayjs en todo src/' > 'actualiza el proyecto'.

// CMD_03 [SKILL]

03 /simplify

Tres agentes revisan tu código antes del PR

PROBLEMA

Después de horas programando, no ves los patrones repetidos ni las fugas.

SOLUCION

Un agente para duplicación, otro para legibilidad, otro para rendimiento.

```
$ /simplify
enfócate en la eficiencia de memoria
```

→ TIP

Conviértelo en ritual antes de cada PR. Pillan cosas que tú ya no ves.

// CMD_04 [INTEGRADO]

04

/security-review

Auditoría de seguridad solo de tus cambios

PROBLEMA

Los SAST tradicionales hacen ruido. Los falsos positivos cansan.

SOLUCION

Claude entiende el contexto: si ya sanitizaste, no te grita. Menos ruido.

- `$ /security-review`
  Analiza el diff de tu rama actual

→ **TIP**

Combo ganador: /simplify + /security-review. Código limpio = auditoría limpia.

// CMD_05 [INTEGRADO]

05

/compact

Comprime el contexto y dirige qué debe recordar

PROBLEMA

A las 2 horas Claude 'se vuelve tonto'. No es magia: es el contexto lleno.

SOLUCION

Resume el historial al 50%. Y tú decides qué preservar y qué descartar.

- `$ /compact`
  `enfócate en el módulo de auth`

→ **TIP**

Añade 'Compact Instructions' en tu CLAUDE.md: lo crítico sobrevive siempre.

// CMD_06 [INTEGRADO]

06

/rewind

Tu máquina del tiempo para código y conversación

PROBLEMA

Claude se va por un camino que no querías. Deshacerlo a mano es un caos.

SOLUCION

Escape x2. Vuelves al punto exacto. Ficheros y conversación. Cero tokens.

- `$ /rewind`
  Atajo: pulsa Escape dos veces

→ **TIP**

Usa /rewind sin miedo. Prueba y error sin coste. Tu red de seguridad real.

// CMD_07 [INTEGRADO]

07

/context

Radiografía visual de tu ventana de contexto

PROBLEMA

Trabajas a ciegas. ¿Vas al 30% o al 85%? La diferencia importa mucho.

SOLUCION

Cuadrícula con colores. De un vistazo ves qué componente ocupa qué.

- `$ /context`
  Ver desglose: sistema, CLAUDE.md, tools

→ **TIP**

Regla: revisa cada 5-10 intercambios. Compacta al 70%. Limpia al 90%.

// CMD_08 [SKILL]

08 /debug

El detective de los logs internos de tu sesión

PROBLEMA

Una herramienta falló sin explicación. Un MCP se desconectó. ¿Por qué?

SOLUCION

Lee el log interno y lo analiza. Puedes describir el problema para enfocar.

- `$ /debug`
  `el MCP dejó de responder tras editar`

→ **TIP**

Antes de abrir issue en GitHub: ejecuta /debug. Probablemente te dé la respuesta.

// CMD_09 [INTEGRADO]

09

/hooks

Automatización por eventos del ciclo de vida

PROBLEMA

Formatear, testear, proteger ficheros sensibles… lo haces a mano siempre.

SOLUCION

12 eventos donde engancharte. Formateo, bloqueos, revisión semántica.

```
$ /hooks
PostToolUse: black tras cada Write
```

→ TIP

Empieza por un hook de formateo (riesgo cero). Después añade protección.

// CMD_10 [INTEGRADO]

10

/stats

Tu actividad al estilo gráfico de GitHub

PROBLEMA

No sabes cuándo eres más productivo con Claude ni qué modelos usas de más.

SOLUCION

Heatmap de uso, rachas, modelo favorito. Pulsa ’r’ para cambiar rango.

```
$ /stats
  Pulsa 'r' para cambiar de rango temporal
```

→ **TIP**

Combínalo con /insights. Stats = cuánto y cuándo. Insights = cómo y qué mejorar.

// CMD_11 [INTEGRADO]

11

/remote-control

Controla tu sesión desde el móvil o el navegador

PROBLEMA

Estás a mitad de refactor y te tienes que mover. ¿Pierdes el contexto?

SOLUCION

URL + QR. Sigues desde cualquier sitio. Tu máquina, tu config, otra pantalla.

- `$ /remote-control`
  Alias: /rc · requiere plan Max

→ **TIP**

Usa /rename antes para darle nombre. Así la identificas rápido en el móvil.

// CMD_12 [INTEGRADO]

12

/voice

Entrada por voz push-to-talk para el terminal

PROBLEMA

Sabes lo que quieres, pero escribirlo con precisión lleva más tiempo que decirlo.

SOLUCION

Espacio pulsado + habla + suelta. La transcripción aparece en el cursor.

- `$ /voice`
  Mantén espacio · habla · suelta

→ **TIP**

No cuesta tokens extra. Ideal para contexto de negocio y specs largas.

// RESUMEN

LOS 12 EN UN VISTAZO

01	**/insights**	análisis de uso
02	**/batch**	cambios masivos
03	**/simplify**	limpieza pre-PR
04	**/security-review**	auditoría de seguridad
05	**/compact**	comprime contexto
06	**/rewind**	deshace sesión
07	**/context**	radiografía de uso
08	**/debug**	diagnóstico
09	**/hooks**	automatización
10	**/stats**	estadísticas
11	**/remote-control**	desde el móvil
12	**/voice**	entrada por voz

// OUTPUT

EL FUTURO LIBERA.

La IA no te quita el trabajo. Te da las herramientas para trabajar como 10 personas.

→ GUARDA ESTO

Sigue @freelance.369_ia

FL369 STUDIO · EST. 2019

Building freedom through AI