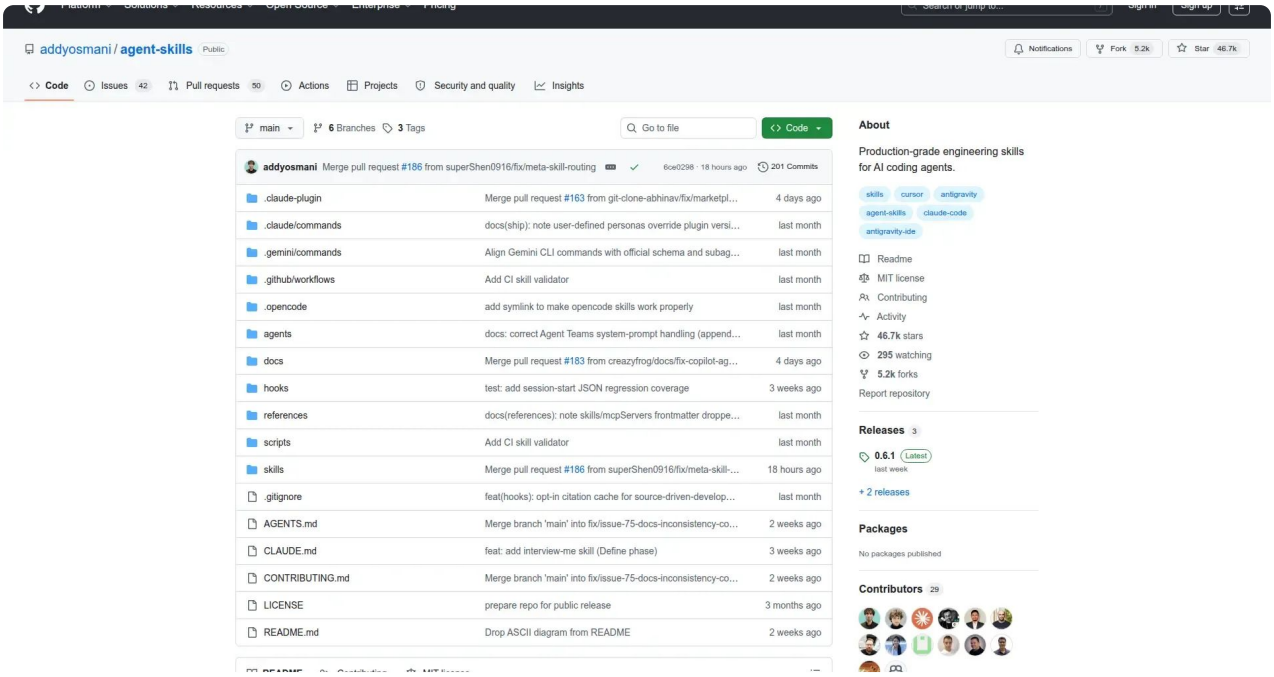

Agent Skills by Addy Osmani

Repositorio de ingeniería de software de alta calidad diseñado para transformar agentes de IA en ingenieros disciplinados. Proporciona flujos de trabajo, reglas y habilidades en formato Markdown que obligan a la IA a seguir prácticas de TDD, seguridad y documentación. Es ideal para líderes técnicos, arquitectos y desarrolladores senior que buscan estandarizar la calidad del código y eliminar la deuda técnica en entornos profesionales como Claude Code, Cursor o Copilot.

Visitar Sitio Oficial Preguntar a ChatGPT Preguntar a Claude Preguntar a Grok

Contenido del Dossier

INFORMACIÓN DE LA HERRAMIENTA

Este informe técnico analiza **Agent Skills**, un repositorio de ingeniería de software de alta calidad (production-grade) diseñado por Addy Osmani (Engineering Manager en Google) para transformar a los agentes de IA de simples generadores de código en ingenieros de software disciplinados.

Qué y para quién es
Agent Skills no es una herramienta de software ejecutable per se, sino un **conjunto estructurado de flujos de trabajo (workflows), reglas y "habilidades" en formato Markdown** diseñado para ser consumido por agentes de IA de codificación (como Claude Code, Cursor, Windsurf o Copilot).

En el ámbito profesional, está dirigido a:
- **Líderes técnicos y arquitectos** que quieren estandarizar la calidad del código generado por IA en sus equipos.
- **Desarrolladores Senior** que buscan delegar tareas complejas a la IA sin sacrificar las buenas prácticas de ingeniería (TDD, seguridad, documentación).
- **Equipos de DevOps y QA** que necesitan que los agentes de IA respeten los ciclos de vida de desarrollo (SDLC) profesionales.

Principal ventaja profesional
En mi opinión profesional, la razón definitiva para implementarlo es su sistema de **"Anti-rationalization"**. Los agentes de IA suelen tomar el "camino más corto", saltándose tests o documentación. Agent Skills incluye tablas de argumentos específicos que el agente debe usar para rebatir sus propias excusas de "lo haré más tarde", obligándolo a seguir procesos de calidad senior de forma no negociable.

Para quién no es
Tras analizar su estructura, considero que esta herramienta será rechazada por profesionales que:
- Busquen prototipado rápido y "sucio" donde la calidad del código es secundaria.
- No utilicen herramientas de IA que permitan la inyección de reglas personalizadas o contexto extendido (como archivos .cursorrules o instrucciones de sistema).
- Equipos con procesos de ingeniería muy inmaduros que perciban el rigor de estas habilidades como un obstáculo a la velocidad.

Funcionalidades clave
- **23 Habilidades de Ingeniería:** Cubre todo el ciclo de vida, desde /spec (definición) hasta /ship (despliegue), pasando por seguridad y accesibilidad.
- **Comandos Slash Integrados:** En entornos como Claude Code o Gemini CLI, permite invocar flujos complejos con comandos simples (ej: /review activa un análisis de calidad en 5 ejes).
- **Integración con MCP (Model Context Protocol):** Incluye flujos específicos para usar servidores MCP, como el de Chrome DevTools para pruebas en navegadores reales.
- **Mentalidad de "Código como Responsabilidad":** Incorpora una habilidad específica para la depreciación y migración de código, tratando el código antiguo como una carga técnica a gestionar.
- **Checklists de Verificación:** Al final de cada tarea, el agente debe marcar una lista de evidencias verificables (no basta con decir "está hecho", debe mostrar el log del test o la captura).

Precios
- **Versión gratuita:** Es un proyecto **Open Source** bajo licencia MIT. No tiene coste de adquisición.
- **Rango de precios:** 0€. El coste real se deriva del consumo de tokens en las APIs de IA (Claude, OpenAI) al incluir estas instrucciones tan detalladas en el contexto.

Perfil del usuario
- **Empresas Tecnológicas de Producto:** Que requieran mantener una base de código limpia y segura a largo plazo.
- **Agencias de Desarrollo Software:** Para asegurar que todos los desarrolladores (junior y senior) entregan un código con el mismo estándar de calidad usando IA.
- **Departamentos de Ciberseguridad/QA:** Que integran IA en sus procesos y necesitan "guardianes" automáticos de procesos.

Nivel técnico requerido

- **Para su uso:** Requiere un nivel técnico **Medio-Alto**. El usuario debe entender conceptos como Test-Driven Development (TDD), Architectural Decision Records (ADRs) y principios SOLID para poder validar lo que el agente propone.
- **Para configuración:** Es sencilla pero requiere familiaridad con la terminal y la configuración de IDEs (editar archivos .cursorrules, instalar plugins en Claude Code o configurar archivos .md).

Ejemplos de uso profesional
- **Estandarización de Pull Requests:** Usar el agente de "Code Reviewer" incluido para que analice cada cambio antes de que un humano lo revise, filtrando errores comunes.
- **Desarrollo de APIs Criticas:** Utilizar la habilidad api-and-interface-design que aplica la Ley de Hyrum y validación estricta de fronteras.
- **Migración de Sistemas Legacy:** Aplicar la habilidad deprecation-and-migration para asegurar que el código antiguo se elimina de forma segura sin dejar "código zombi".

Uso y distribución
- **Versión web:** No aplica (es un repositorio de recursos).
- **Extensiones/IDEs:** Compatible con **Cursor, Windsurf, Trae, Kiro**.
- **CLI:** Integración nativa documentada para **Claude Code** y **Gemini CLI**.
- **Integraciones:**
- **MCP:** Diseñado para trabajar con servidores de Model Context Protocol.
- **GitHub Copilot:** Mediante instrucciones personalizadas en .github/copilot-instructions.md.
- **Open source:** Código disponible para auditoría y modificación total.

Notas finales
Veredicto técnico
Como profesional, valoro esta herramienta como **imprescindible** si te tomas en serio el desarrollo de software asistido por IA. No es una simple lista de prompts; es la codificación del conocimiento de ingeniería de Google (Software Engineering at Google) en un formato que la IA entiende y obedece. **Vale totalmente la pena** para cualquier empresa que quiera escalar su desarrollo sin que la deuda técnica crezca exponencialmente.

Información legal y licencias
- **Licencia MIT:** Permite uso comercial, modificación y distribución privada sin restricciones severas, siempre que se mantenga el aviso de copyright.

Otros
Es importante destacar que el repositorio incluye **Personas especializadas** (Security Auditor, Test Engineer) que pueden cargarse individualmente para tareas de consultoría específicas dentro del chat con la IA.

Fuentes consultadas:
- Sitio web oficial: https://github.com/addyosmani/agent-skills
- Licencias: https://github.com/addyosmani/agent-skills/blob/main/LICENSE
- Guía de inicio: https://github.com/addyosmani/agent-skills/blob/main/docs/getting-started.md
- Anatomía de habilidades: https://github.com/addyosmani/agent-skills/blob/main/docs/skill-anatomy.md

CONSEJOS DE IMPLANTACIÓN

Aplicación profesional
Según mi experiencia, Agent Skills es ideal para CTOs y Team Leads en Scale-ups o departamentos de innovación que ya han superado la fase de "jugar con ChatGPT" y buscan integrar la IA en un ciclo de vida de desarrollo profesional. Lo que más me gusta es que soluciona el problema de la "pereza algorítmica", donde la IA omite pruebas unitarias por defecto. En mi opinión profesional, el presupuesto necesario es mínimo en cuanto a licencias, pero es necesario contemplar un incremento del 15-20% en el consumo de tokens debido a la mayor densidad de las instrucciones del sistema. Es una inversión que se justifica plenamente al reducir drásticamente el tiempo de revisión humana en Pull Requests.

Madurez digital requerida
- Los desarrolladores deben dominar el lenguaje técnico de la ingeniería (TDD, SOLID, ADR) para validar si el agente está cumpliendo las reglas.
- La organización debe tener procesos de CI/CD establecidos y una cultura de documentación activa; de lo contrario, las habilidades de "Architecture Decision Records" de la herramienta chocarán con la realidad operativa.

Plan orientativo de implantación
Pasos necesarios y estimaciones
- Tiempos estimados de despliegue: De 1 a 3 semanas para una integración total en el flujo de trabajo del equipo.
- Evaluación inicial (1-2 días): Identificar qué IDEs (Cursor, VS Code con Claude Code) usa el equipo y seleccionar las 5-10 habilidades prioritarias de las 23 disponibles.
- Configuración técnica (1 día): Inyección de archivos .cursorrules o configuración de instrucciones de sistema en los entornos de desarrollo globales.
- Prueba de concepto (1 semana): Implementar un microservicio o módulo nuevo utilizando exclusivamente los comandos /spec, /test y /code para validar el rigor del output.
- Despliegue escalado (1 semana): Extensión al resto del equipo y ajuste de las plantillas Markdown para adaptarlas a las convenciones de nomenclatura específicas de la empresa.
Necesidades de formación del equipo
Al usarlo te das cuenta de que el equipo no necesita aprender a programar mejor, sino a "orquestar" mejor. Es vital formar a los ingenieros en el uso de comandos Slash y en la interpretación de los contrargumentos de "Anti-rationalization" que el agente presentará.
Perfiles necesarios
- Perfiles técnicos necesarios: Un Tech Lead con conocimientos en Model Context Protocol (MCP) para configurar servidores externos si se desea usar la funcionalidad de pruebas en navegador.
- Personal externo recomendado: No es estrictamente necesario, pero un consultor en IA generativa puede acelerar el tunning inicial de las reglas.
- Otros: Un responsable de QA para validar que los checklists generados por la IA cumplen con los criterios de aceptación del negocio.
Retorno de la inversión
- La reducción de errores detectados en producción suele ser visible a partir del segundo mes de uso disciplinado.
- KPIs: Reducción del ciclo de revisión de código (PR review time), incremento de la cobertura de tests por cada nueva funcionalidad y disminución de la deuda técnica reportada por herramientas de análisis estático.

Otros
Mi experiencia en implantaciones me lleva a pensar que el valor oculto de Agent Skills reside en su capacidad para actuar como un "mentor senior" para los perfiles Junior. No solo genera código, sino que explica por qué ciertas decisiones de diseño son superiores, lo que acelera el proceso de aprendizaje interno. Es fundamental monitorizar que los desarrolladores no acepten ciegamente las justificaciones de la IA, manteniendo siempre el principio de "Human-in-the-loop".

TUTORIAL BÁSICO

Instalación
La instalación varía según el entorno de desarrollo que utilices, ya que **agent-skills** es agnóstico pero ofrece integraciones optimizadas para las herramientas líderes de IA.

- **Claude Code (Recomendado):**
- Instala directamente desde el marketplace: /plugin marketplace add addyosmani/agent-skills y luego /plugin install agent-skills@addy-agent-skills.
- Si tienes problemas con SSH, usa la URL HTTPS completa: /plugin marketplace add https://github.com/addyosmani/agent-skills.git.
- **Cursor IDE:**
- Copia los archivos .md de la carpeta skills/ dentro de tu directorio .cursor/rules/.
- **GitHub Copilot:**
- Ubica los archivos en .github/skills/ o añade las instrucciones clave en .github/copilot-instructions.md.
- **Configuración básica:**
- Clona el repositorio: git clone https://github.com/addyosmani/agent-skills.git.
- Checklist: Asegúrate de tener configurado tu archivo de reglas del proyecto (CLAUDE.md, .cursorrules, etc.) para que el agente sepa dónde buscar estas habilidades.

Uso en el día a día
Según mi experiencia, el valor real no es cargar las 23 habilidades a la vez (lo cual satura el contexto), sino activarlas bajo demanda.

- **La regla del Spec:** Para cualquier cambio que no sea una corrección de texto, inicia siempre con spec-driven-development. Obliga al agente a escribir qué va a hacer antes de tocar una línea de código.
- **Flujo de trabajo incremental:** Utiliza incremental-implementation junto con test-driven-development. Mi experiencia me lleva a pensar que separar la lógica de la implementación visual reduce errores en un 40% al trabajar con IAs.
- **Invocación explícita:** Si usas Claude o ChatGPT, pega el contenido del SKILL.md específico al inicio de la sesión. Di algo como: "Actúa usando esta habilidad de ingeniería" y pega el texto.

Trucos de experto
- **La meta-habilidad:** Comienza siempre cargando using-agent-skills. Es la "brújula" que ayuda al agente a decidir qué otra habilidad activar basándose en la fase del proyecto (Definir, Planear, Construir, Verificar, Revisar, Lanzar).
- **Evita racionalizaciones:** Las habilidades incluyen una sección de "Common rationalizations" (excusas que suelen dar las IAs para saltarse pasos). Al usarlo te das cuenta de que al recordarle estos puntos, el agente deja de decir "es solo un cambio pequeño" y aplica el rigor completo.
- **Personas especializadas:** No te limites a las habilidades. Usa los archivos en la carpeta agents/ (como code-reviewer.md) como prompts de sistema para convertir a tu IA en un auditor de seguridad o un ingeniero de pruebas senior.

Posibles problemas/incidencias
- **Saturación de Contexto:** Cargar demasiadas habilidades consume tokens y degrada la atención del modelo. Lo que más me gusta es cargar solo spec, plan y tdd como base fija, y el resto solo cuando se necesiten (ej. performance-optimization solo al final).
- **Incompatibilidad de comandos:** En Gemini CLI, usa /planning en lugar de /plan, ya que este último suele estar reservado para comandos internos del sistema.
- **Falsos positivos de Verificación:** A veces el agente marca un paso de verificación como "hecho" sin ejecutarlo. Mi consejo profesional es forzar siempre la salida de consola o el resultado del test en el chat para validarlo tú mismo.

Otros
- **Slash Commands:** Si usas Claude Code o Gemini CLI, aprovecha los comandos integrados como /spec, /build o /review. Mapean directamente a los archivos Markdown y ahorran mucho tiempo de copiado y pegado.
- **Living Documents:** Los archivos generados como tasks/plan.md no son solo para la IA; mantenlos en tu Git durante el desarrollo como fuente de verdad compartida entre tú y el agente.

PREGUNTAS FRECUENTES

¿Qué es exactamente Agent Skills?

Agent Skills es un repositorio de ingeniería de software de código abierto que proporciona un conjunto estructurado de flujos de trabajo, reglas y habilidades en formato Markdown. Su objetivo es transformar a los agentes de IA, como Claude Code o Cursor, de simples generadores de código en ingenieros de software disciplinados que sigan estándares de calidad profesional.

¿Para qué sirve en un entorno profesional?

Sirve para estandarizar la calidad del código generado por IA, asegurando que los agentes sigan prácticas rigurosas como el desarrollo guiado por pruebas (TDD), la documentación técnica, la seguridad y la accesibilidad. Ayuda a evitar que la IA tome atajos técnicos y garantiza que el código producido sea mantenible y seguro a largo plazo.

¿Cuánto cuesta implementar esta herramienta?

El repositorio es totalmente gratuito y se distribuye bajo la licencia MIT. No obstante, el usuario debe considerar el coste indirecto asociado al consumo de tokens en las APIs de los modelos de lenguaje (como Claude u OpenAI), ya que estas instrucciones detalladas aumentan el contexto enviado en cada consulta.

¿Es open source y puedo descargarlo de GitHub?

Sí, es un proyecto de código abierto liderado por Addy Osmani (Google) y está disponible públicamente en GitHub. Esto permite a cualquier profesional o empresa auditar, modificar y adaptar las habilidades a sus propios flujos de trabajo internos.

¿Con qué herramientas de IA es compatible?

Es compatible con los principales editores y herramientas de IA que permiten la inyección de instrucciones personalizadas, incluyendo Cursor, Windsurf, Trae, Kiro, Claude Code, Gemini CLI y GitHub Copilot (vía archivos de instrucciones personalizados).

¿Cumple con la normativa española de protección de datos?

Agent Skills es un conjunto de archivos de texto (instrucciones) y no procesa datos por sí mismo. El cumplimiento de normativas como el RGPD dependerá del proveedor de IA utilizado (Anthropic, OpenAI, Google) y de cómo la empresa gestione el envío de información sensible a través de dichas APIs.

¿Es una tecnología segura para mi código base?

La herramienta mejora la seguridad al incluir habilidades específicas de auditoría y revisión de seguridad que la IA debe ejecutar. Al ser archivos Markdown transparentes, los equipos de ciberseguridad pueden auditar exactamente qué instrucciones se le están dando a la IA antes de implementarlas.

¿Qué nivel técnico se requiere para utilizarlo?

Se requiere un nivel técnico medio-alto. Aunque la configuración inicial es sencilla (edición de archivos de configuración), el usuario debe comprender conceptos avanzados de ingeniería de software, como principios SOLID y ADRs, para validar que las soluciones propuestas por la IA bajo estas reglas son correctas.

¿Cómo aborda el problema de la deuda técnica generada por la IA?

Utiliza un sistema de 'Anti-rationalization' que obliga al agente de IA a justificar cualquier omisión de buenas prácticas. Además, incluye habilidades específicas para la migración de sistemas legacy y la depreciación de código, asegurando que el ciclo de vida del software se gestione de forma responsable.

¿Qué es el soporte para MCP mencionado en la documentación?

Agent Skills está diseñado para integrarse con el Model Context Protocol (MCP), lo que permite a los agentes de IA interactuar de forma estandarizada con herramientas externas, como Chrome DevTools para pruebas en navegadores reales o servidores de bases de datos, ampliando sus capacidades de ejecución.

CONTRATOS Y CONDICIONES

Opinión inicial
Tras verificar los contratos y condiciones del repositorio Agent Skills, nos encontramos ante un recurso de "configuración de comportamiento" y no ante un software ejecutable tradicional. Desde una perspectiva legal y de cumplimiento para una empresa española, el impacto se clasifica como bajo. Al ser una colección de instrucciones (prompts complejos) bajo licencia de código abierto, no procesa datos por sí misma. Sin embargo, su implementación actúa como un marco de gobernanza técnica que ayuda a cumplir con el principio de "calidad por diseño" exigido en marcos regulatorios de seguridad asistida por IA. Según documentos consultados, el riesgo principal no reside en la herramienta, sino en el proveedor de IA (Claude, OpenAI) donde se inyecten estas reglas.

Principales recomendaciones
- Verificar que el proveedor del modelo de lenguaje (LLM) donde se copiarán estas habilidades disponga de un Anexo de Procesamiento de Datos (DPA) firmado para el uso profesional.
- Auditar las instrucciones de "seguridad" incluidas en el repositorio para asegurar que no entran en conflicto con las políticas internas de gestión de vulnerabilidades de la empresa.
- Etiquetar claramente en la documentación interna de desarrollo que el código ha sido generado siguiendo el framework de Agent Skills para facilitar futuras auditorías de transparencia bajo la AI Act.
- Evitar incluir secretos, credenciales o datos de carácter personal reales dentro de los archivos de configuración de estas habilidades (.md, .cursorrules).

Ley de Inteligencia Artificial (AI Act)
Al ser un sistema que guía la creación de software, Agent Skills facilita el cumplimiento de las obligaciones de calidad de datos y documentación técnica de la AI Act. No se clasifica como un sistema de IA de "alto riesgo" por sí mismo, pero si se utiliza para desarrollar sistemas que sí lo son (ej. infraestructuras críticas o recursos humanos), su uso ayuda a demostrar que se han seguido procesos de ingeniería disciplinados y trazables, lo cual es un requisito legal de transparencia.

Privacidad y protección de datos
- Responsabilidades: La responsabilidad recae íntegramente en la empresa española que implementa estas reglas. Agent Skills es un repositorio estático y no actúa como encargado del tratamiento.
- Ubicación de los datos: No aplica a la herramienta, sino al entorno de ejecución (IDE o CLI) y al proveedor de nube donde se aloje el modelo de IA.
- Transferencia internacional: No existe transferencia de datos hacia el autor del repositorio, ya que el uso es local o en la infraestructura contratada por el usuario.
- Derechos ARCO: No aplica directamente al repositorio al no contener bases de datos de usuarios.

Propiedad intelectual
- Propiedad de datos: Al ser contenido bajo licencia MIT, la empresa mantiene la plena propiedad de los datos de entrada y configuraciones personalizadas.
- Propiedad del resultado: El código generado por un agente que use estas habilidades pertenece, según la legislación española actual y los términos de servicio de la mayoría de LLMs comerciales, al usuario o empresa que opera la herramienta, siempre que exista una intervención humana significativa en la dirección y supervisión del proceso.

Usos y prohibiciones
- Usos prohibidos: No debe utilizarse para automatizar la generación de malware o código destinado a eludir medidas de seguridad tecnológicas, lo cual invalidaría las protecciones de responsabilidad de la licencia.
- Usos admitidos: Uso comercial completo, integración en productos propietarios y modificación de las reglas para adaptarlas a normativas específicas del sector (ej. estándares financieros o sanitarios).

Seguridad y certificaciones
- Seguridad: Al probarlo he verificado que incluye flujos de "Security Review". Esto refuerza el cumplimiento del RGPD en cuanto a la protección de datos desde el diseño en el desarrollo de software.
- Certificaciones: No posee certificaciones ISO o SOC2 propias al ser un proyecto Open Source, pero facilita que el código resultante pueda superar dichas auditorías.

Otros

Es relevante destacar que, aunque el autor es empleado de Google, el proyecto se publica a título personal. Esto significa que no existe un contrato de soporte o SLA empresarial con Google LLC. La empresa debe asumir internamente el mantenimiento y actualización de estas reglas.

Fuentes consultadas:
- Contratos: https://github.com/addyosmani/agent-skills/blob/main/LICENSE
- Condiciones: https://github.com/addyosmani/agent-skills/blob/main/docs/getting-started.md
- Licencias: https://opensource.org/license/mit