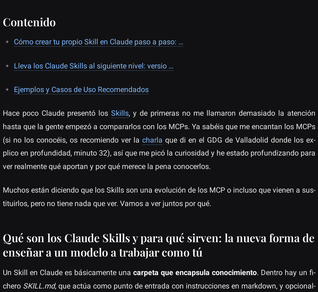
Introducción a los Claude Skills: qué son, cómo crearlos y por qué NO reemplazan a los MCP

12 de Enero de 2026

Introducción a los Claude Skills: qué son, cómo crearlos y por qué NO reemplazan a los MCP

Descubre cómo los Skills te permiten enseñar a Claude a generar outputs consistentes y repetibles, con ejemplos reales y casos de uso que puedes aplicar hoy mismo.

29 October 2025



Contenido

Hace poco Claude presentó los Skills, y de primeras no me llamaron demasiado la atención hasta que la gente empezó a compararlos con los MCPs. Ya sabéis que me encantan los MCPs (si no los conocéis, os recomiendo ver la charla que di en el GDG de Valladolid donde los explico en profundidad, minuto 32), así que me picó la curiosidad y he estado profundizando para ver realmente qué aportan y por qué merece la pena conocerlos.

Muchos están diciendo que los Skills son una evolución de los MCP o incluso que vienen a sustituirlos, pero no tiene nada que ver. Vamos a ver juntos por qué.

Qué son los Claude Skills y para qué sirven: la nueva forma de enseñar a un modelo a trabajar como tú

Un Skill en Claude es básicamente una **carpeta que encapsula conocimiento**. Dentro hay un fichero *SKILL.md*, que actúa como punto de entrada con instrucciones en markdown, y opcionalmente otros recursos adicionales en subcarpetas como resources, templates o incluso scripts de Python.

En ese *SKILL.md* defines qué hace el skill, cómo debe hacerlo y qué necesita para ejecutarse. Claude lo usa de forma progresiva: primero lee el resumen YAML para decidir si ese skill aplica a la tarea que le estás pidiendo y, si es así, carga las instrucciones completas y los recursos cuando los necesita.

Así mantiene eficiencia en el contexto y evita consumir tokens innecesarios. Si conoces bien los MCPs te sonará esta estructura que Anthropic está reutilizando.

El objetivo de un Skill no es conectar a Claude con datos externos ni darle memoria persistente, sino enseñarle a producir un tipo de output muy concreto con calidad, consistencia y repetibilidad. Es, por ejemplo, lo que le permite generar documentos, presentaciones o informes siguiendo el mismo formato y tono cada vez.

Cómo crear tu propio Skill en Claude paso a paso: estructura, tipos y ejemplos reales

Crear un Skill es bastante sencillo. La estructura básica es una carpeta con el SKILL.md y, si lo necesitas, subcarpetas con ficheros auxiliares. Esta sería la estructura correcta para un Skill

```
my-Skill.zip
└── my-Skill/
    ├── Skill.md
    └── resources/
```

Mientras que esta sería un ejemplo de estructura incorrecta:

```
my-Skill.zip
└── (files directly in ZIP root)
```

Hay dos **tipos** principales: los "Markdown-only" y los "code-enabled".

Los primeros se usan para tareas de generación de documentos como PowerPoints, hojas de cálculo o PDFs. Son básicamente **instrucciones escritas en texto** que guían a Claude sobre cómo estructurar el contenido. Podemos entenderlos como un GPT o un Gem.

Los segundos, en cambio, permiten **incluir scripts de Python**. Esto abre la puerta a automatizar tareas más complejas: análisis de datos, cálculos avanzados, generación de gráficos, procesado de imágenes o incluso validaciones automáticas. En este caso, Claude puede ejecutar ese código dentro de un entorno seguro y combinarlo con sus capacidades lingüísticas.

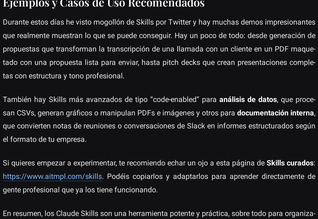
Un ejemplo que Anthropic enseñó es el de un skill que genera GIFs optimizados para Slack: incluye scripts que controlan el tamaño, número de frames y validan que no supere los 2 MB. Todo eso vive dentro del propio skill, como si fuera una miniaplicación empaquetada.

Y aquí es donde viene la confusión con los MCP. Aunque ambos amplían las capacidades de los modelos, lo hacen de formas totalmente distintas y con objetivos diferentes.

En qué se diferencian los Claude Skills de los MCP, Google Gems y GPTs

MCP es un protocolo estándar diseñado para conectar a los LLM con herramientas y fuentes de datos externas: bases de datos, APIs, sistemas internos...

Actúan como un **puente entre el modelo y el mundo exterior**. Un MCP define cómo intercambiar mensajes, recursos y acciones de manera interoperable entre distintos entornos y clientes.



Protocolo MCP

Los **Skills**, en cambio, no conectan a nada. No son una puerta al exterior, sino un **bloque de conocimiento especializado** que vive dentro del propio entorno del modelo. Si lo comparamos con los GPTs personalizados de OpenAI, los Skills serían algo así como un "**GPT en esteroides**": en lugar de limitarse a unas instrucciones estáticas, incluyen scripts, plantillas y lógica ejecutable. Pero **no almacenan contexto permanente** (para eso están los Claude projects) ni se integran por red, son más autocontenidos y pensados para **tareas muy concretas, repetibles y consistentes que buscan outputs estructurados muy similares.**

Dicho de otra forma: los MCP amplían el alcance de lo que el modelo puede ver y hacer fuera mientras que los Skills afinan cómo lo hace dentro.

Lleva los Claude Skills al siguiente nivel: versionado, APIs y agentes

Otro aspecto interesante es que los Skills pueden versionarse, compartirse e iterarse como si fueran proyectos de código. Si estás trabajando ya con proyectos serios te recomiendo subir los **Skills a repositorios**, probar nuevas versiones, documentar cambios y desplegarlos de forma controlada como si se tratase de una aplicación.

Anthropic incluso ha lanzado un endpoint para gestionarlos mediante **API**, de forma que puedan integrarse en flujos automatizados o entornos agenticos donde Claude actúe como un agente más dentro de un pipeline.

Esto significa que puedes construir un skill que genere reportes semanales, lo invoques desde la API y mantengas la misma salida consistente sin depender de prompts largos o impredecibles. La idea de "enseñar" al modelo nuevas habilidades mediante carpetas versionables tiene mucho sentido si pensamos en equipos que quieren mantener control y coherencia sobre sus outputs.

Ejemplos y Casos de Uso Recomendados

Durante estos días he visto mogollón de Skills por Twitter y hay muchas demos impresionantes que realmente muestran lo que se puede conseguir. Hay un poco de todo: desde generación de propuestas que transforman la transcripción de una llamada con un cliente en un PDF maquetado con una propuesta lista para enviar, hasta pitch decks que crean presentaciones completas con estructura y tono profesional.

También hay Skills más avanzados de tipo "code-enabled" para **análisis de datos**, que procesan CSVs, generan gráficos o manipulan PDFs e imágenes y otros para **documentación interna**, que convierten notas de reuniones o conversaciones de Slack en informes estructurados según el formato de tu empresa.

Si quieres empezar a experimentar, te recomiendo echar un ojo a esta página de **Skills curados**: https://www.aitmpl.com/skills. Podéis copiarlos y adaptarlos para aprender directamente de gente profesional que ya los tiene funcionando.

En resumen, los Claude Skills son una herramienta potente y práctica, sobre todo para organizaciones o usuarios que necesiten consistencia y control sobre tareas repetitivas.

Posiblemente veamos algo parecido en Gemini 3 con la evolución de sus "Gems" y la integración de llamadas a funciones, porque el patrón es evidente. Pero que no os lien: esto **no sustituye a los MCP**.

El MCP es un estándar abierto pensado para comunicación e interoperabilidad entre modelos y sistemas externos y los Skills son una forma de especializar el comportamiento interno de un modelo dentro de su propio entorno. Uno se enfoca en la conexión, el otro en la ejecución. Y entender esa diferencia es clave para no confundir herramientas que resuelven problemas totalmente distintos.