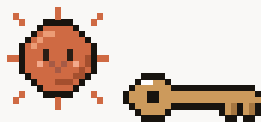


Claves de API, dónde van y dónde nunca



Una clave de API es la llave con la que un programa entra a tu cuenta. Quien la tiene, entra, y lo que use con ella puede generar cargos a tu nombre. Los ejemplos son de Claude, pero la regla vale para cualquier servicio.

1 Dónde se crea

En Claude Console (platform.claude.com) › Settings › API keys › Create key. Empieza por `sk-ant-` y se muestra una sola vez: cópiala en ese momento.

2 Dónde va: en una variable de entorno

Es un dato que tu equipo guarda fuera del código.

```
MAC Y LINUX
export ANTHROPIC_API_KEY="sk-ant-..."
```

```
WINDOWS · POWERSHELL
$env:ANTHROPIC_API_KEY = "sk-ant-..."
```

Las dos duran lo que dura esa ventana de la terminal. Otra opción es un archivo `.env`, con la línea `.env` en tu `.gitignore` (la lista de archivos que no se suben a tu repositorio).

3 Dónde nunca va

En un chat, en un correo, en un ticket de soporte, en el código de tu web ni en un repositorio público (la carpeta de tu proyecto subida a GitHub, a la vista de todos).

4 Si se filtró, bórrala hoy

En la Console, menú de tres puntos › Delete API Key, y crea otra. Anthropic sugiere rotarlas cada cierto tiempo (su ejemplo es cada 90 días) y revisar el uso de la cuenta.

Si GitHub encuentra una clave de Anthropic en un repositorio público, avisa a Anthropic, que la desactiva y te lo notifica por correo.

5 En Claude Code no la necesitas

Con una suscripción Pro, Max, Team o Enterprise, Claude Code funciona sin clave. Si tienes ANTHROPIC_API_KEY definida, pregunta una vez si usarla; si aceptas, trabaja con la clave (créditos de la API, aparte de tu suscripción). Para ver cuál está usando, escribe `/status`.

6 Si tu web está en Vercel

Guárdala como variable de entorno de tipo Secret: después de guardarla nadie puede volver a leerla. Úsala desde una función del servidor, nunca desde el HTML. Los cambios en variables se aplican en la siguiente subida.

Mi criterio

- Lo que llega al navegador lo puede leer cualquiera. Por eso la clave se usa solo en el servidor.
- Una revisión automática no reemplaza a una persona que sepa de seguridad, pero atrapa lo obvio.

7

Antes de publicar, pide una revisión

```
/security-review
```

En Claude Code, revisa los cambios de tu rama frente a la rama principal del remoto origin y busca riesgos como inyección, fallos de autenticación y datos expuestos. Necesitas el proyecto en Git, subido a un repositorio remoto, y tus cambios guardados con commit.

¿Sin Git? Pídelo en palabras: «revisa el proyecto buscando fallos de seguridad y claves expuestas». No es el mismo comando, pero sirve como primera pasada.

Repaso de 1 minuto

- ☐ Ninguna clave en el HTML, en el JavaScript que llega al navegador ni en el repositorio.
- ☐ El archivo .env está en .gitignore.
- ☐ Las claves que ya no usas, o que pegaste en algún chat, están borradas.
- ☐ Pasaste /security-review o, sin Git, la revisión pedida en palabras.

FUENTES · COMPROBADAS EL 8 DE OCTUBRE DE 2026

- 1 platform.claude.com/docs/en/get-api-key
- 2 support.claude.com/en/articles/9767949-api-key-best-practices-keeping-your-keys-safe-and-secure
- 3 code.claude.com/docs/en/authentication
- 4 [code.claude.com/docs/en/commands \(/security-review\)](https://code.claude.com/docs/en/commands (/security-review))
- 5 vercel.com/docs/environment-variables

El comando de PowerShell sigue la sintaxis que la documentación de Claude Code usa para otra variable. «Mi criterio» es opinión de Francisco Reyes.