GUÍA PRÁCTICA · EZELERO

MANUAL DE
Claude Opus 5

12 claves para escribir prompts
que sí funcionan con los modelos nuevos.

Ezelero

ANTES DE EMPEZAR

Tus prompts viejos ya no son óptimos

Claude Opus 5 todavía entiende los prompts de la versión anterior. Pero eso no significa que sigan siendo la mejor forma de trabajar. El modelo nuevo es **más autónomo**: planifica tareas complejas, controla su propio trabajo y usa herramientas o subagentes por su cuenta.

Por eso, muchas ayudas que antes servían hoy **estorban**. En los prompts de sistema de su herramienta de código, el equipo quitó **más del 80%** del texto anterior — sin pérdidas de calidad medibles. No hace falta reescribir todo, pero sí **aligerar y estructurar mejor**.

> **Cómo leer este manual:** cada clave trae una explicación, un ejemplo de lo que ya no conviene y la forma óptima de hoy. Al final: la regla de oro, un FAQ y cómo dar el primer paso con nosotros.

Vas a entender

por qué el modelo nuevo cambia las reglas del prompting.

Vas a poder

reescribir tus prompts para que sean más rápidos y baratos.

CLAVE 01 / 12

Menos reglas, más metas claras

En los modelos anteriores convenía dictar con precisión cómo debía resolver cada tarea. Reglas largas evitaban que el modelo cambiara archivos sin permiso o escribiera comentarios de más. Con Opus 5 eso genera problemas: el modelo debe considerar cada instrucción y resolver los conflictos entre ellas. Frases como «documentá el código adecuadamente» y «nunca agregues comentarios» se bloquean entre sí. Suele entregar un resultado igual, pero gasta más tiempo de cómputo y más tokens.

ANTES «No escribas comentarios de varias líneas. No crees archivos adicionales.»

MEJOR ASÍ **«Orientate en el código existente. Tomá su nomenclatura, estructura y densidad de comentarios.»**

Nota · La forma nueva describe el resultado deseado sin anticipar cada excepción. Los límites duros —seguridad o un formato de salida fijo— siguen siendo imprescindibles.

CLAVE 02 / 12

Sacá los “revisá todo” genéricos

Opus 5 controla y corrige su propio trabajo. Instrucciones como «revisá tu respuesta una vez más al final» o «usá un segundo agente para verificar el resultado» quedaron obsoletas. No mejoran el resultado de forma automática: disparan ciclos de verificación redundantes que suben el costo. Lo mismo vale para sistemas de agentes que lanzan un paso de verificación separado después de cada tarea.

ANTES «Controlá todo cuidadosamente.»

MEJOR ASÍ **«Compará los precios mencionados con la tabla adjunta.»**

Nota · Los pedidos de control concretos sí sirven: el modelo sabe exactamente qué revisar, en vez de repetir el proceso a ciegas.

CLAVE 03 / 12

Delimitá el alcance de la tarea

Más autonomía también trae riesgos. Opus 5 a veces interpreta la tarea más allá de lo previsto: agrega pasos extra o decide por su cuenta que otra solución es mejor. En tareas abiertas eso suele ser deseable. En cambios pequeños o flujos automatizados, provoca resultados inesperados y costo adicional. Por eso conviene fijar el alcance con exactitud.

ANTES «Mejorá esto.» (sin límites de alcance)

MEJOR ASÍ **Para tareas normales: «Trabajá solo la tarea descrita. Tomá decisiones de rutina menores por tu cuenta. No modifiques áreas adyacentes.»**

MEJOR ASÍ **Para cambios chicos: «Cambiá solo la sección indicada y dejá el resto sin modificar.»**

CLAVE 04 / 12

Decí explícitamente el largo

Opus 5 responde más extenso por defecto. El parámetro de esfuerzo solo controla cuánto piensa por dentro, así que un nivel más bajo no acorta de forma confiable la salida. Si necesitás una respuesta corta, tenés que pedir el largo directamente en el prompt. «Explicá los cambios de Opus 5» es demasiado impreciso.

MEJOR ASÍ

«Explicá los 5 cambios más importantes en máximo 500 palabras. Usá párrafos cortos y no repitas un resumen al final.»

MEJOR ASÍ

En documentos largos, para evitar capítulos artificiales: «Ajustá el largo al contenido real. Evitá resúmenes redundantes.»

CLAVE 05 / 12

Controlá los avisos de progreso

En tareas largas, Opus 5 suele explicar qué hará a continuación. Antes de llamar a una herramienta describe su plan y entrega estados intermedios. En procesos automatizados eso genera rápidamente demasiado texto. El prompt debería definir si el modelo reporta, cuándo y con cuánto detalle.

MEJOR ASÍ

Para pocos updates: «Antes de la primera llamada a herramienta, explicá en una frase qué vas a hacer. Después, solo un update corto si tenés que cambiar tu enfoque.»

MEJOR ASÍ

Para un proceso en silencio total: «Ejecutá los pasos necesarios sin explicaciones. Devolvé únicamente el resultado final.»

CLAVE 06 / 12

Subagentes solo para tareas paralelas

Opus 5 delega tareas a subagentes con más facilidad que los modelos anteriores. En análisis grandes o bases de código extensas es útil. En tareas chicas, la delegación multiplica el tiempo de ejecución y el costo — sobre todo cuando un paso depende de la respuesta del anterior y no se puede correr en paralelo.

MEJOR ASÍ

«Usá subagentes solo para tareas grandes, que puedan resolverse de forma independiente y en paralelo. Limitá su cantidad al mínimo necesario.»

CLAVE 07 / 12

Dá el contexto por partes

Antes era habitual meter la mayor cantidad de reglas, ejemplos e información en un prompt de sistema gigante o un CLAUDE.md enorme, por miedo a que el modelo no encontrara lo importante. Para la generación nueva, Anthropic recomienda lo contrario: entregar el contexto por etapas (lo llaman «progressive disclosure»). Los prompts generales quedan cortos; lo específico se carga solo cuando la tarea lo necesita.

- El prompt de sistema explica el objetivo general.
- El CLAUDE.md describe brevemente el proyecto y las particularidades reales que el modelo no deduce solo del repositorio.
- Los skills contienen instrucciones especializadas que se cargan solo en las tareas que las requieren.
- Los archivos de referencia guardan especificaciones, mock-ups, tests o ejemplos de la tarea actual.

Nota · Un único archivo con todas las reglas de desarrollo, tests, documentación y revisión suele ser peor para Opus 5 que varias fuentes puntuales que se consultan cuando hacen falta.

CLAVE 08 / 12

Eliminá las repeticiones

Las versiones viejas necesitaban ciertas instrucciones varias veces; por eso la misma regla aparecía tanto en el prompt de sistema como en la descripción de una herramienta. Con Opus 5 esas repeticiones se pueden quitar. Las reglas propias de una herramienta van en su descripción; el prompt de sistema no las repite.

- Para qué sirve la herramienta.
- Qué parámetros acepta.
- Qué valores están permitidos.
- Qué resultado devuelve.
- Qué límites importantes tiene.

Nota · Cuidado también con demasiados ejemplos predefinidos: pueden fijar a Opus 5 a un flujo concreto, aunque para una tarea puntual otro camino sea mejor. Se refiere sobre todo a ejemplos que le dictan cómo operar una herramienta paso a paso.

CLAVE 09 / 12

Referencias, no descripciones vagas

Opus 5 procesa muy bien material de referencia complejo. En vez de describir un objetivo de forma abstracta, entregale lo real: código existente, mock-ups en HTML, tests o archivos completos del proyecto.

ANTES «Creá una interfaz moderna, más o menos como la aplicación que ya tenemos.»

MEJOR ASÍ **«Guiate por dashboard.html: tomá sus espacios y su estructura de tarjetas. Usá los campos de datos de schema.ts.»**

Nota · La diferencia con la clave anterior: en la 8 hablamos de ejemplos innecesarios sobre el CÓMO. Acá se trata de referencias útiles sobre el QUÉ. Las referencias concretan el objetivo, el estilo o el comportamiento esperado — pero el modelo sigue eligiendo cómo resolverlo.

CLAVE 10 / 12

Probá niveles de esfuerzo bajos

El modelo ya entrega alta calidad en niveles de esfuerzo bajos y medios. No copies configuraciones viejas a ciegas; muchas veces un nivel más bajo alcanza.

- Esfuerzo bajo: extracciones y cambios bien acotados.
- Esfuerzo medio: análisis y desarrollo normales.
- Esfuerzo alto: depuración compleja y tareas de agente de largo plazo.

Nota · La mejor elección es el nivel más bajo que asegure de forma confiable la calidad que necesitás. Ahorra costo y, a menudo, también tiempo.

CLAVE 11 / 12

Reducí el razonamiento, no lo apagues

Opus 5 trabaja con el razonamiento activado por defecto. Anthropic recomienda controlar el costo con el parámetro de esfuerzo, en vez de desactivarlo por completo. Con el razonamiento apagado pueden aparecer artefactos: llamadas a herramientas escritas como texto (que no se ejecutan de verdad) o etiquetas XML internas, que en procesos de agente largos pueden incluso afectar pasos posteriores.

MEJOR ASÍ

Si igual lo apagás, no le prohíbas pensar. Mejor una regla general de salida: «No muestres etiquetas XML internas o del sistema.»

Nota · Para un uso confiable de herramientas, razonamiento activado con un nivel de esfuerzo bajo suele ser la opción más segura.

CLAVE 12 / 12

El cierre: la meta es el camino

Un buen prompt para Opus 5 no es automáticamente más corto. Tiene menos reglas de control genéricas y más información específica de la tarea. El error más grande al migrar no es reusar un prompt viejo — es arrastrar intactas todas las reglas, ejemplos, repeticiones y mecanismos de control del pasado. Eso puede volver a Opus 5 más lento, más caro y menos flexible.

MEJOR ASÍ

La regla nueva: «Describí la meta completa, fijá los límites importantes, entregá referencias de calidad y dejá que Opus 5 elija el camino más sensato dentro de ese marco.»

LA REGLA DE ORO

Un buen prompt hoy responde 5 preguntas

1. **¿Qué debe existir al final?** El resultado concreto que buscás.
2. **¿Qué información o referencias usar?** Datos, ejemplos y archivos reales.
3. **¿Qué límites duros no cruzar?** Seguridad, formato fijo, alcance.
4. **¿Qué largo y en qué formato?** Cuán extenso y con qué estructura.
5. **¿Qué requiere control o delegación?** Qué revisar y qué repartir.

> «Describí la meta completa, poné los límites, dá referencias de calidad y dejá que Opus 5 elija el camino más sensato dentro de ese marco.»

PREGUNTAS FRECUENTES

FAQ

¿Qué cambió con Claude Opus 5?

Trabaja más autónomo: planifica, revisa su trabajo y usa herramientas por su cuenta. Por eso muchas reglas de control, repeticiones y reglas menudas ya no hacen falta — e incluso consumen tokens de más.

¿Puedo seguir usando mis prompts viejos?

Sí, funcionan. Pero muchas veces ya no son óptimos. Conviene revisar y acortar sobre todo los prompts de sistema largos, los pedidos de verificación genéricos y las instrucciones de herramienta repetidas.

¿Por qué un prompt viejo puede dar peor resultado?

Porque contiene reglas pensadas para modelos menos autónomos. Opus 5 tiene que resolver las contradicciones entre ellas, lo que lo vuelve más lento, más caro y menos flexible.

¿Y AHORA?

Poné la IA a trabajar bien para vos

Escribir buenos prompts es solo una parte. En **Ezelero** implementamos la inteligencia artificial en tu negocio — con estrategia, límites y resultados. Escaneá y hablá con Robby.

Diagnóstico gratis

Escaneá con la cámara de tu celular

Escribí “IA” por WhatsApp

Ezelero

Inteligencia artificial para pymes · San Pedro Sula, Honduras · ezelero.com