Chuleta de modelos de Claude: cuál usar y cuándo

La regla corta: **Opus 5 para trabajo de verdad, Sonnet 5 cuando el volumen manda, Haiku 4.5 para tareas masivas y simples, Fable 5 solo cuando Opus se ha quedado corto.** Debajo tienes la tabla con precios, contexto y los ajustes que cambian tanto el resultado como la factura.

⬇️ Descargar en PDF (gratis, sin registro)

Los modelos, de un vistazo

Precios en dólares por millón de tokens. La **salida cuesta cinco veces más que la entrada** en todos ellos: es ahí donde se va el dinero.

Modelo	Contexto · precio	Cuándo usarlo
`claude-opus-5` Opus 5	1M · **$5** entrada / **$25** salida	**El de por defecto.** Programar, refactorizar, analizar, escribir lo que se firma. La mejor relación capacidad/precio para trabajo real.
`claude-sonnet-5` Sonnet 5	1M · **$3** / **$15** *($2/$10 hasta el 31-08-2026)*	Cuando el volumen manda. Calidad muy cercana a Opus en tareas acotadas por bastante menos dinero.
`claude-haiku-4-5` Haiku 4.5	200K · **$1** / **$5**	Tareas masivas y simples: clasificar, etiquetar, extraer un campo. Donde el criterio no importa y el volumen sí.
`claude-fable-5` Fable 5	1M · **$10** / **$50**	El más capaz. Solo cuando Opus se ha quedado corto de verdad: razonamiento muy duro y trabajo autónomo de horas.
`claude-opus-4-8` Opus 4.8	1M · **$5** / **$25**	La generación anterior de Opus. Sigue viva; útil como plan B si algo te funcionaba mejor ahí.

Qué modelo para qué tarea

Lo que vas a hacer	Modelo	Por qué ese
Programar una función, arreglar un bug	Opus 5	Es donde más se nota la diferencia. Termina el trabajo en vez de dejar el hueco a medias.
Refactor grande, varios ficheros, sesión larga	Opus 5 con effort alto	El trabajo de largo recorrido es justo su terreno. Dale la especificación completa de entrada.
Miles de llamadas iguales (clasificar, etiquetar)	Haiku 4.5	Cinco veces más barato en la salida y la calidad sobra para eso.

Lo que vas a hacer	Modelo	Por qué ese
Resumir, redactar, responder correos	Sonnet 5	Calidad de sobra y precio contenido. Opus aquí es pagar de más.
Analizar un documento muy largo	Cualquiera con 1M de contexto	Todos menos Haiku (200K). El límite es el contexto, no la inteligencia.
Un problema que ya se te ha atascado con Opus	Fable 5	El escalón de capacidad existe, pero cuesta el doble. Úsalo como excepción, no como norma.
Prototipar, cacharrear, aprender	Sonnet 5	Iteras más veces por el mismo dinero, y en esta fase iterar vale más que acertar a la primera.

Ajustes que cambian el resultado (y la factura)

Ajuste	Qué hace	Cuándo tocarlo
`effort`	Cuánto piensa y trabaja antes de responder: `low`, `medium`, `high`, `xhigh`, `max`. Por defecto `high`.	**La primera palanca, antes de cambiar de modelo.** Baja a `medium` para ahorrar; sube a `xhigh` en lo agéntico y de programación.
Pensamiento adaptativo	El modelo decide solo cuánto razonar antes de contestar.	Déjalo puesto. Desactivarlo ahorra menos de lo que parece y empeora más de lo que crees; para gastar menos, baja el `effort`.
`max_tokens`	Tope duro de la respuesta — **incluye lo que piensa**, no solo lo que escribe.	Súbelo si las respuestas se cortan a media frase. Con effort alto, dale sitio de sobra.
Caché de prompt	Reutiliza la parte fija del prompt entre llamadas, a una décima parte del precio.	Siempre que repitas un contexto grande. Es el ahorro más grande y el menos usado.
Modo rápido	El mismo modelo, respondiendo más deprisa, a precio premium.	Solo en Opus 5 y Opus 4.8, y solo si la latencia es lo que te duele.
Procesamiento por lotes	Mandas el trabajo y lo recoges luego, al 50%.	Cuando no necesitas la respuesta ya. La mitad de la factura por esperar un rato.

Señales de que te has equivocado de modelo

Lo que ves	Qué está pasando	Qué hacer
La factura sube y el trabajo es sencillo	Modelo de más para la tarea	Baja a Sonnet 5, o a Haiku si son miles de llamadas iguales.
Respuestas superficiales en problemas difíciles	`effort` demasiado bajo	Sube el `effort` antes de cambiar de modelo. Suele bastar.
La respuesta se corta a media frase	`max_tokens` corto: lo que piensa también cuenta	Súbelo. No es un problema del modelo.
Va lento y no es una tarea difícil	Effort alto donde no hacía falta	Baja a `medium` o `low`. Rinden mejor de lo que la gente espera.
Se pierde a mitad de un documento largo	Ventana de contexto agotada	Si es Haiku (200K), cambia a uno de 1M. Si ya es de 1M, parte el documento.
Error al llamar a la API con el nombre del modelo	ID inventado o con sufijo de fecha de más	Usa el ID exacto de la tabla. `claude-opus-5`, sin fecha detrás.

Errores típicos

- **Usar el modelo más caro para todo.** El reflejo de "el mejor por si acaso" multiplica la factura sin mejorar el resultado en tareas que Sonnet o Haiku resuelven igual de bien. Elige por tarea, no por miedo.
- **Comparar modelos solo por el precio de entrada.** La salida cuesta cinco veces más, y es la que crece cuando el modelo se enrolla. Compara siempre los dos números.
- **Cambiar de modelo cuando el problema era el `effort`.** Antes de saltar a uno más caro, sube el esfuerzo del que ya tienes. Es gratis probarlo y muchas veces es lo único que faltaba.
- **Inventarse el ID del modelo.** Añadir una fecha al final porque "los modelos llevan fecha" da un error 404 y diez minutos de desconcierto. El ID es literalmente el de la tabla.
- **No usar la caché de prompt teniendo un contexto fijo grande.** Si mandas el mismo documento o las mismas instrucciones en cada llamada, estás pagando diez veces lo que deberías.

Preguntas frecuentes

Si solo quiero acordarme de un modelo, ¿cuál?

Opus 5. Es el que mejor equilibra capacidad y precio para trabajo real: programar, analizar, escribir cosas que importan. Cambia solo cuando tengas un motivo concreto: Haiku si es masivo y simple, Fable 5 si te has estrellado con Opus.

¿Cuándo compensa Opus frente a Sonnet?

Cuando el trabajo es largo, toca varios ficheros o la respuesta se va a firmar con tu nombre. Sonnet 5 está muy cerca en calidad y cuesta bastante menos, así que para volumen alto y tareas acotadas suele ganar. La forma honesta de decidirlo es probar la misma tarea real en los dos.

¿Qué es el `effort` y cómo afecta al precio?

Es cuánto piensa y trabaja el modelo antes de responder: `low`, `medium`, `high`, `xhigh` y `max`. Por defecto va en `high`. Más effort son más tokens y más tiempo, pero en tareas agénticas a veces sale más barato porque resuelve en menos vueltas. Es la primera palanca que tocar, antes de cambiar de modelo.

¿Para qué sirve Haiku si es el menos capaz?

Para lo que haces miles de veces y no exige criterio: clasificar, etiquetar, extraer un campo, decidir si un texto es positivo o negativo. Ahí la diferencia de calidad no se nota y la de precio sí — cinco veces más barato que Sonnet en la salida.

¿Cómo sé cuánto me va a costar antes de lanzarlo?

Con el endpoint de conteo de tokens del propio modelo que vas a usar, no con calculadoras de otros proveedores: cada modelo cuenta distinto. Y mira siempre el precio de salida, no solo el de entrada — la salida cuesta cinco veces más.

Vídeos relacionados

- ¿Fable? ¿Sonnet? ¿Opus? Deja de usar el modelo equivocado en Claude — la guía completa en vídeo.
- Claude Sonnet 5 vs Opus 4.8: ¿cuál usar (y cuándo)? — el duelo, tarea por tarea.
- Deja de tirar tu dinero con Claude, haz esto en su lugar — dónde se te va la factura.
- GPT 5.6 vs Fable: OpenAI lanza 3 modelos y ataca por el precio — cómo queda frente a la competencia.

¿Quieres dejar de pedirle cosas sueltas a un chat y empezar a diseñar sistemas que trabajan solos? Eso es lo que enseñamos, paso a paso y con proyectos reales, en el curso de Frogames.

De Vibe Coder a Ingeniero Agéntico →